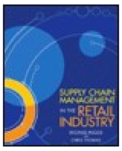


Supply Chain Management in the Retail Industry

by Michael Hugos and Chris Thomas
John Wiley & Sons (US). (c) 2006. Copying Prohibited.

Chapter 9: Developing Supply Chain Systems

Overview

Much of the future success of retailers depends on their (and their supply chains') ability to adapt to technological advances. Apparently, they are catching on. The demand for information technology (IT) staff members in the retail industry grew 25 percent in the first six months of 2004, with no letup in sight.

As Jon Butterfield, managing director of IT Business for Spring (a British-based recruitment consulting company), told *Computer Weekly* magazine: "There is good job security in the retail sector, with an exciting range of technologies to manage." [1]

How often do you hear *that* about *any* type of employment nowadays? Luckily, you don't have to be in an IT position with a major global retailer to be part of the excitement—you could lead, or be a member of, a project team that implements a new system.

Chapter 8 introduced a three-step process to create new systems to improve either a business or a supply chain of businesses using computer technology. The three steps are define, design, and build. The first step (define) was also discussed in Chapter 8; this chapter presents the last two steps in the process, which include the following tasks:

- Organize a systems development project for a new supply chain computer system.
- Investigate supply chain processes and document the findings.
- Produce detailed specifications for a conceptual system design.
- Test the design and/or other vendors' applications.
- Create accurate project plans and budgets based on the detailed system specifications.
- Evaluate progress on projects and recognize problems as they emerge.

The basic structure of this planning process, especially the portion introduced in Chapter 8, can be adapted for many different kinds of projects. But this chapter deals with designing and/or upgrading a store's computer systems. Often, to do what a store needs done or to work with its partners in a supply chain, an existing computer system must be enhanced or a new system needs to be built. This chapter presents a process to follow to create more detailed system designs, and to build those systems if necessary.

By "build," we don't mean you *have* to program your own computers from scratch! A business will frequently employ the help of consultants and software vendors to do this work. However, no company can delegate the work entirely to outsiders and expect that its best interests will be served. The retail project team must be able to show the programmers or software developers, step by step, exactly what they need the system to do, and when, and why. Or, if the store is considering purchasing a packaged software application—sometimes at the insistence of another supply chain member—this basic background knowledge will help the store's management team decide if the system features and price tag are really worth it. This is why it is important even for non-IT professionals to understand how systems development works.

[1]Antony Savvas, "Drive to roll out new technologies makes retail a hotspot for IT job opportunities," *Computer Weekly*, June 8, 2004.

Organizing a Systems Development Project

A simple three-step process may be used to organize the project. For each step, a certain amount of time and a budget must be allocated. This provides a framework of regular deadlines, through which each step is completed within the boundaries of its time and budget limits.

Before we introduce the three-step process, there is a short list of six principles that should be used to run a project. Just as there are guidelines for conceptual design, which were listed in Chapter 8, if these six principles are consistently applied, the probability of success for the project is high. However, if any one of these principles is ignored, then special precautions must be taken to compensate for it; and if two or more principles are violated, then the project is almost sure to fail.

Supply Chain Skills—Six Principles of Project Management

- Every project needs a full-time leader with overall responsibility and authority.
- Define a set of measurable and nonoverlapping objectives that are necessary and sufficient to accomplish the project goal or mission.
- Assign project objectives to teams of two to seven people with hands-on team leaders and the appropriate mix of business and technical skills.
- Tell the teams *what* to do but not *how* to do it.
- Break project work into tasks that are each a week or less in duration, and produce something of value to the business every 30 to 90 days.
- Provide project office staff to work with the project leader and team leaders to update plans and budgets.

---

1. **Every project needs a full-time project leader with overall responsibility and the appropriate authority.** There must be an individual who is responsible for the project's success and totally focused on getting the job done. This person must also have the authority to make decisions and delegate responsibilities. It is wise to have a steering committee or management oversight group in place to whom the project leader reports, but a committee cannot generally make decisions in a timely manner. If there is no one person in this role, then the project progress and cost will reflect it. Progress will be slower, and costs will be higher.

