Avoiding the 10 Worst Practices of SAP Development Projects

by Amy Stapleton, Founder, Hutch.AI

Although it is commonly believed that unsuccessful SAP projects fail due to a lack of time, money, or people, research shows that this is rarely the case. They typically fail for other reasons, like faulty business practices. This article zeroes in on the 10 worst practices. Keeping these in mind during your project will provide a strong safety net.

It is a common misconception that unsuccessful SAP projects, regardless of purpose or scale, fail due to a lack of either time, money, and/or people. Research shows that this is rarely the case.

Projects typically fail for other reasons, like faulty business practices. Over the years I have compiled my own list of the worst practices of SAP development projects, based on the mistakes and failures I've seen occur firsthand during the past 10 years. While there are obviously an infinite number of things that can go wrong, in this article I have zeroed in on the 10 worst — the ones that are, in my opinion, the most common, potentially detrimental, and above all avoidable. I'll also offer some guidelines to help you avoid the pitfalls I have often seen lead to failure. (Update: This article was published at about the same time as the Agile Manifesto was being penned and before Agile practices became mainstream. While this article does not refer to Agile, many of the concepts are tightly aligned with Agile thinking and are still relevant in today's software development environment.)

The observations about project practices in these pages are generic enough to apply to any SAP development initiative, regardless of what releases or modules are involved. In fact, you'll find that many elements also apply to any sizable IT project. While some of these observations may seem obvious at first glance, I think that you will find, as an intertwined whole, they will provide a strong safety net for your next project.

Now let's examine the 10 worst practices and what you can do to avoid falling victim to them.

Worst Practice #1: Proceeding Without Clear Project Objectives

SAP development projects are sometimes handed off to the IT department, while at other times they become the responsibility of business departments. IT teams are generally given the lead on projects that appear on the surface to be primarily "technical" — such as building interfaces to another system, or upgrading to a new release. Business teams tend to take the lead when business process expertise is involved, such as creating new transactions or complex workflows.

In either case, the worst practice (but unfortunately not an uncommon one) is to initiate the development effort without heeding the principles of good project management. Industry best practices tell us to start every project by documenting its objectives in a project proposal (or project charter), yet we all know how bothersome this effort can be.

Especially in today's economy, it would be unwise to begin any mid-sized or

The 10 Worst Practices of SAP Development Projects

1. Proceeding without clear project objectives.
2. Believing in the existence of "as-is" process definitions.
3. Not following a best practices process.
4. Paying for consultants, not your own people, to get smarter.
5. Making it difficult to retain project knowledge.
6. Underestimating the importance of communicating with end users.
7. Failing to secure and maintain executive buy-in.
8. Poorly documenting and tracking code modifications.
9. Inability to see and understand the entire project portfolio.
10. Not learning from your mistakes.

large development effort without first establishing the right foundation by understanding the basic objectives of the endeavor. Technology is never implemented for its own sake, but rather to support the business in achieving its short- and long-term goals. If an organization has not clearly defined its goals, then chances are high that any IT project will be a waste of money.

When the IT department is responsible for the entire development project, the project manager must identify the project sponsors and take the time to understand exactly what benefits the business expects to gain as a result of the project. Even if the project appears to be purely technical in nature, you need to tie the initiative to tangible business results. For example, suppose your project involves archiving obsolete data to reduce the size of the operational database. Even this "purely technical" project has clear business benefits that you can spell out:

- Response times for mission-critical transactions will be reduced, resulting in operational and/or time savings that you can at least approximate
- The reduction in data volumes will eliminate the need for additional storage media, lowering hardware expenditures.

Wherever possible, be sure to tie these and other resulting benefits back to the organization's strategic objectives.

Development efforts that lack strong project proposals with crystal-clear business objectives are much more likely to fail. For example, poor project proposals contain vague and/or unjustified objectives, such as:

- We are implementing a data warehouse because the finance department wants the ability to generate some additional reports.
- We are developing this custom transaction because it existed in the old system and the users want to have it in SAP.

As you can see, it simply makes good business sense to require all SAP development projects to submit a detailed project proposal that clearly outlines the project's major objectives. Without a project proposal, you will find it impossible to:

- Prioritize a list of projects and choose the ones that will produce the most positive impact for the business.
- Define the project scope to ensure that valuable time and money are not wasted.
- Measure the results of the project after it is completed, to ensure that it was successful.

As you develop the project proposal, keep in mind that you also need to determine and explicitly record the tangible output of the project effort. Be sure to consider specific metrics to measure after the project is complete that will show how your development efforts had a positive impact on the business.

One way to facilitate the creation of good project proposals is to make it a step in a methodology that you use for every project. Why do you need a methodology? Because you want your project teams to adhere to ways of doing things that are consistently repeatable and that will reduce the risk of project failure. As a group undertaking, this task can also serve as an effective team-building exercise. Ensuring that a good project proposal exists before getting started can also help you avoid many other problems, as you will see when I explore the rest of the10 worst practices list.

Table 1 shows an example of a project proposal outline, highlighting the major elements that you should address.

Worst Practice #2: Believing in the Existence of "As-Is" Process Definitions

If you believe in the Tooth Fairy, you may also believe your business analysts

Project proposal component	Description
Business objectives	Clearly state the major business objectives of the development effort. These objectives must be understood by both the business and IT teams, and aligned with the organization's strategic goals.
Project scope	List the business and technical areas that the project will affect, and identify what is beyond the scope of the project.
Benefits	You must be able to clearly articulate the benefits of the project before proceeding. These benefits must not be purely technical in nature.
Project output	Describe what system, reports, or functionality the project will produce.
Primary risks	Identify the major risks that could prevent the success of the project.
Estimated time scale and costs	Provide a rough estimate of the project time frame and costs. This information is important for ranking this project against other project opportunities.

