2

RAPID IMPLEMENTATION ROADMAP: WHAT IS THE IMPLEMENTATION PROCESS?

It is important to appreciate all the things that are involved in implementing package systems in order to better understand what is different and important for rapid implementations. These are extremely complex projects at the detail level but are relatively easy to understand at higher levels. This chapter provides a conceptual framework for these projects by walking you through the activities and tasks that occur in the various stages of a rapid implementation.

The implementation process, for a single project, is divided into several stages. At a higher level, the term phases refers to multiple projects or versions that an application or a business process goes through as it evolves to meet the changing needs of business.

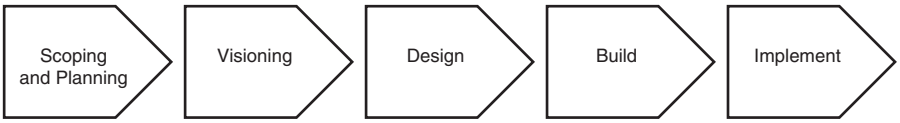
For example, in phase 1, an organization may put in the general ledger module from a vendor's suite of packages and in phase 2 (a separate project) it may implement the accounts payable module from the same vendor. In phase 3, the organization may go back and change the general ledger configuration to handle financial processing for a new subsidiary. Under this scenario, the organization's financial systems have gone through three phases of development through the work done on three separate, but related, projects.

Each of the stages of a rapid implementation project produces major deliverables (e.g., reports, plans) for the project and serves as a milestone to show the completion of a significant piece of work. The stages sometimes serve as a decision point to get buy-in for what has been accomplished (e.g., vision, business case) and a decision to proceed with the next segment of the project.

Generally, the same tasks occur in both traditional and rapid implementations. However, as you will see, the ways they are organized and managed are different.

Figure 2-1 depicts the traditional approach to implementation. It consists of five stages. Each of these stages is a major milestone for the project. The major deliver-

FIGURE 2-1 Traditional Approach to Package Implementation



able that is created at the end of each stage is reviewed by management and a decision is made as to whether the project should proceed to the next stage. Each stage is completed and approved before the team begins work on the next stage. Unfortunately, this approach does not produce rapid results.

Every software package vendor and consulting organization has its own methodology for doing these projects. In the methodologies there is a formal *work breakdown structure.* In the work breakdown structure, the stages are broken down into activities, which are a logical grouping of project tasks that result in one or more major deliverables. The activities are then broken down into detail tasks. In some of the commercial implementation methodologies there are hundreds of activities and thousands of tasks.

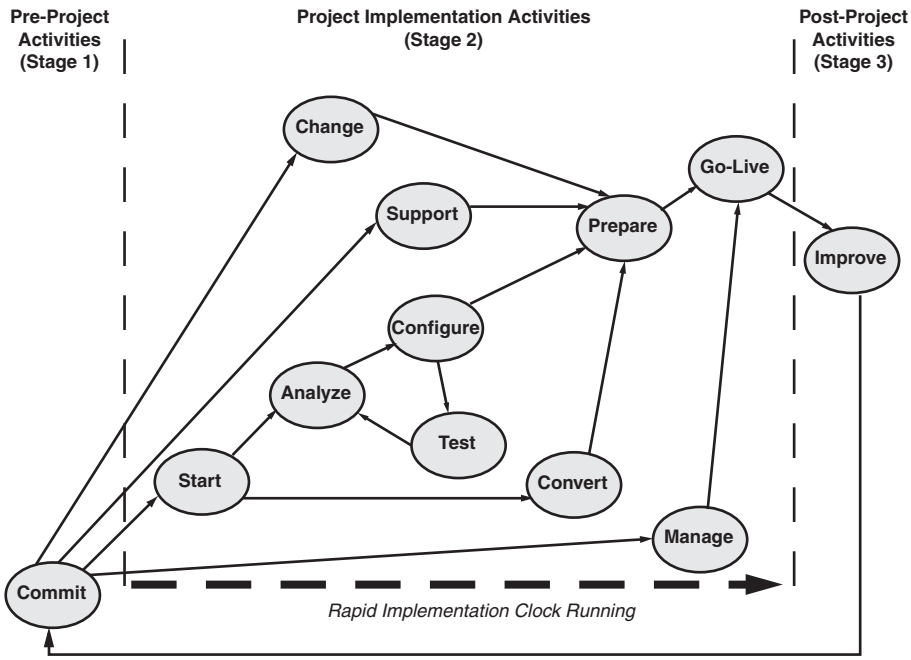
Even though they may do many of the same tasks that are defined by the traditional methodologies, a rapid implementation team needs to approach these tasks differently. In rapid implementation projects, many things have to happen in parallel. These projects are also more iterative in nature and require more trial-and-error than formal design.

In a rapid implementation there is no time to produce formal deliverables that take weeks or months for management to review and approve before the team begins the next stage of work. Things happen too fast. Management is consulted and involved throughout the implementation but many of the decisions on scope and design are delegated to the project team and key users. Also, the entire project is approved ahead of time—not just one stage at a time.

A rapid implementation is executed more like the diagram in Figure 2-2. This roadmap shows the project's 12 major activities, arranged in three stages. This framework will be used to provide an overview on the work done on these projects. It also serves to highlight some of the differences from a traditional implementation approach.

Even at a high level we can immediately see some differences with a rapid approach. First, Figure 2-2 shows a lot of things going on in parallel throughout these projects. It also implies that there is a lot of iteration and looping back and forth as the project progresses. Finally, it highlights that certain activities must occur before the project is formally kicked off and after it is completed. These pre- and post-implementation activities are necessary for a rapid approach to be successful. They are normally not adequately covered in traditional implementation methodologies.

FIGURE 2-2 Rapid Implementation Roadmap



We turn now to what actually occurs in each of the 12 activities on the rapid implementation roadmap.

COMMIT

The Commit activities make up the preproject stage of a rapid implementation. They occur prior to formal project kickoff. Many of the major decisions and actions that ultimately determine whether a project is a success or a failure occur very early in the life of the project. Many of these decisions create expectations for the project, which serve as the standard against which the project will be measured. As a result, handling preproject activities effectively is a key first step toward ensuring that the project will be successful.

One of the primary reasons implementation projects fail is because there is insufficient commitment to the project by top management. However, in order to get this commitment, managers need to know what they are committing to. Therefore, a primary concern of the commit activity is to define the project in terms of its objectives, benefits, and expectations.

There are a number of steps that need to be completed before the implementation project can begin. To start a project without completing these tasks puts the project at risk. Here are some of the things to be accomplished during the commit stage:

- Select the project managers.
- Develop a business case.
- Set up the project team infrastructure.
- Prepare the initial project plans.
- Recruit the project team.

Select the Project Managers

One of the first things that must to be done is to select two project managers. One of the project managers is the lead project manager, and comes from the organization implementing the new system. The other project manager is the person from a consulting organization or the software vendor who will coach the organization's project manager through the implementation process. Once selected, these individuals will do the high-level planning for the project and will be responsible for driving the project to completion.

The lead project manager should be a highly respected individual with broad organizational knowledge and contacts. This person will be ultimately held accountable for the success of the project.

As will be discussed in more detail in the project staffing chapter (Chapter 5), a consulting coach is needed because many key decisions are made early in a rapid implementation that impact the ultimate success of the project. These decisions set the direction and tone for the project and need to be made with the knowledge of someone who has a lot of experience with these types of projects. An experienced consulting manager also helps the organization's project manager avoid mistakes and false starts throughout the project that could be fatal to a rapid implementation.

The coach brings a methodology for doing these projects, experience in managing the risks of these projects, and an independent perspective. The organization's lead project manager cannot be expected to have these tools and traits. However, for important reasons that will be covered later in the book, the organization should *not* have a consultant lead the project. This is a role for someone from the organization.