Table 1 Sample project proposal outline

when they tell you that they know exactly how the current business process (the one that your development effort will interface with, or somehow impact) works. As a matter of fact, they will probably assure you that they have the process written down somewhere, and promise to email the latest version to you long before your development project gets under way.

You certainly do not want to accuse anyone of telling outright falsehoods. It is just that from what I've seen, 95 percent of all mission-critical business processes in all corporations throughout the world are not clearly documented, so what are the chances that the ones you need to know about will be?

One of the most common causes of failure for SAP development projects is that the IT team is forced to aim at an unclear and moving target. The current business process that the team is trying to replace, improve, or interface with was never well documented. As a result, the way the business works "now" only begins to come into focus as the development effort proceeds. The way the process is supposed to work upon project completion is even more tenuously defined. The more business processes that are being affected by the project, the more compounded the problem becomes, so the development effort inevitably stalls, waits for the picture to clear up, moves forward again, stalls, and sometimes is ultimately canceled altogether.

While there is no quick-and-easy way to tackle the job of mapping your current business processes, the task is not impossible. The knowledge of how your business works is distributed across many people in your organization. The lesson for the IT team is that regardless of how far removed it may seem from your sphere of responsibility, you will need to work hard to document, or at least encourage and support the documentation of, your business processes. Referring back to worst practice #1 (proceeding without clear project objectives), having the information required to put together a thorough project pro- posal is a good start. If your team is having difficulty articulating the objectives, scope, and benefits of the project, beware — this may be a sign that the current business processes that affect your project are unclear.

>> Note

The Three Basics Steps of Process Mapping

1. Conduct interviews with process owners to gather raw information.
2. Create a visual representation (or map) of the process.
3. Store the process map in a central, preferably searchable, place.

Make the task of mapping your processes faster and easier by approaching the task as a series of sequential steps. First, you need to conduct interviews with the process owners. In my experience, it often helps to prepare questionnaires that solicit the type of process information you need. There are many tools that enable you to create questionnaires online, route them to process owners, and consolidate the results automatically. And you can certainly create these types of documents on your own and distribute them to the appropriate individuals via email.

The goal of these questionnaires is to jump-start the often cumbersome work of describing an existing process, and to glean enough information about the process so you know where to drill down for more detail later. So be sure to ask questions like:

- Who are the suppliers and customers of the business process?
- What is the core process, and what are the primary subprocesses?
- Who are the process owners within the organization?
- What are the major inputs and outputs of the process?
- What are the customer requirements for the process outputs (sometimes referred to as "Critical to Quality" or "CTQ" factors)?
- Which IT systems currently support the subprocesses?
- What are the major integration points with other business processes or subprocesses?
- What metrics are currently used to measure the effectiveness of the process?
- What external parties are participants in the process?

In **Figure 1**, you can see a portion of a sample online questionnaire or survey form, designed to facilitate the definition of a distribution process.

You may also need to follow up on the questionnaires with face-to-face meetings.

After gathering the raw information, you can attempt to sketch a complete flowchart of the business process. If creating a flowchart seems daunting, you could start by creating an input-output process map in a spreadsheet, or by designing a use-case table. These methods are both good precursors to creating a detailed flowchart.