If the project managers were not involved in the package selection and justification activities, they need to quickly get up to speed on a number of things: What is the project really about? What is the expected scope of the project? What things are specifically excluded from the project scope? What are the expected timeframes? Who are the supporters and who is the opposition? What is the business imperative for the project and how does it fit into the overall strategies of the organization? Much

of this information is available from the project sponsor, those who participated in the software selection, and those who worked on the business case for the project.

Develop a Business Case

The business case is the next area to examine. The project managers must ensure that there is a business case or charter for the project. The business case quantifies the benefits and costs associated with the investment in new systems and business processes.

Ideally, a business case was developed as part of the process of selecting the package and justifying the investment to implement parts of that system. If so, this information can be reviewed and updated at this time. If a formal business case was not prepared, then, at a minimum, the high-level goals and objectives for the project need to be put into writing and approved by the project sponsor and steering committee. This may be documented in the form of a project charter. These goals and objectives can be taken to a lower level of detail as required as the project progresses.

This business case will be useful in managing the scope, boundaries, constraints, risks, expectations, and assumptions for the project. The assumptions include things like whether the organization will modify the vendors' code or follow a strategy of going with the vanilla version of the software and change business processes to the methods supported by the package. The constraints might include the budget for the project and a go-live date that must be met for critical business reasons.

Set Up the Project Team Infrastructure

The third step in Commit is to set up the project team infrastructure—before the project starts. This includes constructing the *war room* for the team with PC and phone access for each team member. The team members should be connected to both a local area network and development and test versions of the package software.

The IT support group for the project has a lot of work to do in setting up this environment in time for project kickoff. Since the organization's personnel may not have been to the technical training on the package yet, they may need the assistance of technical support people from either the vendor or the consulting organization in order to get these tasks done on time. In the early stages of the project the organization's IS personnel are often watching, learning, expediting, and making sure things get done to prepare a suitable technical environment for the team. As the project progresses they will be setting up the production environment for the organization, mainly on their own. After go-live, they will be responsible for supporting the new system in actual use.

Other administrative aspects of the team infrastructure also need to be set up in time for the kickoff meeting. The network (LAN) support for the project team (e.g.,

folder structure, access rights, email, calendaring process) needs to be established. Forms and policies need to be created for issues logging, status tracking and reporting, and expense reimbursement. Many of these items will be covered with the project team members in their team meeting following the formal project kickoff meeting.

Prepare the Initial Project Plans

A number of plans need to be prepared before the project is staffed and started. One of these is a risk management plan. The project managers need to assess all the things that may cause this project to fail and those things that are key to its success. Plans and actions should be identified to mitigate or respond to the risks and to ensure the key success factors will be in place. These responses might include things like adding additional steps and checks to the project workplan or staffing the project with more experienced personnel in certain areas.

All of this should be documented in a risk management plan. If there is not support for actions that are necessary to make the project successful, then the project manager should address this issue with the steering committee. Perhaps the organization is not ready to start this project at this time. After all, there is little use in starting a project that has a high likelihood of failing.

A preliminary workplan should also be developed at this time. It lays out a roadmap for the entire project. It will also be used in determining the proper staffing level for the project. The project managers will not attach names to the various tasks until they have finalized the team organization and evaluated each individual team member's areas of interest, experience, and skills.

The project organizational structure and roles and assignments need to be developed next. The same person may perform several of the roles. If the project is using an application service provider or contract programmers, some of the roles will be outsourced.

For a rapid implementation with a small project team there will be a flat organizational structure. These projects do not need a lot of chiefs; everyone should be doing direct, value-adding work. In addition, all team members will pitch in, wherever required, to get the project complete on time while achieving its business objectives.

Finally, the project managers need to determine the key deliverables that will be created in the project. There will be hundreds of deliverables described in the methodology used by the vendor or the consulting organization. But, in line with the philosophy of doing only value-adding tasks, the project managers need to decide what are the really important, and therefore mandatory, deliverables from a rapid implementation.

These deliverables will guide the activities for each task, control the level of de-

tail that the team members go to, and help the team members know when they are done with a task. The deliverable plan should also address whether the team will require signoff of the deliverables by the end users, the sponsor, or the steering committee.

Recruit the Project Team

The final step in this activity (and stage) identifies those people that will make up the core implementation team. A deliberate effort should be made to recruit the best possible people to be on the team. A lot more will be said on this topic in the chapter on staffing these implementations (Chapter 5). However, it should be noted at this time that the quality of the people who will work on this project is one of the keys to its success.

In a rapid implementation, management is going to have to empower the team to make a lot of the detail decisions that will affect the way future work gets done in the organization. The team members will be required to come up with creative ways to use the package to accomplish what has to be done. They will have to do this under intense time pressures while facing resistance to the changes that will be made from many areas of the organization. This is a lot to ask. Therefore, we need an organization's *A team* for these projects.

If the tasks covered in the Commit activity are not completed before the project kickoff, it will be difficult for the project team to hit the road running. Good planning and preparation *before* project kickoff gets the project off to a good start.

START

The Start activities initiate the project implementation stage of a rapid implementation. This activity officially begins with the project kickoff meeting. This meeting normally takes a full day. In attendance during the morning session should be several top managers of the organization, the steering committee members, key functional managers, the project team, and representatives from the vendor and consulting firms. The afternoon session is just for the project team.

This meeting should be preceded by individual discussions with the project sponsor, the steering committee members, and other key managers. With these meetings, the project managers begin the process of building understanding and commitment for the project throughout the organization. There should be no surprises in the kickoff session.

The organization's project manager should lead the kickoff meeting. However, the most senior executive present should open the meeting with a statement that describes the importance of the project to the organization and the support the team has

from top management. It is also useful for this person to lay out some rules of the road for the relationship of the team to the rest of the organization. Some sample rules include:

- Top management expects the departmental managers to make time to be involved with the project.
- The organization is using a rapid implementation approach because management believes it is the best way to accelerate the benefits of the system and respond to quickly changing business requirements. As such, rapid implementation will become a core competency for successful organizations.
- This approach will require the organization to limit the scope of this project. The organization will go live on the new system with all the processes that are ready on the target date. Some good capabilities of the vendor's package will have to be deferred to later phases.
- This approach requires numerous trade-offs. It will only succeed if it has the support of all areas of the organization. Since the future of the organization depends on learning how to do these implementations rapidly to meet critical business requirements, top management expects that the team will get full support from all departments.
- This is not a one-time effort. The organization will do a lot of these rapid implementation projects in the future. Therefore, we need to evaluate the process and capture lessons learned so they can be applied in future projects.
- These projects will be successful only if there is a great deal of user involvement. Therefore, we expect the departments to do whatever is necessary to make their people available for the tasks that require extensive user involvement.
- Top management will not allow individuals and groups to criticize the project and the team behind their backs. If people have a problem with the way the project is going, they should bring it to the attention of the project manager or the sponsor.
- The organization is implementing this new application software in order to have the ability to change business processes in the future and have them supported by the vendor's products. This software has a lot of flexibility in the way it can be implemented. We will not modify the vendor's code to meet the way we do things currently.
- I have instructed the project manager to come to me with any roadblocks to the successful completion of this project that cannot be resolved by the sponsor or the steering committee. I will make whatever decisions and take whatever actions are necessary to support the successful completion of this initiative.

One of the first indicators of the importance of the project is whether the representatives from top management stay for the entire morning session. Another will be

the actions top management and the steering committee take to keep this project on track when the inevitable bumps in the road appear. With swift response to resolve the first few project problems, word will get out that something different is going on with these rapid implementation projects.

By this time most of the steps from the Commit activities should have been completed. For example, the workspace is ready, and the team members have been transitioned off their old jobs. It is very disruptive to project progress for the team members to join the project at different times. Those joining the project late miss a lot of the team-building activities and initial training and planning that occur in the first weeks. No matter how hard they try, they often stay behind the others for most of the project.

If all the Commit activities were not done by the project kickoff, they should be completed as quickly as possible at the beginning of this activity. However, it is not a good sign for the project if there are a lot of these activities to complete.

The other main step that occurs in the Start activity is providing initial training for the project team members. This training should be scheduled in the four or five days that follow the kickoff meeting. Initial training should cover the methodology and approach that will be used for the project and provide an overview of the design and capabilities of the software package.

Many of the team members may not have been on the software selection project and therefore may never have seen the vendor's system. It is important to get them into the software quickly. Therefore, these first training sessions should minimize the amount of slide show-based lectures and provide, instead, some hands-on training on things like signing on and navigating through the system, as well as having actual demos of the software by the trainers.

A difficult decision is whether the team should be given the standard training from the vendor at the vendor's training site or receive a tailored version at the organization's location. The problem with the generic training is that it will cover a lot of capabilities in each of the package's modules that are not within the scope of the implementation—and may never be used by the organization. Also, if team members train at the vendor's training locations, they will usually be in classes with people from a lot of different organizations. In these situations a great deal of the class time can be spent answering questions that are relevant only to other organizations.

However, there are benefits from the team members seeing all the capabilities of the system in the vendor's generic training. After this exposure, they may be in a better position to identify functions that could be implemented in the future.

Using tailored in-house training has the advantage of focusing the training on the specific needs of the project. This approach should simplify the training by reducing the number of areas that have to be covered. It also is more in line with the overall philosophy of eliminating non-value-adding or low-value-adding activities from rapid implementation projects.

No matter which training approach is chosen (generic or tailored), the project team

usually benefits from receiving the initial training as a group, near the organization's site. This is often easier on the project budget since it minimizes travel expenses for a large number of training attendees.

MANAGE

This activity includes all the things the project managers do to plan and control the project. In a rapid implementation, the managers act a lot like player/coaches. The teams are small and the organizational structure is flat. So, the managers spend a lot of their time doing on-the-job training with team members, offering suggestions, providing examples, guiding the work, reviewing deliverables, and removing barriers in the team's path.

The project workplan is a key tool for the project manager. It lists all the tasks that have to be accomplished before going live with the new system. The project is planned and controlled by means of this document.

The project workplan must be a living document that is used and updated throughout the project. Too often, the plan is prepared at the beginning of the project and then never reviewed, used, or updated. This is unfortunate, because the workplan is the primary tool that helps the project manager determine how things are really going.

The project manager determines project status by reviewing and accepting deliverables from the team members. In this process, there are only two task states that are important: the task is either (1) complete or (2) still in process. The practice of asking people for estimates of the percent of each task that has been completed has never been an effective way for determining project status.

The tasks should be defined at a level where each one should take no more than two days to complete. At the appropriate time, the project manager sits down with each team member and reviews the deliverables from the task. They are either *done* or *in process*—nothing in between. If there is no completed deliverable, the task is not done. Although this may seem simplistic, it is difficult to fall weeks behind schedule and not know about it with this approach.

One of the requirements for rapid implementation is templates or sample deliverables for most of the tasks. They are key accelerators for project activities. It is much easier for team members to tailor documents than create them from scratch. These sample deliverables help define the level of work for each task and provide a consistent format for the documentation. A key project manager task is to select the templates for the project and teach the team members how to use them.

The project manager is responsible for the performance of the team members. If team members cannot carry their assigned part of the load, or have problems that cause them to disrupt project activities, they must be replaced. The project manager must address these problems quickly and decisively. With a flat organizational structure and aggressive schedule, the project managers cannot spend a lot of time coun-

seling and correcting the work of any specific team member. Project manager time is needed in all areas.

Risk management is also a key task for project managers. They must anticipate things that could go wrong in the project and do whatever is necessary to prevent them from occurring. For those risks that are outside the control of the project team, contingency plans should be developed to allow the team to respond quickly to minimize impact on the project. The team must be able to make quick course corrections during the project.

The project managers are responsible for developing the team members. These projects are great opportunities for knowledge transfer and provide team members with a broad understanding of how the organization works. Team members become valuable assets to the organization once the projects are completed. The project managers should look for opportunities to take advantage of the knowledge and experience the team members bring to the project and give them opportunities to develop new skills and knowledge.

The project managers are responsible for keeping the team focused on the goals and objectives for the implementation. Time is valuable in these projects. Every task needs to be challenged to determine the value it provides toward meeting the project goals and objectives. Non-value-adding tasks must be eliminated.

A big part of managing these projects is managing the scope of the implementation. To a certain extent, the detailed scope is not known at the beginning of these projects. The team has project goals and objectives but does not know all the detailed transactions that will be included in this current phase of the package implementation. Therefore, there is a continuous process that defines the scope in more detail. Deciding and documenting all the things that are not included in scope is equally important to the success of these projects.

Finally, issues management will be an ongoing task for the project managers. The issues (e.g., problems, opportunities, decisions) need to be identified by all the team members and then prioritized and resolved. In a rapid implementation, the team cannot take weeks or months to resolve issues. They must be taken care of in a matter of days. Therefore, a useful project metric is how many unresolved issues are in the issues log at any one time and how long is it taking to resolve them.

Chapter 4 covers project management approaches, activities, and responsibilities in more detail.

ANALYZE

In the Commit and Start activities the organization establishes a vision and business imperative for implementing the new system. Also, goals and objectives for the project are documented in the business case or project charter. In this activity

the implementation team will determine how to achieve that vision and meet those objectives.

What does the new system have to do to support the desired business process changes? How will the organization use the standard software to achieve these changes? To answer these questions, the team will have to determine how the new subprocesses will work using the capabilities of the package.

This activity is the analysis and design portion of the project. It is where the team puts on its business hat and gets creative about how to improve business processes and solve business problems. It means looking at the as-is processes and determining how they will change going forward. These activities require knowledge of the business and its processes, as well as the capabilities of the software package being implemented. This portion of the project also requires creativity and a focus on finding the simplest, best ways to do things within the capabilities of the vendor's software.

The team analyzes how work is being done now and the problems that people experience with the current process design and technology support. They look for new opportunities and ways to do things that are possible because of the rich capabilities of the package that has been selected. They determine what has to be done within the new system to support the business in the areas that are included in the project's scope. In these activities the team determines the requirements for the new system and begins designing ways to meet these requirements.

Designing new processes addresses how the organization will get information into the system, what processing and decisions will occur (either manually or within the system), and what will be the outputs. These outputs may be in the form of reports, business forms (e.g., checks, invoices), updated databases, and inquiry screens that provide information and help evaluate the results of processing. All these requirements for the new system will be documented in a design specification.

In the past, design specifications were turned over to programmers to code using programming languages, such as COBOL, C++, Visual Basic, or PowerBuilder. Once the programs were written, there was extensive testing to ensure the programs worked as designed. This testing occurred on two levels. First, the programmers tested whether the system worked as they interpreted the requirements in the specifications. Second, the analysts and users tested to determine whether the system worked as they had intended when writing the specifications.

This approach for computer system development has been going on for almost 50 years. It still is used in the development of custom, standalone systems, bolt-on functionality to augment a package's functions, and interface and data conversion programs. However, development gets done differently when using a standard software package and a rapid implementation approach.

The main difference is in the nature of the work that occurs after the team determines the requirements for the new system. Rather than designing a new system in the traditional sense, the team members select from a menu of transactions and pack-

age functions and determine how these functions can be configured to fit the needs of the organization.