Finally, since defining business processes is an arduous task, be sure to

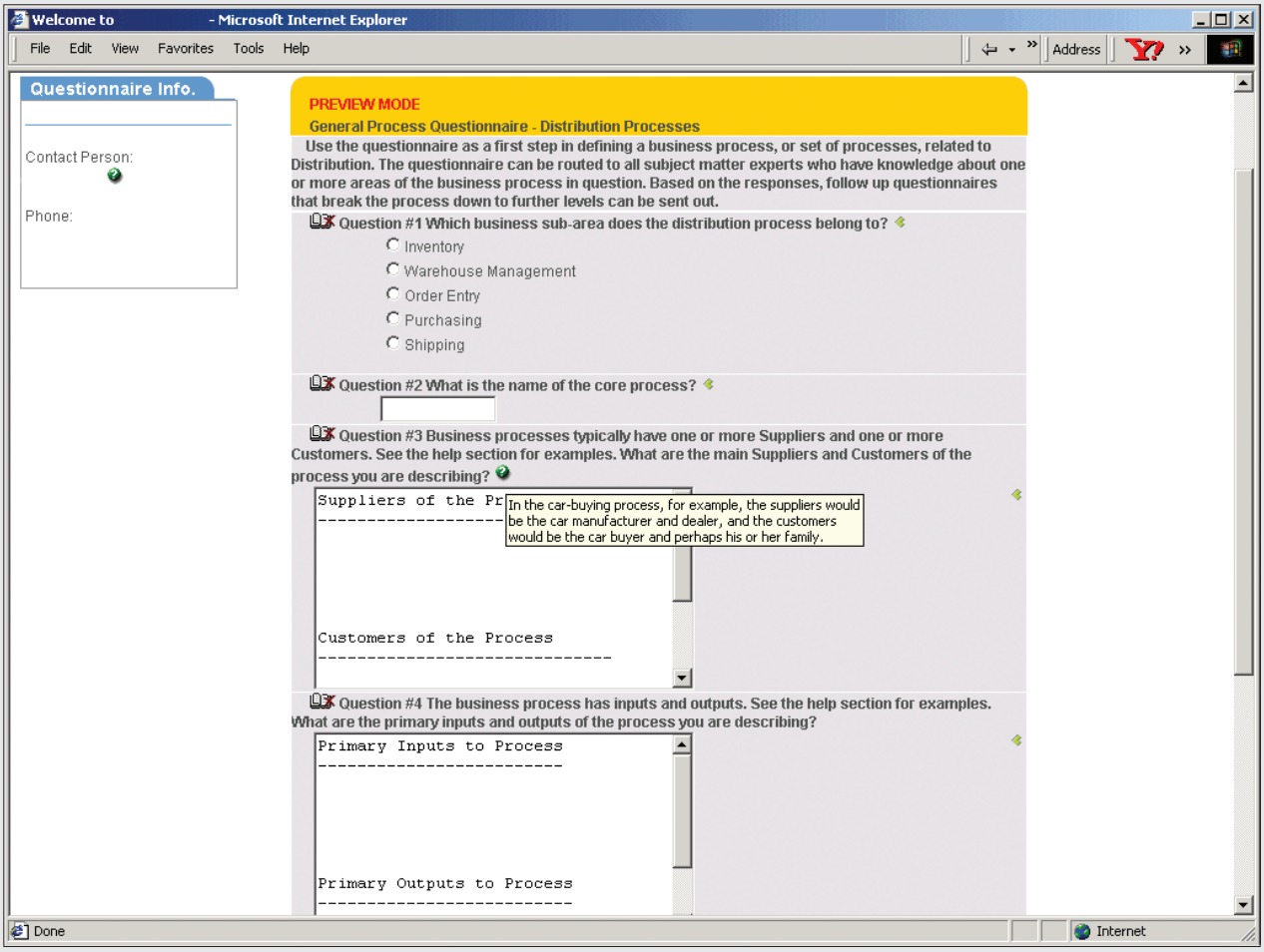

Figure 1 Sample online questionnaire

store the results of your work in a place that is secure, yet accessible and preferably searchable. This precaution will reduce or eliminate the need to go through the painful process definition experience again when new development projects arise in the future.

Worst Practice #3: Not Following a Best Practices Process

Methodologies are great things. When the sun is shining and the flowers are just beginning to bloom, we love to stand around and talk about how a structured project process is something we absolutely, positively adore and want to use. But once things get under way, it is often difficult to put our good intentions into practice.

Attempting an SAP development project without a proven process is a risky path to follow, like starting out alone on a long trek through an uncharted, dark wood. But unfortunately, methodologies have two annoying problems that give them a bad reputation:

- They seem to get in your way when you don't need them.
- They are nowhere to be found when you do need them.

Technical managers, and especially systems analysts and programmers, tend to feel that methodologies get in their way. For a development project, the methodology may consist of a change and/or configuration management procedure, or some other process derived from industry best practices. For a full-blown software implementation, the methodology may be as all-encompassing as SAP's ASAP methodology. In either case, following these guidelines can often seem cumbersome and a waste of valuable time. For example, everyone knows how annoying it is to fill out five documents and get 10 signatures just to implement a small code fix.

On the other hand, when you could really use some direction, such as when you start a complex development effort, you often struggle to find a methodology that can really help. Where is the ready-made process that gives you advice on when and why you need to create a project proposal, or that tells you what the proposal needs to look like, and what steps should follow it?

In my experience, the more well-defined processes you have in place, the more successful your projects and daily work products will be. Following a routine plan of action, specifically one that has worked well for your organization in the past, increases people and project efficiency. Thus, the real challenge is to find a way to prevent these processes and procedures from becoming cumbersome and value-detracting, while at the same time integrating them into everyone's normal work environment.

A variety of software tools exist today to make this difficult task easier. For organizations working without the benefit of such tools, I suggest that you develop a few standard procedures that can be easily documented and propagated throughout the organization. These procedures should be flexible, so that project teams can use some components and ignore others, based on the complexity of their initiatives.

For Medium to Large Projects

For example, a basic methodology that would be suitable for any sizable SAP development project might consist of the following phases:

- Initial Proposal
- Feasibility
- Implementation
- Test
- Rollout/Training

Whatever project phases you choose to adopt, make sure that they explicitly require the team to take steps to avoid the major pitfalls, such as proceeding without a good proposal, failing to secure stakeholder and user buy-in, and shortcutting the test and rollout phases.

During the Initial Proposal phase, for example, you might require the project team to prepare a project proposal that outlines the business objectives and scope of the project. It also makes sense to have the project stakeholders sign off on the proposal before the team continues on to the Feasibility phase. Sometimes this approval stage is referred to as a project "gate." Make these steps as easy as possible by providing project teams with ready access to project proposal templates. Store completed documents in a database that is well organized and searchable, so people can learn from past projects. Where possible, simplify the signoff process by creating approval forms that can be distributed and stored electronically.

For Small Projects

For SAP development initiatives that are shorter in time frame and smaller in scope, such as change requests, you can (and should!) use a more streamlined methodology. In fact, since these efforts are really "processes" rather than "projects," they are better suited to a process-oriented approach. For example, a basic process for handling IT change requests might include the following steps:

1. Submit a change request form for approval.
2. Provide a functional specification or instructions.
3. Create technical specifications.

4. Implement the change in the development system.
5. Perform the alpha tests.
6. Implement the change in the test system.
7. Perform the beta tests.
8. Document the change in the code change tracking system.
9. Get signoff on all work.
10. Move the change into production.

Remember, the purpose of having methodologies and processes is not to burden your staff with more bureaucracy, but to increase employee productivity and ensure that important steps are always completed. You can pay a little now by adopting whatever methodology best suits your project needs, or pay a lot more later in cleaning up the mess.