This difference makes sense when you consider that standard software has already been designed and programmed. The vendor, with the assistance of its customers, has gone through years of requirements definition, data modeling, program design, coding, and various levels of testing. This cycle occurs many times as the vendors provide new releases of their packages.

The vendor's developers have done extensive testing of their own programs and how they integrate with other programs in the application suite. Then, after they believe that the product is relatively stable (and they have completed *alpha testing*), companies are recruited to act as *beta test sites*—to become the first actual users of the software. These beta customers help find additional bugs in the software that can be fixed before the package is made generally available to the public.

In addition to providing tested software, the vendors also have developed system documentation, training programs, implementation templates and accelerators, process diagrams, and data conversion utilities. In some cases, they have developed interfaces to popular packaged software modules offered by other software vendors. These are all resources that the project team does not have to develop from scratch.

Therefore, in a package implementation project, the team starts from a different place than it would for a project to custom-develop a new system from scratch. If the organization has chosen a good package and it supports a preconfigured version for their industry, the project team may have a complete, fully functioning system to work with in the first weeks of the project. However, that still leaves a lot of work to be done.

The standard software usually has several different methods for supporting most processes. So analysis activities focus on selecting which parts of the system will be implemented in this project and determining how to enable these parts of the package and tailor them to meet the needs of the organization.

Lists of all the transactions supported by the system can be annotated to show those in the scope of the project. Process diagrams showing the processes supported by the package can be used for the same purpose. Either method generally shows that most organizations use only a small subset of the total functionality of the top packages. The challenge is to determine what parts of the system must be used to meet the business requirements. Other parts of the system can be implemented in future phases as part of the Improve activities.

For the tasks discussed in this activity it should be obvious that the team needs to understand both the as-is and to-be business processes and the capabilities of the package. The organization's personnel on the team usually understand the organization's current processes very well. However, they usually rely on the knowledge of vendor personnel or consultants on the team who have experience with the package to provide the detailed package knowledge.

As the project progresses, the core team members learn a great deal about the system through additional training, discussions with vendor and consulting personnel, and hands-on experience with the software. There will come a point when all the team members know the package well. After that knowledge transfer has occurred it will be possible for the organization to do these tasks in future implementations of this package with much less consulting support.

The team uses two primary guidelines in making scope decisions during the Analyze activity. First, they need to include enough functions and processes in the scope of the project to address the major problems to be solved and benefits to be achieved. Second, the organization must still be able to carry on its normal work (e.g., manufacture and ship goods; register, teach, and administer student records; bill patients) after it goes live with the new system. These implementations should be transparent to customers—or should delight them with improved service.

Since the team is not implementing all the functions of the package in the first rapid implementation project, they must determine how the other processes in the businesses will continue to get done. This usually means interfacing the packaged software with legacy systems and employing short-term workarounds so the work can still get done while later implementation phases are completed.

The deliverables from of all this analysis work will include documentation, at a high level, that shows how the processes will work in the future to meet the project's business requirements. It will also include a delineation of the parts of the package that will be used to support these processes.

The Analyze activities are shown on the roadmap in Figure 2-2 as part of an iterative loop of analysis, configuration, and testing. The team will not get the analysis and design right the first time they try. They may decide that they can use the system in a certain way only to find out that that approach will not work. During testing they may learn that the package works in a way that is different from what was thought or could be determined from its documentation. On a more positive note, as they learn more about the package and its capabilities, the team may find an easier way to support a process.

During testing and review, key users may surface problems with the design that require the team to find alternative ways to perform processes. The team may also find that some parts of the process are not supported in the current version of the software and, therefore, must be handled with legacy systems, bolt-on functionality, or workarounds. As a worst case, the team may even find that some of the objectives of the project are not supported by the package and cannot be achieved in the initial implementation.

These issues are all part of the challenges and messiness inherent in implementing any new, complex application system. However, after several loops through Analyze, Configure, and Test the team will usually come up with a way to support the business processes with the package capabilities. Then they must configure the system to make it work as designed.

CONFIGURE

Configuration requires detailed knowledge of a vendor's package. Each of these packages has hundreds and thousands of parameter tables that are used to change the way the software works in order to address different business methods and procedures. Changing these parameters is how the team tailors the software to meet the unique needs of the organization. Certain parts of the package need to be turned on and other parts disabled. There are often several different ways the package can be used to support a particular business process. The team must select and enable the alternatives that best meet the organization's needs.

Configuration also maps the organizational structure (e.g., companies, departments, cost centers) to the software database structure. This mapping can have significant implications for reporting and processing of intra- and interorganizational transactions. The system can often be set up to represent one organization with all the related entities as components or it can be set up with each entity as a separate, related organization. These decisions determine how transactions between related entities are handled by the package.

In addition to choosing parameter options and organizational structures, the team defines a financial chart of account and other system codes. These numbers and codes tell the system how to account for and validate the various transactions that occur during processing.

A major risk in configuration is not identifying the "load-bearing walls." These are choices that the team must get right in the initial configuration because they are difficult, or even impossible, to change at a later time. There are many other configuration parameters than can be changed on the fly, at any time, to alter the way the system works. The key is to know how easy it is to switch specific configuration decisions in response to project or business changes.

The nature and number of configuration parameters, and the ease of changing them, varies among the vendors' packages. Some packages have a reputation (e.g., SAP R/3) for great functionality (supporting all the ways an organization could want to perform its processes) but a requirement that the team get configuration decisions right the first time. Other packages seem to be more forgiving of the need to change decisions in the future. In general, the things that are difficult to change are those that affect the way information is organized in the package's databases.

Packaged software needs to be able to be changed continuously to support business strategy and process changes. Making significant changes in some packages can be a difficult and cumbersome process. Other systems are more flexible and can handle certain changes more easily. However, flexibility is a relative term.

There are certain things that are easy or difficult to change in every package. However, the ease of making other changes varies by software product. Some vendors (e.g., J. D. Edwards) market that their product is so easy to change, in response to

changing business requirements, that "you can change the tires while going down the road at 60 miles per hour." The truth in all these claims, pro and con, needs to be tested with real-world examples.

Configure is part of the Analyze-Configure-Test loop. The team will find situations where the package cannot be configured to do what has been designed. Also, errors will be made in configuring the package that will be caught and corrected in testing. Additional configuration tasks occur throughout the project as transactions and processes are added and deleted from the project scope. Each of these situations requires the team to cycle through this loop many times.

TEST

The testing that occurs in this activity is primarily at the transaction level. The team is testing whether the individual transactions can be successfully completed with the way the system has been configured. Examples of things to test at this point might be the ability to enter a new customer, take an order, ship goods, receive materials, prepare an invoice, enter a purchase requisition, or print payroll checks.

The tests designed and conducted at this stage require an in-depth knowledge of the business requirements and how the package works. The core implementation team can provide some of the business knowledge based on their experience working in different areas of the company. However, this is one area of the project where the team needs a lot of end-user involvement to define the detail requirements.

The testing activity involves seven steps for each of the transactions included in scope:

1. Create test cases.
2. Prepare test data.
3. Define the expected results.
4. Run the test.
5. Evaluate the test results.
6. Correct problems and retest.
7. Document successful completion of the test.

For each transaction there may be several scenarios or cases to be tested. They usually cover the normal things that occur as well as the exceptions. To get a flavor for the level of testing in this activity, consider some of the things that may need to be tested for a single transaction such as Enter a Sales Order (SO):

- Enter an SO with just one line item.
- Enter an SO with a large number of line items that require multiple entry screens.

- Enter an SO that causes a customer to exceed its credit limit.
- Enter an SO that gives the customer a price discount based on quantity breaks.
- Enter an SO with a manual override on the discount percentage.
- Enter an SO where items are not available in stock and must be back ordered.
- Enter an SO using the customer's item number instead of the organization's number.
- Enter an SO where the customer is exempt from sales tax.
- Enter an SO where freight charges must be calculated.
- Enter an SO where the customer will pick up the item at the shipping dock and should not be charged for freight.
- Enter an SO for an item that is not carried in inventory but can be special ordered.
- Enter an SO where the items must be shipped from several distribution centers.
- Enter an SO that is a release from a blanket order recorded in the system.