Worst Practice #4: Paying for Consultants, Not Your Own People, to Get Smarter

Augmenting your in-house development staff with experienced consultants can be a good way to help your SAP project team achieve its critical project goals. And working side by side with more experienced individuals should, in theory, help to narrow the knowledge gap between the consultants and the staffers. But it often strikes me that the way many companies do business actually widens this gap.

Practices that make consultants smarter at the expense of the home team include:

- Giving consultants more exposure to the new systems and technologies
- Teaching consultants, but not your own people, how to leverage methodologies
- Making it difficult for consultants to share the knowledge they already have, and the knowledge they acquire during the project

First, consultants are often assigned the tasks that will increase their experience level with new, critical technologies. In the meantime, the home team is stuck minding the store (such as maintaining legacy applications or minding the EDI system that is scheduled to be replaced). For projects where time is an important factor, the thought is that the consultants have done this work before, so they will be able to deliver the desired result faster and better than the home team. This approach is especially prevalent when facing the complexities of a large-scale SAP development initiative.

Instead, I recommend giving the in-house staff as much hands-on time with the new technology as possible, using the outside consulting team for mentoring and technical guidance. This means that, where possible, you should seek out consultants who can be good mentors, not just another pair of hands.

Second, consultants generally work for companies that live or die based on their success at implementing projects on time, on budget, and on track. Consequently, consulting organizations typically supply their consultants with the relevant project management tools and training. By giving consultants the privilege of assisting with your SAP development effort, you are essentially training them in how to leverage project methodologies. Good consultants will take whatever information they learn during your project, such as effective project phases and useful document templates, and add them to the project management tools that their employer has already supplied.

On the other hand, your own development staff may not have had as much exposure to project management techniques. Often, unlike their consultant counterparts, they will also not have access to easy-to-use project management tools. So while the consultants learn from the project and use it to enhance their toolkit, your in-house staff does not benefit from the same learning experience. While they may be exposed to the project steps, procedures, and templates, they lack the tools and background that would enable them to assimilate this knowledge for the benefit of their next project.

Third, because the in-house staff lacks this context, it becomes difficult for the consultants to share knowledge with them. I have participated in many projects in which a "knowledge transfer" period was supposed to take place for one or two days before a key consultant rolled off the project. Like teleportation, which is an equally mysterious concept that consultants often dream about, I have never seen "knowledge transfer" actually work in the real world. You simply cannot transfer knowledge to someone about a complex development effort in just a few hours.

Because of their project management skills, good consultants gather a wealth of valuable information during a project that they will probably never have the time or inclination to share. They will copy all their functional and technical specifications, their configuration scripts, and their development objects and program code into their employer's best-practices database, or at least onto their hard drives. They then take this treasure trove with them when they walk out the door. If your company does not have its own project knowledge base, or if your knowledge base is not easy to access and use, chances are quite high that your own people will never benefit from this learning.

I recommend putting procedures in place that will help you avoid the three practices just discussed. Start by giving your own people as much hands-on experience with the new technology as possible. Then recognize the value of project management skills and tools, and invest in providing the relevant training for your staff. Finally, provide consultants with tools and incentives that will help them share their knowledge more effectively with your organization. If you try this approach, you may find that consultants are not the only ones who get smarter from your SAP development efforts.

Worst Practice #5: Making It Difficult to Retain Project Knowledge

Worst practice #5 ties directly back to the concepts discussed in #3 (not following a best practices process) and #4 (paying for consultants, not your own people, to get smarter). In my experience, successful organizations make the effort to ensure their people have as much information as possible about effective processes and past projects. An IT team that understands the value of exercising good project management skills, and that has access to documentation and lessons learned, will be better equipped to handle complex SAP development efforts.

Unfortunately, organizations often do not place much emphasis on the need to retain project knowledge. Instead, they focus on successfully navigating through one project at a time, and reaping the immediate rewards of its completion. It is almost as if they view each new development effort as its own animal, with no relation to anything that preceded it. As a result, the IT organization becomes very reactive, instead of continually building the skills that could make it more resilient and proactive.

>> Note

Tips for Maximizing Investment in Your Own People

- Hire consultants who can be good mentors to internal staff, not just another pair of hands.
- Provide internal staff with plenty of hands-on experience with the new technology.
- Train internal staff in the use of project management skills and tools.
- Implement incentives for consultants to share their knowledge throughout the project.

Retaining the knowledge that evolves from your projects is admittedly a challenging task, especially for projects with the complexity and broad impact of SAP development initiatives. This is a challenge that I've spent a lot of time thinking about, and I actively follow industry advancements in the area of "continuous process improvement" (sometimes referred to simply as "CPI"). But an organization simply cannot improve without mechanisms that allow it to retain project knowledge in the first place, thereby building a kind of global memory. Without that memory, every new project will, to some degree, reinvent the wheel.

I have found that you can commonly find project knowledge in:

- The "Z" drive on a shared network directory
- Paper documents in a filing cabinet
- Lotus Notes or a similar database (Update: These days it is typically Sharepoint or another collaboration tool)
- Email correspondence
- Documents on the hard drives of individual team members (Update: Or these days it may be in their "personal cloud")
- Information in people's heads

As you can see, there is a wide range of knowledge types. It is, of course, impossible to rank them in importance, as that can vary for every project. But it is possible to make it easier to gather and later retrieve key sources of information.