Although this list is not complete, even for this one transaction, it gives an idea of the level of detailed testing that needs to take place for each transaction. For each of the cases above there may be one or more tables or switches that need to be set in the system (configured) for these situations to be processed correctly.

The end users who process transactions with the existing systems should be involved in identifying all the different test cases. They know all the variations that occur as a result of business policies and unique customer requirements.

All the test cases should be documented in a testing log. This log can also record the results of each cycle of tests and the actions that were taken to correct errors.

Finding errors is a good thing! The project team does testing to find errors; they should not expect that all the tests would run the first time without error. The idea is to find the errors early, when they are easy to correct. The team also wants to find and correct the bugs before the system is used to train users or process real transactions.

In order to run these tests, it is necessary to set up test data in the system. For our simple sales order tests we need customers, items, multiple distribution centers, and pricing, freight, and sales tax tables. At this stage of the project the team does not want a large numbers of items in the test databases because that would make it more difficult to trace the transactions. Therefore, the team should not dump all the production data from the legacy systems into the test databases. Rather, they should create the minimum amount of test data to run the tests identified.

However, it is helpful to use actual customer names and item numbers in the tests so the end users can relate to the test scenarios. Often a small amount of data can be

entered manually using the entry screens in the system. This implies a hierarchy for the tests. We need to test entering a new customer and changing credit limits before entering transactions for our sales order tests.

A key step in the testing process that is sometimes omitted is documenting the expected results before running the tests. Experience has shown that the expected results need to be put in writing to support an effective testing process. Without this step, people evaluating the test results may check only part of the outputs that are produced or accept the results without sufficient thought or challenge.

As examples, the expected results from sales order tests could include:

- As SO 10887 is created, the five items in inventory for this SO should be reserved for the quantities on the SO and the available-to-promise amounts for each item should be reduced accordingly.
- The customer in test case 4 should have its credit status field changed to "COD only."
- The freight charges for test case 12 should be $26.42.
- The sales tax for test case 14 should be $6.45.
- The freight charge on the pickup order in test case 15 should be zero.

Without documenting the expected results, errors can slip through the cracks and not show up until later in the project when they are more difficult to correct. Even worse, they may be discovered first, after go-live, by the organization's customers in the form of incorrect invoices or wrong shipments.

Once the tests are run, the actual results and the expected results are compared. This is not as straightforward a task as might be expected. When there are discrepancies the team will sometimes find, perhaps after trying several changes to the system, that the system results were correct all along and the expected results were wrong. Researching result gaps requires a good eye and an open mind.

If the system does not perform as designed, it may be necessary to change parameters or data tables to correct the problem. Before rerunning the test it may be necessary to refresh the test data—setting it back to what it was at the beginning of the test. Running several test cycles corrupts the initial test data and the expected results for future tests. Therefore, it is advisable to create a standard test bank (set of test data) that can be used for all the tests. Then the IS support personnel need to provide a mechanism to refresh the test data, as required.

A final testing concern is the fact that changing parameters to fix one problem might cause new problems in another area. A test that worked correctly yesterday might fail today. So, there is a need to run through all the tests multiple times until the system passes all the unit tests in a process area.

The actual and expected results should be retained so project managers and others

working with the team can review the tests. This documentation can be used to research other test problems later in the project.

Testing should go more rapidly if the team is using a proven, preconfigured version of the software. A lot of simple errors that occur when setting up all the system parameters from scratch can be avoided. The team will also be able to start testing earlier with a stable, operational system as a starting point. So, in the areas of configuration and testing, having a preconfigured version of the system is a key factor to support a rapid implementation approach.

CHANGE

Putting in new computer systems produces significant benefits for the organization only if the implementations improve key business processes. It is hard to cost-justify these projects solely on improvements they make to the technical infrastructure. Also, it is clear that there is little benefit from putting in new systems that are not used. Therefore, to get the maximum return on application system investments, people and processes have to change, and for the good. The organization must be able to do things faster, better, or cheaper than they did before the new system was implemented.

The changes that result from a rapid implementation occur in three areas: (1) technology, (2) processes, and (3) organizational structure. There will be many groups of people who will be impacted by the changes. The responses from the project stakeholders (i.e., those impacted by the project) will vary across the spectrum from denial, active resistance, and a wait-and-see attitude to excitement and support.

It is not just management and the end users of these systems who must change. As a result of these projects, the people in the IS department often must learn new skills and utilize different tools in their work. In some cases the custom-developed systems they created (and for which the organization relies on them for knowledge on how they work) will be replaced by standard software developed by a vendor. For the first time, users may know as much or more about the system as the application analyst in the IS department. The implementation team may even be discussing the possibility of outsourcing support and processing of the new system.

The users of the new system will be anxious because getting new systems usually means they must learn new ways to do their work. It will change the way the work is done as well as the tools that are used. The new system may replace the Excel spreadsheets, Access databases, and three-ring binders they really use to do their job. It may also change job roles and responsibilities. Users may also have concerns about how quickly they can learn to use new technologies and complicated systems with different terminology, screens, and reports.

Most of the users are not on the core implementation team. Therefore, they may also have worries about the project team—off in another building or location—de-

signing new processes without understanding all the problems and challenges users face on a daily basis.

Managers, especially supervisors and middle managers, often are concerned about how the changes brought on by the new system will transform the organizational structure and the roles, responsibilities, and authority of their department or group. This is especially the situation when one of the stated benefits of the project is to cut out a layer of middle managers. New business systems can sometimes be used to substitute for one of the primary roles performed by middle management: acting as a channel for passing information up and down from top management to the users.

In addition, the managers often designed the current processes and have a sense of authorship and ownership for the ways things are done. Any suggestions for changes to process designs may be taken personally.

All these issues, whether real or imagined, must be addressed during a rapid implementation. The project team members must understand that perception is often more important than reality. These concerns and fears are important because it is the people issues, rather than technology issues, that usually cause these projects to fail.

Therefore, during a rapid implementation, the team must be proactive in managing change. Given the shorter duration of these projects, the change will come quicker to those impacted by the deployment of the new system. Positive changes do not happen by accident. To ensure proper focus on these aspects of the project, someone on the implementation team should be assigned primary responsibility for change management.

There are a number of activities that can be done during a rapid implementation to help manage the changes that occur. They fall into the following areas:

- Assessment
- Involvement
- Communication
- Documentation
- Training

At the beginning of the project, the team needs to assess the readiness of the stakeholders for the changes that will occur. They need to determine the various groups impacted by the new system and their views on the project. Who will be the supporters for the project and who will be its enemies? What groups will be the potential winners or losers in resources, power, and responsibilities once the new system is implemented? Who are the leaders and influencers the team needs to win over? What can be done to develop power users and champions for the new system and the changes it produces? What do the stakeholders know about the project? How much of what they know is accurate?

The project team will perform this change readiness assessment sometimes

through survey documents but more frequently through discussions with a lot of people throughout the organization.

A key insight on human nature will have a dramatic effect on the success of the project: no involvement, no commitment. The stakeholders need to be, and feel, actively involved in the design, testing, and rollout of the new system. The implementation team has to have a core group of people working full time on the project; but a lot of other people need to participate in project activities. The nonteam members need to provide information and advice on current processes, system requirements, design alternatives, training approaches, test cases, deployment strategies, and organizational changes.

The worst thing that can happen is for the users and managers to feel that their experience and insights are not valued in the design of the new system. They must not believe that someone else is determining their future without asking for their input. One way to ensure that this does not happen is to proactively work on the two-way communications between the team and its stakeholders.