Why Knowledge Management Is Harder Than You Think

There is nothing inherently wrong with storing knowledge in any of these places. But the goal is to devise a way to ensure that as much of this knowledge can be shared as easily and effectively, and with as many people, as possible. Obviously, knowledge in one individual's head is more valuable to the organization if it can be distributed across a broader audience. My experience has been that companies often do not place enough emphasis on mining and reusing all of this learning and know-how.

I would like to point out a few weaknesses of common knowledge repositories. For example, storing project documents on a shared network drive often leads to a false sense of security. As the project begins, someone creates folders with meaningful names such as "Presentations," "Functional Documents," "ABAP Requirements," "Change Forms," "Project Plan," or "Status Reports." While team members often make real attempts to store their documents in this repository, there is often no compelling reason for them to do so.

One solution is to assign a supervisor or project librarian to be responsible for checking after each team member to ensure that the latest versions of their documentation have been stored in the appropriate folders. Often, there is simply no time and/or budget for this type of monitoring. As a result, critical documents are never stored. Also, many organizations do not provide the ability to do full-text searches through the documents in a central project folder. So even if the document you need is really there, you may not be able to find it. (Update: although current collaboration tools address many of these challenges, they still cannot ensure that team members record their knowledge and that it is organized in a meaningful way.)

Storing paper documents in a filing cabinet has all of the weaknesses of storing them electronically on a central file server. However, this method has an added detraction: paper documents tend to become lost more easily, since people don't download copies of them, but take the originals with them to their desks.

Of all the knowledge capture methods presented here, a database like Lotus Notes/Domino or a document management system is probably the best. (Update: these days the nod would go to a collaboration platform such as Sharepoint.) Yet even these systems share some of the problems inherent with a central file server. The weaknesses of email correspondence, data on hard drives, and information in people's heads are readily apparent.

Making Knowledge Management Easier

Making it necessary, but easy, for team members to store knowledge is a practice you want to encourage, either through homegrown processes and tools, or via third-party offerings. Ideally, you want to link a project execution environment with an integrated document management system. Let me offer you an example that demonstrates the benefit of making information available to project members at the right time, and in the proper context. Suppose members of your team are assigned a task that has a deliverable (such as a configuration script). Downloading the configuration script template and then storing the completed script in a project library are routine operations that occur naturally in the context of using the application. The application is also integrated with collaboration tools, so that valuable information is collected in online discussions or surveys, instead of in email messages.

Knowledge management is a relatively new science, and even the most sophisticated and expensive tools on the market do not offer an optimal solution. (Update: Knowledge management tools have come a long way, but you still need a plan to help you implement and leverage them effectively.) For project managers, I suggest that you examine how you gather knowledge, and see if existing tools or incentive mechanisms can help you do it more effectively. **Figure 2** on the next page summarizes the fundamental components of a knowledge management strategy, with information sources and supporting activities for you to consider. While ongoing, this effort will deliver very real rewards to your organization.

Worst Practice #6: Underestimating the Importance of Communicating with End Users

I know of at least a few high-budget development initiatives that ultimately failed because the majority of users who were expected to work with the new system rejected it. The development team had completed its work on time and within budget, but the end-user community felt broadsided when they were finally exposed to the new application.

This seemingly improbable occurrence is, in fact, not at all rare, as a result of:

- The departmental structure of many corporations
- Lack of time on both the part of the IT team and the end users
- Inability of IT people and end users to talk to each other effectively

Second, the demands of an intensive SAP development initiative keep you very busy. As IT professionals, we are typically involved in not just one project, but in several overlapping projects that consume more time than we have in an eight-hour (or even 10- or 11-hour) workday. And the end users who will be affected by the changes we are making are similarly consumed. As part of a working business operation, they are focused on taking orders, responding to customer inquiries, stocking the warehouse, paying vendors, and carrying out the fundamental processes that keep the business running.

We programmers and system analysts think that the end user is giving us the brushoff. But, in fact, we often talk in a language that has no meaning for the end user. We speak in the language of bits and bytes, when what they want to know is how the new application will change the way they have to enter an order, or whether we can get rid of that irritating screen that pops up every time they click "Cancel."

Bridging the Communication Gap

Becoming aware of communication issues is the first step, but then there are many small steps that you can take right