There is no such thing as too much communication in these projects. Communication needs to come from top management, the core team, and the extended team of stakeholders. The themes to be communicated are simple: This is an important project for the organization. It has the support of top management. This is when the new system will go live. This is the status of the project. Several different methods will be used to get the appropriate involvement of the people in the organization. These are the ways the organization will ensure that people are ready for the new system and their changed jobs.

The modes of communication are briefings at meetings, status reports, newsletters, voice-mail, and email messages. And in this new web economy, the team may even create a project web site, on an intranet or extranet, to communicate throughout the organization.

Good documentation on the new system is important. Often there was never very good written documentation for the old legacy systems. As a result users had to learn through trial-and-error and from information passed down by more experienced employees. With the fast pace of a rapid implementation, there is a need for better aids to enable users to get up to speed more quickly.

The primary documentation for standard systems should be available from the vendor in the form of system manuals and online help text. Often that information is available in electronic format and it can be tailored to the needs of the organization. In addition, the vendor or the system integrator should provide user procedures templates and process flow diagrams to support development of customized user procedures. These procedures can include policies and guidance on how the organization will use the system.

The availability of quality documentation should be a requirement in the selection of a package that will be implemented with a rapid approach. There is just not enough time in these projects to develop this documentation from scratch.

The users must be adequately trained for a rapid implementation (or any implementation) project to be successful. The people using the system need to be comfortable in their understanding of how to do their jobs using the new system. Good end-user training requires quality training materials and well-prepared instructors.

The key resources used in developing training materials should be the user procedures developed by the core team and standard training materials available from the package vendor. Increasingly, the vendors provide some of their training courses in the form of CD or web-based just-in-time instructional materials. Most of the training materials come in the form of instructor-led classes. The end-user training is given in the Prepare activities, just before go-live.

Most projects will use a train-the-trainer strategy to deliver the training. Under this approach, the core team members and power users attend vendor training sessions to learn about the package and its capabilities. They also develop knowledge about the system by working side by side with the consultants on the project team. These organizational representatives then prepare and deliver the initial training to the end users.

There are definite advantages in having people from the organization—not the vendor or the consultants—deliver the training. These people can better teach the users the new system because they are family, already have credibility within the organization, and can use organization-specific examples throughout the training.

All these change activities are important in managing the changes that will result from implementing new systems and business processes. Often they are either ignored or short-changed. This happens because much of this work occurs toward the end of the project, as time and budget are running out and the project is usually behind schedule. However, these activities are some of the most important things that must be done well to make a rapid implementation successful.

SUPPORT

Support refers to the IT activities that must go on, often behind the scenes, to ensure that the rest of the implementation teams have a developmental sand-box in which to test various system configurations. The IS personnel also work during the project to prepare a production environment for use to process real transactions after cutover. Although it has been stated several times in this book that a rapid implementation is business-driven, there are still a lot of technical tasks that have to be done properly to make these projects successful.

The IS department must get an early jump on creating a development environment for the project team. Some of the tasks required to set up this environment have long lead times. Since the team technical environment should be ready when the team members return from preliminary training on the new system, most of these activities must occur during the Commit period of the project.

The project war room must be set up with workstations for all the team members. These workstations should be connected to a local area network so there can be common access to all project deliverables and working documents. A phone should also be available for each team member. In addition, printers, copiers, and fax machines should be available in the project work area. The team will also need a display device to project package screen images for meetings, demos, and testing.

The vendor's software must be loaded on a development server. This is often a smaller server than will be used for the production system. If not already available, this hardware and other network devices may have to be purchased for the project. This same hardware can be used after the project is completed as a developmental environment to test package upgrades, vendor code patches, and configuration changes.

This development system supports the project team, users who are assisting in the design and testing of the system, and IS personnel developing data conversion, interface, and reporting programs. In addition to the vendor's standard software, a database management system and preconfigured versions of the system may also have to be installed.

Once the initial development system is operational, representatives from IS must assume the role of package system administrators and perform other technical support tasks during the project. These support tasks include:

- Creating development, test, and production instances of the system and migrating parameter settings and data between the different versions of the system
- Working with system security personnel or internal auditors to set up security (access authorizations) for the project team members and researching, defining, and establishing all the access capabilities for the end users of the production system
- Establishing backup and recovery capabilities for the different instances of the system
- Programming data-conversion programs and creating data files from the legacy system that can be used to import data into the new system
- Tuning the databases so they operate efficiently with the new applications
- Supporting project team members as problems occur that require maintenance on workstations, software, or peripheral devices
- Researching potential system bugs and working with the vendor's technical support personnel to install code patches to resolve system errors
- Developing special reports and forms using the tools available with the software and other common report-writing tools
- Developing program interfaces between the new system and existing legacy systems so data can be passed back and forth between these systems in real time or with periodic updates to system files

These activities require strong technical skills. Many also require knowledge about the current systems used by the organization. Unfortunately, information about legacy systems is usually not well documented and often exists only in the heads of the IS personnel. Since most of these IT activities require significant amounts of detail work, having full-time IS personnel assigned to the core team is critical to the success of the project.

The new system may use technologies (e.g., databases, operating systems, development languages, and tools) that are new to the IS organization. Consequently, IS personnel must receive training on the technical aspects of the package while the other team members learn the functional aspects.

To ensure the timely completion of the technical activities, the project team often leverages the technical resources of the vendor and other organizations. Vendor personnel may even install the software of the development server. In addition to this initial support, it may be necessary to include, as a full-time member of the implementation team, a technical consultant who has implemented this package before, in the same technical environment. Even with these outside resources, there are a lot of technical challenges to implementing these packages. A lot of IT knowledge transfer and learning must occur throughout the project so the organization's IS department can take on technical support of the system once it is put into production.

A new trend is to use an application service provider (ASP) to host the processing of the system and take some of this load off the organization's IS department. If the right ASP is selected and appropriate contract terms negotiated, this approach can help manage some of the IT risks of the project. This option and a more detailed discussion of some of the technical issues of implementation are covered in Chapter 7.

CONVERT

One of the potential hurdles to rapid implementations is the time that it takes to convert data from existing systems into databases for the new system. The data conversion challenges are in five areas:

1. Mapping
2. Cleanup
3. Loading
4. Verification
5. Synchronization

Mapping is the process of examining the information required in the new system and determining where each piece of data will come from. This is complicated by the fact that the new system will have literally thousands of data items in its database but

only a subset of these are required fields. Therefore, a first step is to determine which fields will have to be populated before going live on the new system.

The fields that will be required depend on the transactions included in the scope of the implementation phase. If certain processes or transactions are dropped from scope, there may be less data to convert. However, if new transactions are added midway through the project, it may be necessary to plan for additional data requirements.

In a traditional multiyear implementation this is less of a problem. The implementation team would go through a formal design process and produce a system design that would then be frozen as configuration and testing were begun. However, in a rapid implementation a lot of these activities happen in parallel and require close communication and coordination between team members.

Much of the data for the new system will come from existing systems. However, it is common to find certain items of data that are needed that do not even exist in the legacy systems. As a result, some of the data will have to be created for use in the new system. Data creation is a task better delegated to the eventual system users than the project team members. However, it needs to be managed as part of the overall project.

While most of the data to be converted will come from legacy systems, it is not uncommon to find some of this data in manual systems and other ancillary storage areas. Some key organization data is in an Access database or Excel spreadsheet on someone's desktop computer. Just finding out where the information comes from to support the current processes is sometimes a challenge. Often the same data is stored in multiple places, for example, in the legacy system and in an Excel spreadsheet. When this happens, the team needs to determine which source has the most accurate and complete data.