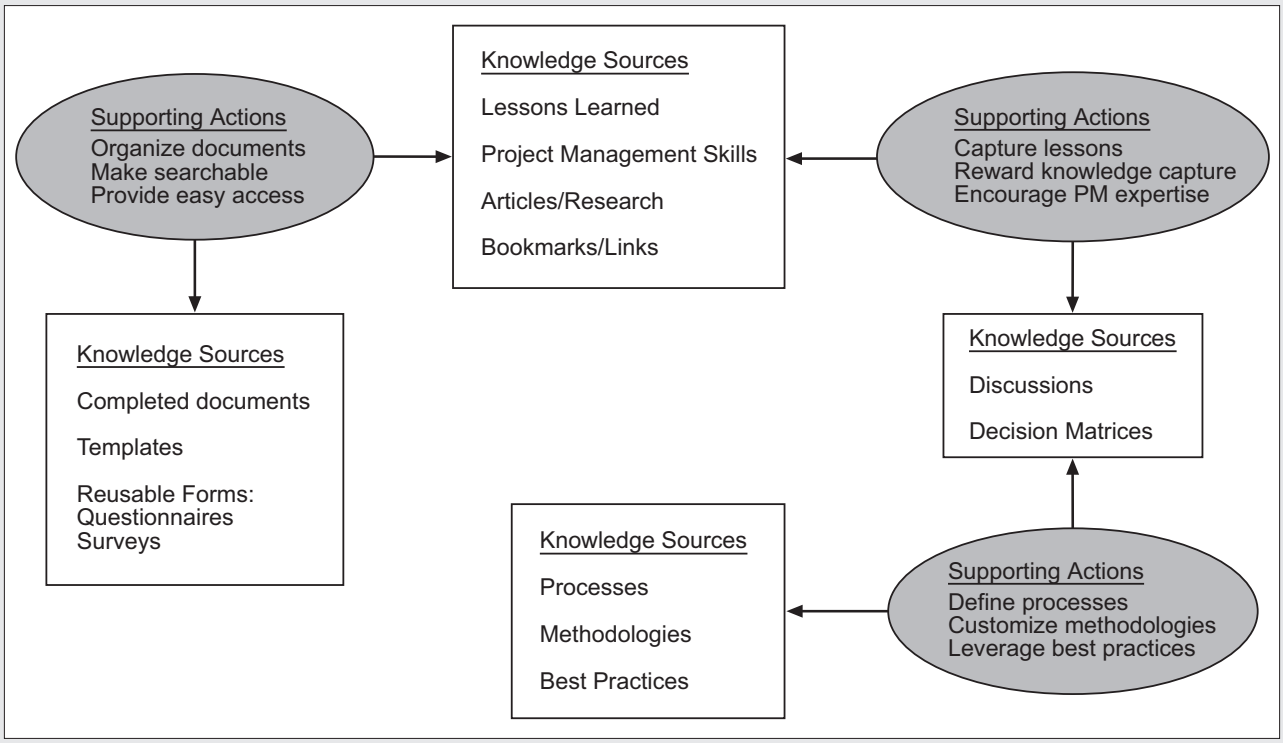

Figure 2 Opportunities for knowledge management

away to narrow the gap, and ensure that IT and the end users communicate more effectively throughout the development effort. For example, you might consider the following options:

- Bring the project teams and end users together in the same workspace, as mentioned earlier.
- Offer "working lunch" sessions where end users can see prototypes of new systems or features during their lunch hour.
- Train IT experts to communicate more effectively with end users.

Internet technologies, including intranet applications and corporate portals, also provide an excellent mechanism for fostering project information exchange. I'm sure you'll agree that involving end users in the design, testing, and final acceptance of your new SAP application is essential. One way of overcoming the challenges presented by departmental barriers, time limitations, and personal communication issues is to establish a project information page or portal on your intranet. This "project portal" is an effective method of sharing various types of information simultaneously with a wide range of people:

- The main objective would be to publish ongoing information about your project, including its objectives, the estimated project time frame, and the likely impact on an end user's job.
- You could also publish images of new screens, and report layouts or other features that the users will be confronted with.
- If your organization has access to the necessary technology, the site would be an excellent place to post surveys to get feedback from end users on design work that has been completed so far.

A project portal would be easy for everyone to use, require a minimum amount of development and maintenance time, and could be designed in such a way that the most important information is clearly communicated to end users in a language that makes sense to them. And finally, a properly designed project portal can be a springboard to end-user training, providing users with the understanding and skills they need to feel comfortable with the new application as soon as it is deployed.

Worst Practice #7: Failing to Secure and Maintain Executive Buy-In

Worst practice #7 corresponds in many ways to worst practice #6 (underestimating the importance of communicating with end users), since it is rooted in a lack of communication, but it can be even more detrimental. The ultimate success of a project might be based on whether the end user is happy with the final product. However, if you fail to maintain executive support, you may never even have the opportunity to complete your project. Without executive buy-in, the project has a good chance of being canceled or deferred, regardless of how important it may be for the organization.

I have seen many IT projects fail due to a lack of support from the executive team. Because this scenario skirts the realm of politics, sometimes you can do nothing to overcome a lack of executive interest or commitment. However, I believe that you can often convince executives to support a project if you take the appropriate steps to address their needs from the start.

Highly compensated executives are not paid to be easily impressed by technology, or to fund projects with vague rewards, such as "users will only have to log on to one system every morning, instead of three or four." Perhaps the corporate portal that you want to implement will bring greater benefits than simply reducing the number of systems that employees need to interact with, but to gain the executive support you need, you must articulate those benefits clearly, in a way that is compelling from a business perspective.

So, once again, we find ourselves revisiting worst practice #1 (proceeding without clear project objectives). Running headlong into a project without stopping to examine the objectives and benefits is not only a poor practice, it can also lead directly to the demise of your project. You will not win executive buy-in for your project if the business benefits are not real, objectively measurable, and clearly articulated.

To expand on the solution presented under worst practice #1, start by developing a project proposal that spells out the benefits in business terms. Ask the business teams to help you identify some key indicators that you can jointly monitor throughout the project to measure its success in concrete numbers (for example, increased revenues, decreased cost of inventory, or reduced procurement costs). Present these metrics to the executive team. They are much more likely to adopt a project that will obviously increase bottom-line returns or shareholder value than one that simply glorifies technology.

Just as a project portal is a great way to communicate with end users, an executive project portal would be ideal for keeping the executive team up to date on your progress. For example, you might use it to inform them at various points during the project:

- During the initial planning and implementation stages, provide some

of the same information as the end-user project portal. Consider also emphasizing return on investment (ROI) factors.

- As the new system or application is deployed, add near-real-time indicators of business metrics that are linked to the project's success factors.

This shared progress update gives everyone the opportunity to see and measure the ROI. Once one of your projects has delivered this type of tangible return for the business, I'm sure you will find it much easier to get executive buy-in for the next one.