This mapping activity is an ongoing task throughout the implementation project and requires close coordination between the IS personnel who know the record layouts in the legacy systems and the process analysts who are determining which parts of the standard package will be implemented.

Once the data is found, the real work begins. Data integrity is often a big problem that requires a large number of people to be involved in cleanup activities. Data integrity problems are not caused by the implementation project; but they become more visible during a system implementation and must be resolved before the data is loaded into the new system. The old saying "garbage in, garbage out" still applies.

The old systems often have data that is redundant, inaccurate, inconsistent, and incomplete. There are many examples of the symptoms: the same customers or vendors are listed multiple times in the legacy system files with different spellings and numbers. The materials and quantities in bills of materials are often wrong, resulting in invalid inventory valuation and workarounds to produce goods on the shop floor. The balances in subsidiary files do not reconcile with the control totals in the general ledger. Thousands of items appear in the inventory files that have not been produced or sold any time in the last 10 years.

Not all the data in the old system has to be converted to the new system. In a rapid implementation the team often uses various techniques (e.g., ABC analysis, 80/20 rule) to determine what really needs to be converted to be able to run the new system. Some data can be created post-go-live to support specific transactions as they occur. The implementation team should convert only what is really required.

Usually, people using the legacy systems clean up the data in these systems before data conversion programs are run to transfer this information to the new system. In situations where new data is created, this new data can be captured in Access or other databases for automatic conversion at a later time. Consequently, cleanup can be done in parallel with other implementation activities. This also means cleanup activities can start early in the project. There are often surprises in this area. Therefore, the team wants to identify problems as early as possible, while there is still time to resolve them.

Data gets loaded into the new production databases through three primary means. Most of the packages have data-conversion utilities that read external files that have been organized in a prescribed format and load the data into the production databases. In addition, the programmers from the IS department may code custom programs to load the new data directly into the production files. Finally, it is often necessary to load some of the data manually, using the input screens of the new system. In all three methods the data should be subject to the same edit and validation checks that would occur if the data were loaded through regular system processes.

Once the data has been loaded into the system, someone needs to verify that the data was transferred accurately and completely. This task is usually considered standard procedure for financial data but it is just as important for operational data. Various methods are used to verify the validity of the conversion activities.

Integrity checks include looking at the total number of records in each of the tables before and after the conversion runs, calculating control totals for numerical fields, and performing spot checks of all the fields in a sample of the records that were converted. It is best to do some of these specific checks using the screens in the application to make sure that the new system can read the data as stored.

The data-conversion procedures and tools should be tested as early as possible—perhaps as early as transactions testing. Transactions testing uses small quantities of data. Some of this data could be made available using data-conversion routines.

The data-conversion process for the production database should be tested as one of the first steps in integration testing, in the Prepare activity. The team cannot afford to discover a few weeks before going live that they do not have all the data needed for the new system, or any way to get data out of the old system. One of a project manager's responsibilities in rapid implementations is to anticipate and manage the project so there are no surprises. Data integrity and conversion is an area that must be monitored and managed carefully or else it may jeopardize the implementation schedule.

Some of the data may be converted ahead of the go-live date because often there is not enough time or resources to do all the conversions at once. Because some of the data is more static than other data, there are alternative strategies that can be used for doing data conversion, including early conversion and synchronization. For example, the team can convert the customer file and inventory master file weeks before cutover as long as a control process is put in place to ensure that any customers or items added to the legacy systems are also added to the new production databases. In addition, any modifications to records in the legacy systems already converted also have to be done in both systems.

Some files and items with a lot of activity (e.g., open orders) have to be converted right before the go-live date. Therefore, these items are often converted on the weekend before transition to the new system, while processing is frozen on the old system.

PREPARE

By the time the team gets to this activity, they are finally able to see the light at the end of the tunnel. They have done a lot of work throughout the project to get ready for the final push to go-live. How well they prepare for the cutover to the new system in the final weeks of the project will often determine whether the light they see is daylight or an approaching train. A lot of good work can be put at risk by sloppy work right before go-live. The team needs a smooth transition to the new system.

There are several key tasks that take place in the final four to six weeks of a rapid implementation before beginning operation with the new system:

- Integration testing
- End-user training
- Preparing the production environment
- Data conversion
- Establishing ongoing support capabilities

The team did transaction or unit testing in the Analyze-Configure-Test loop. Therefore, they know that the system is set up so that individual transactions work. Now is time to test the business processes end-to-end.

Extending the earlier manufacturing example, for order fulfillment the team tests the entire process from preparing a quote, converting it into an order, reserving inventory, generating a production order, issuing materials to production, recording completed production and scrap, shipping the order, preparing an invoice, recognizing revenue and the corresponding receivable from the customer, and applying the payment received.

Depending on the implementation scope, there may be 2 to 10 processes to test for completeness and integration in this activity. For example, after testing the fulfillment transactions the team may next test the financial close process or payroll processing. Then they may turn to production scheduling or procurement.

Integration testing takes time because of the number of different processes to thoroughly test and the number of steps in each process. It requires a cross-functional team to verify the effects throughout the system. Like transaction testing, it requires preparation of a test plan, test data, and expected results. And, like any testing activity, integration testing will uncover errors that need to be researched, fixed (perhaps by table or parameter changes), and then retested.

Managing the testing process is challenging because of the number of people involved, the complexity of the tests that address integration across the entire system, and the need for strict configuration and version control. Integration testing, as well as the other activities in Prepare, occurs at the end of the project as budgets and time are running out. This is an area of the project where there is a lot of pressure to take shortcuts and not follow a thorough test plan. These pressures must be resisted. No one benefits from having a system go into production that does not work properly.

Some members of the implementation team can complete the preparation of end-user training materials while others conduct the integration tests. However, hands-on training should not occur until the system correctly processes transactions. The team will lose a lot of credibility and create anxiety throughout the organization if the new system does not work properly in the training sessions.

End-user training should leverage the user procedures and system documentation that have been developed by the team. These are the aids the users should refer to as they perform their jobs; so, they should be introduced and used in the training courses. These procedures walk the users through the steps for doing their jobs with the new system. Using this approach, training becomes a simulation of real transaction processing using the job aids and a new system.

Depending on the number of users to be trained and the number of times each training session must be offered, it is not unusual for end-user training to take two to four weeks. Users may receive from several days to a week of training on the details of how to do their work using the new system. All users should begin their training with a system overview that includes practice navigating around the package.

End-user training cannot be scheduled too early in the project or people will forget what they learned before they get to apply it. However, if it is started too late, there will not be enough time to do sufficient training. Since end users still have their regular jobs to do while attending training on the new system, scheduling the training to meet project and job demands is a complicated administrative task.

Providing training of high quality is important to ensure people feel confident that they know how to do their jobs with the new system. It is an investment in the organization's people and a prerequisite to achieving the benefits from the system.

Training also provides an opportunity to sell the new system to the users. Therefore, it is a key change management tool.

While integration testing and end-user training are going on, the IS department is working hard to finish setting up the production environment for the new system. In some cases, this requires purchasing and installing new hardware to handle the large number of users and high volume of transactions that will be processed daily after cutover. In other cases, the new system will run on the existing technical infrastructure. In either case, it is necessary to stress-test the system to ensure that it can handle the production load.

Another time-consuming task for IS team members is setting up security authorizations for all the users of the new system. Users should have access to all the parts of the system required to do their jobs, but only to those for which they are authorized. This task goes quicker if it is possible to create role profiles that define authorizations for a particular job. These profiles can then be cloned to develop access lists for all the users doing the same tasks. The internal audit department should be actively involved in this activity and all others involving internal control considerations.

By the time the team has reached these last weeks of the project, all data-conversion procedures and tools should have been thoroughly tested. As the team approaches go-live, it must run the data-conversion procedures and programs to populate the production databases. Throughout this process, audit trails are produced to provide information to research problems and document the conversion results.

Error reports are produced throughout the conversion process. These reports show what information was not accepted by the new system. There are a number of reasons for posting failures. The file sizes on the new system may have been set too small. In addition, some of the data may not have been of the quality required by the new system. This converted data should be put through the same edits and validations that occur during normal transaction processing on the system. If the legacy data cleanup efforts were not completed in time, follow-up activities will have to be scheduled to clear data exceptions and complete the data-conversion process.

Finally, before go-live, the team needs to activate help desk and systems support functions to help users and managers with problems once they start processing on the new system. The project team members will be around for a short period after go-live to assist in this effort. But a permanent function should be established for the ongoing support and maintenance of the system.

This support function may include a help desk, support resources in the IS department, and power users in functional departments who can answer questions and resolve problems people will have in the production environment. Knowledge transfer to this support group should take place throughout the project. Often, some of the team members take on support roles after the implementation project is finished.

GO-LIVE

Go-live is a hectic and confusing time. People have been trained on the new system but now they have to use it in their daily jobs. Often, they find out that they did not understand how to use the new system as well as they thought. The organization has real customer orders coming in and needs to keep customers happy and ship goods on time—even as it works through the inevitable problems that will occur in the transition to any new system.

The organization should expect problems for a period of time following the cutover. This time period varies from a few days to a few months, depending on the situation. In spite of all the preparation and contingency planning that was done, problems will occur. Systems are developed by and used by human beings who, despite the best of intentions, make errors. So things will happen that were not expected and tasks will be done differently than planned.

The limited scope of a rapid implementation may help to minimize the amount of disruption that occurs. However, the organization has had less time to get ready for the new system with a rapid approach. Experience from a large number of package implementations has shown that all organizations should expect a performance dip immediately following the cutover to the new system.

People need a lot of support in the days immediately after beginning to use a new system. Help desk personnel will be available to provide support. However, during the first weeks after go-live, there need to be additional resources assigned to diagnose and solve problems that surface. The project team members provide a lot of this additional support. They often engage in triage activities to identify the biggest problems and ensure that adequate priorities and resources are available to solve them.

During this time people will be tempted to fall back to their old ways of doing things. Therefore, one of the key post-go-live tasks is to remove the old systems and tools and force the users to rely on the new system. This may mean removing access to legacy systems, Excel spreadsheets, and other applications running on the LAN or individual workstations. Management and the project team need to enforce use of the new system.

Some people may try to sabotage the new system and its processes. They may believe that if the new way fails, management will go back to the old way of doing things. Again, these activities and this attitude should be nipped in the bud. Management should show its support and commitment to the new system. This requires visibility and communication in the key days after processing has begun with the new system.

A key, and often missed, opportunity during the final days of the implementation is to capture the lessons learned from the project. The organization should be trying to develop expertise in making rapid changes to business processes and the technologies that support them. If they want the next rapid implementation project to be

better, the organization should learn from the mistakes and successes of its prior projects.

A postmortem should take a realistic looks at the project while the information is fresh in people's minds. What went well and should be reproduced in the next rapid implementation project? What did not and therefore requires a different approach in future projects? Given 20/20 hindsight, what could the project sponsor, project manager, and the team have done differently? These items should be discussed in detail, maybe with the same group that attended the kickoff meeting. This could be considered a project close meeting.

This meeting should be followed by a celebration dinner or event for the project. Participating in a rapid implementation project requires a lot of hard work and places a lot of stress on all the project team members. If the project was successful, you can bet that a lot of people made a lot of sacrifices to make that happen.

What the team has accomplished is important to the organization and should be recognized and rewarded as such. Top management should make its support and appreciation for these efforts visible to the team and the rest of the organization. This is especially important if they want others to sign up for these projects in the future.

After the project is completed, management should make good on any commitments made to the team members before the project began. And immediately following the celebration, planning should begin for the next rapid implementation project. No organization is ever finished adapting its processes and systems to changing business requirements.

IMPROVE

In order to do rapid implementations, trade-offs must be made. The benefits from a rapid implementation approach do not come without costs. The assumption is that before management chose a rapid implementation approach for the system that just went live, they deemed the trade-offs acceptable. These trade-offs result from several characteristics required to make this approach successful:

- Time boxing
- Aggressive scope management
- Package-driven process design
- No customizations

As will be discussed in more detail in Chapter 4, many of the activities in a rapid implementation project are time boxed. This means that the team does the best they can on specific activities—within some pretty strict time and resources constraints. Priority was given to making business process and system changes in a short period

of time rather than necessarily dotting all the i's and crossing all the t's. These projects are done rapidly in order to accelerate the benefits and change business processes to respond to significant, often short-term business needs.

However, taking this approach does not mean that there is not a case for going back and enhancing the work that was done, at some later time. Examples of things that may be done in the future include preparing additional procedures for handling exception situations, implementing additional processes, and developing and delivering additional training in areas where people are having problems.

The scope of the rapid implementation project is aggressively managed. The 80/20 rule is the guideline for establishing priorities. The implementation team tries to identify and implement the 20 percent of the business processes and systems functionality that provide 80 percent of the benefits. That is the right thing to do if the organization wants to maximize the return on the implementation investment. But again, there may be merit in implementing some of the remaining 80 percent of functionality in a future project.

What additional functionality should be implemented may not be clear at this point in the system life cycle. Sometimes it is best to require the users to make do with the system they have before deciding what additional functionality is *really* required. A good example is making them use the standard reports from the package for a period of time before deciding that customized reports are needed. The fact is that users will be in a better position to determine priorities for future enhancements to a system once they have experience working with the processes that were implemented.

One of the design guidelines for rapid implementation is that it is better to have the organization change its business processes to conform to the way that is supported by the package than to change the package. It is very expensive (time and money) to make modifications to the vendor's software to support the current way that processes are performed. Even worse, these programming changes will have to be reapplied and tested every time the organization implements newer versions of the vendor's package.

Many organizations have some pretty bad business processes that users want to replicate in any new system. Also, if the organization does the same process three different ways in its three locations, managers and users want all three alternatives supported by the software. So, there are good reasons not to automatically implement the existing practices with the new software.

Many best practices are incorporated in the design of the standard packages. These practices leverage the benefits of real-time, integrated systems with enhanced functionality that was often never available in an organization's legacy systems. Also, there are a lot of benefits from standardizing processes within an organization (i.e., deciding on the best way to do things and having everyone do it that way, without exception). Therefore, in many cases, conforming to the way the package does things is a good strategy.

A rapid implementation strategy generally does not support modification of the vendor's code. The project team uses the plain-vanilla version of the vendor's software and then tailors the package to the needs of the organization through the use of parameter tables and reporting tools.

However, there are going to be situations where there is a good reason to add bolt-on functionality to do things not supported by the package. Bolt-on functionality is interfaced to the vendor's system and usually does not involve modifying the vendor's code.

This new functionality may be provided by another vendor's package or may be custom-developed by the organization's IS department, with or without the assistance of consultants. As long as there is a valid business case for these enhancements, there is nothing in the rapid implementation approach that says that these changes cannot be made in a future implementation phase. However, before jumping into these changes, the organization should give the system, as implemented, a chance. Often, after spending some time using the new system, the users will come up with ways to handle difficult requirements using the existing capabilities of the package.

A key message from the roadmap is that the initial rapid implementation project is the first phase in the life cycle of the new system. The organization and its customers and suppliers will be constantly changing. Business strategies, models, processes, and organizational structure will change. Therefore, the system will have to continuously change to meet these changing requirements.

Consequently, there will be many opportunities to change and improve the system that has been implemented. The organization should establish a process to identify and prioritize these changes and, when appropriate, begin the Commit activity all over again.