Worst Practice #8: Poorly Documenting and Tracking Code Modifications

Now it's time to shift gears from people issues to technology challenges. Packaged software has an annoying characteristic of not always doing exactly what your business needs it to do. As a result, you sometimes have no choice but to tweak the standard software. There are many schools of thought on if, when, and how SAP code should be modified, and I know it is a touchy subject with many corporations. The bottom line is that I have seen a significant number of follow-on projects fail as a direct result of code modifications that were made during prior development initiatives.

Some of these follow-on projects failed because so many code modifications were made in earlier phases that the transactions being used for normal business processes bore only a passing resemblance to the standard SAP transactions. Obviously, it is very difficult to upgrade to a new release of SAP software, or to take advantage of major new functionality, when your corporation is using a homegrown system that just happens to be written in ABAP/4. But fortunately this type of drastic modification is rare, so I don't need to focus on this nightmare.

Rather, I want to talk more about the little changes that companies make all the time, such as the inclusion of new routines within existing customer exits, or even the application of SAP Notes. SAP has worked diligently to improve its system for tracking modifications. I urge all developers to educate yourselves about the functionality provided via customer exits and the Modification Assistant. But I also urge every development team to adopt a method of documenting and tracking all code changes and modifications. While I have seen a few companies that have efficient tracking systems in place, the majority of projects that I have been exposed to have no viable system for long-term modification tracking.

Negative results of poorly recorded code modifications include:

- Increased time and expense for every upgrade initiative
- Longer, more difficult follow-on development projects
- Potential of production-stopping errors that are difficult to trace

These consequences are not theoretical or hypothetical — they happen every day. I was involved in one project in which, one day before go-live, a test team spent nearly eight hours trying to figure out a problem. After being upgraded, the test system allowed split-package orders, which were not allowed in the current production system. The problem was finally traced to a user exit that had been created more than a year earlier, but had never been documented. As a result, the user exit code had not made it into the upgraded test system.

I strongly recommend establishing an electronic database for recording essential information about every code change. **Table 2** lists the minimum information requirements that your database should support. This suggestion relates to the issues raised in worst practice #5 (making it difficult to retain project knowledge). Since code modifications are often implemented by outside consultants, you must require them to share their knowledge of any system changes with your organization. Providing them with an electronic database, which requires them to enter specified information before their work will be signed off, is an approach that I have seen work quite well.

Worst Practice #9: Inability to See and Understand the Entire Project Portfolio

Most organizations have a host of SAP-related and other IT projects under way at any given point in time. Many of these projects affect the IT team directly, others indirectly, and some not at all. But the ability to effectively manage the enterprise-wide project portfolio can bring tremendous advantages. At a minimum, being able to understand how projects interrelate can save time and money, while averting potential risks.

I am continually amazed at how often IT teams are asked to begin projects that, due to other initiatives under way within the company, obviously have little chance of being brought to any purposeful conclusion. But I am also troubled by the opportunities that organizations miss due to ineffective project portfolio management. In my view, the negative impacts of poorly managed project portfolios include:

- Money spent on projects that have very short-lived results

Field	Description
Keywords that describe the modification	You will need to be able to search for the modification later using these keywords or phrases.
Business reason for the modification	Documenting the business reason serves two purposes: 1. It discourages programmers from modifying code unless there is a business reason that has received signoff. 2. It helps you remember the purpose of the modification later, when you need to decide quickly whether you still need it.
Functional description of the modification	Describe how the application will work with the modification, and describe in detail the steps that the end user needs to perform in order to use the new code. Also describe the outcome or output.
Technical description of the modification	Include all information that would be required if a new technical person, who has never seen the application before, is assigned to maintain the modification and/or provide technical support for it. Include the code and explanations of it. List all development objects involved, and any special instructions involved with transporting or activating the code or objects.
Upgrade history	Indicate whether the modification was retained during the last upgrade, and whether you expect that it will be needed in the future. Keeping track of the upgrade history will make future upgrades smoother and faster.

Table 2 Requirements for a code modification tracking system

- Missed opportunities to combine projects to produce greater rewards
- Conflicts between projects that lead to failure

The IT team, in a way, is like a group of physicians and surgeons who are responsible for the overall health of an irreplaceable human being. Just as the team of doctors uses every technology at its disposal to keep the patient alive and well, the IT team is focused on maintaining the major life-support systems of the business. If a patient's critical organ fails and no donor organ is readily available, the doctors must use every possible resource to keep the patient alive until the optimal solution of a transplanted organ can be accomplished. In a similar fashion, the IT team is sometimes called upon to build temporary interfaces to systems that are scheduled to be replaced, or provide infusions of code to keep the business going until better technologies are available. This makes a lot of sense. But just imagine what it would be like for the patient if the team of doctors and surgeons couldn't communicate effectively. What if the doctors do not know that the surgeon has a donor organ coming in? The surgeon has the organ ready and waiting, but the doctors are still pumping the patient full of drugs and transfusions just to keep him or her alive for another day.

The lesson here is that the IT and executive teams both need to understand the broader picture of projects and initiatives under way across the organization. By managing the project portfolio effectively, you can also minimize the need for bridge projects.

A high level of efficient communication and awareness can also help teams to recognize opportunities for greater rewards by combining project initiatives. At larger corporations, I have actually witnessed two or more separate divisions spend valuable time evaluating the same tools from the same vendors, without even realizing that they were engaged in the same pursuit. If the project charters of these initiatives were posted in a portfolio management system, the project managers could find one another and combine (or at least coordinate) initiatives. Imagine the cost savings and increased efficiency of a single vendor solution that is chosen to fill the requirements of three separate departments, instead of each department individually.

Finally, having access to a well-managed project portfolio helps project managers to avoid potential conflicts with other initiatives that are under way.

One source of conflict is the need for projects to share human resources. Especially now, in a weakened economy, we all have to make efficient use of the people in our organization. While the workload has not gone down, the number of people available for each project may have. For this reason, it is essential for you to keep track of what other initiatives people are involved with in order to avoid conflicts that could negatively impact one or more projects.

Another source of cross-project conflict is the potential impact on systems and applications. It is not rare to see two groups working on the same system, with each team negatively affecting the work of the other. This scenario is especially common with SAP development initiatives, which typically impact multiple departments. Again, you can avoid these types of costly conflicts by managing the complete project portfolio more effectively. Don't forget that project portfolio management is more complex than managing each individual project. While it is possible to perform basic portfolio management using spreadsheets, if at all possible take advantage of software tools on the market that were specifically designed for that purpose.

Worst Practice #10: Not Learning from Your Mistakes

This final worst practice is to some extent a summation of all those that came before it. Perhaps the most disheartening aspect of SAP development projects is the realization that many IT project managers and teams not only make some or all of the preceding mistakes, but they fail to learn anything from them.

Learning from your mistakes (see **Table 3**), especially on an organizational level, is admittedly difficult. At the very least, learning from past projects requires both a time commitment and a willingness to enhance the project expertise of the project manager and key project members. In some cases, the ability to learn as an organization may actually require a reevaluation of the existing corporate culture. As an example, instead of rewarding people for tasks they have accomplished, you may need to reward team members for the quantity and quality of the knowledge they capture and make available to others.

Ready and Armed with Information: The Road to Success

Whether you are the project manager or a member of the extended project team, my hope is that you will now be well informed and equipped to make your next SAP development project as successful as possible.

Let's take a final look at how you can avoid the possible pitfalls associated

Technique	How It Works
1. Capture procedures that work well as "best practices."	When you find something that works, package it so others in the organization can take advantage of it. As you saw in worst practice #3 (not following a best practices process), organizations that have easy access to proven processes can improve both productivity and efficiency. You can build a real competitive advantage by capturing and continuously improving best practices, which includes project methodologies and procedures that have been successful for you in the past.
2. Facilitate the capture of project knowledge.	Providing a means for team members to easily document the knowledge associated with projects, regardless of what form that knowledge is in, is critical to keeping your organization adaptable and efficient. This knowledge may be information about code changes, or data associated with business metrics, or information shared during an online discussion. But if it can be captured, organized, and made available and searchable, it becomes a source of tangible value for the organization.
3. Provide a project information portal to facilitate communication.	You have seen that it is essential to communicate effectively with both end users and the executive management team. By providing all employees with a project information portal, IT and other project teams can publish essential project information easily and quickly across the organization. By going a step further and offering personalized dashboards, individuals can have access to the precise information that they need most. The project portal can also be a way to gather input and feedback from end users and managers, and to prepare everyone for the new solution deployment.
4. Maintain and use standard templates for project documents.	Almost every organization has some form of boilerplate documents that have been handed down from one team to another. In some organizations, these templates are seldom used, often because new team members do not know they exist. In other organizations, the templates are used, but they no longer reflect the needs of the team or take into account lessons learned. Project managers and teams can work better and faster if they have access to well documented, carefully thought-out templates. These templates should be readily accessible to project teams, and they should be maintained and updated on a regular basis.

Table 3 Four approaches to learning from your mistakes

Don't let this worst practice derail your project...	...Take these steps to avoid it!
1. Proceeding without clear project objectives.	Insist on a formal project proposal and get signoff on it before proceeding.
2. Believing in the existence of "as-is" process definitions.	Be proactive in helping the business teams to define and document the existing processes.
3. Not following a best practices process.	Define methodologies that make sense for your organization, and encourage their use.
4. Paying for consultants, not your own people, to get smarter.	Insist that project knowledge be retained, and encourage your own people to learn.
5. Making it difficult to retain project knowledge.	Experiment with methods and tools that make it easier for your teams to store and reuse knowledge.
6. Underestimating the importance of communicating with end users.	Involve end users throughout the project. Ideally, create a project portal.
7. Failing to secure and maintain executive buy-in.	Define the business objectives up front. Supply executives with project metrics to measure success.
8. Poorly documenting and tracking code modifications.	Create a tracking system and insist that it be used for all code modifications.
9. Inability to see and understand the entire project portfolio.	Implement a project portfolio management system.
10. Not learning from your mistakes.	Take steps to reinforce the use of best practices and ensure the capture of project knowledge.

Table 4 Strategies for avoiding the 10 worst practices of SAP development projects

with each of the items on the "10 Worst Practices" list. **Table 4** summarizes the strategies discussed along the way.

Navigate around these worst practices and steer your SAP development projects toward the road to success. Becoming an organization that follows good practices instead of poor ones takes hard work, planning, a willingness for real collaboration on the part of the IT and business teams, and an emphasis on reusing knowledge. Software tools can aid in this process, but the real key to success is setting a goal of project excellence, and giving everyone on the team the support and incentives to continuously work toward achieving it. ■

Amy Stapleton has several decades of experience managing software development projects. She began her career at SAP in Walldorf, Germany. After returning to the US, Amy gained hands-on project management experience in ERP implementations at major corporations across many industries. Amy joined the National Aeronautics and Space Administration (NASA) in 2001 where she served as an IT Manager. She assisted in the Space Agency's implementation of an enterprise core financial system using SAP. Amy retired from NASA in 2015 and is now Founder of Hutch.AI and an analyst at Opus Research, working in the area of artificial intelligence. You may reach her via email at amy@hutch.ai.