2. **A set of measurable and nonoverlapping objectives must be defined that are necessary and sufficient to accomplish the project goal or mission.** It is crucial that clear project objectives be defined so that the people who are assigned responsibility to achieve the objectives know exactly what is expected of them. We mention "nonoverlapping" because overlapping objectives cause confusion and conflict between the teams assigned to work on them. Who does what? Who's usurping whose authority? Whose job is this, anyway? Minimize these conflicts by being specific about the objectives.

   Along the same lines, make sure that each objective is absolutely necessary to the accomplishment of the project goal. Do not pursue an objective just because it seems like a good idea. And the list of objectives must "add up" to a final result—that is, if each objective is achieved, then the mission or goal has been accomplished. The objectives must cover *everything* that needs to happen.

3. Assign project objectives to teams of two to seven people with the appropriate mix of business and technical skills, and assign handson team leaders. If you're in a position to select the project team members, choose the people who, in your judgment, have among them the necessary business and technical skills and the experience to address the issues that will arise in the work that is being delegated to them. A team is a group of people with complementary skills who organize themselves so that all members can contribute their strengths and not be penalized for their weaknesses.

   Each member of the team concentrates on the aspects of designing and building the system that they are good at and/or most interested in. For the most part, no one is required to do things they are not interested in or not good at. Within a team, the operative word is "we," not "me." The whole team is rewarded for successes and takes responsibility for mistakes. Singling out superstars or scapegoats undermines team morale and performance.

4. **Tell the teams *what* to do—but not *how* to do it.** Point a project team in the right direction by giving it a well-defined project goal and clearly identifying the project objectives the team is responsible for. If the project goal is "the game," and the objectives are "what the team must do to be successful in the game," then the team itself must make "the game plan" for meeting the objectives.

   The famous U.S. military general George Patton said, "Tell people what you want but don't tell them how to do it—you will be surprised by their resourcefulness in accomplishing their tasks." The teams can make changes or additions to the objectives they are given, as long as the project leader agrees that the modified objectives are still necessary and sufficient to accomplish the project goal.

5. **Break project work into tasks that are each a week or less in duration and that produce something of value to the business every 30 to 90 days.** This kind of schedule allows the project team to keep an eye on the prize. Each task must have a well-defined deliverable and can be tracked: in progress, delayed, or completed. Do not fall

into the trap of tracking tasks by a percentage of completion, because it is often unclear what "percent complete" really means. What matters is whether the task deliverable has been produced—and if not, when it will be produced. The project leader must be able to track progress at this level of detail in order to understand what's really going on week to week, and to keep accurate projections of the time and cost to complete each of the project's objectives.

Multiweek tasks make progress hard to measure, and they are the ones that inevitably cause cost overruns and confusion, which compounds when they are reported by "percent complete." They seem to be making good progress—and then in the last week, they suddenly turn out to be nowhere near completion and the team members ask for several more weeks to complete them. To avoid this problem, break big tasks into a set of subtasks, each of which require a week or less to complete.

These tasks should combine to produce *something that is of value to the business* every 30 to 90 days. This provides the opportunity for the business to verify that the project is on the right track. It also provides deliverables that the business can perhaps begin to use even before the entire project is complete. In this way, the company may start to recoup the cost of the project.

6. **Every project needs an office staff, dedicated to this particular project, to work with the project leader and team leaders to update plans and budgets.** The project plan and budget are analogous to the profit and loss statements of a business. They must be updated continuously and accurately in order to provide the people running the project with the information they need to make good decisions. There is a common but misguided notion that the project leader and team leaders should be the ones who keep the plans updated, which is like saying the president of a company and its managers should spend their time keeping the company's books! The project leader and the team leaders are responsible for creating the initial plan and budget. After that, their time, attention, and energy are better spent making the plan a reality.

   Just as there is an accounting department to keep a company's books, there should be administrative staff members assigned to each project to keep that project's plans and budgets. This staff reports to the project leader and works with the team leaders on a weekly basis to review and update the plans and budgets. In this way the project leader can accurately monitor project progress, and the team leaders are able to focus on running their teams—and not on filling out reports.

Supply Chain Skills—Types of Systems In Retail

If you think your little corner of the retail world has hardly been touched by technology, think again! Here's just a partial list of the types of technology that store managers and employees may have to deal with daily. And, remember, each of them has its own hardware, software, and system requirements. Most involve the storage and sharing of data on demand.

Bar code readers	Handheld inventory terminals	Point-of-sale terminals (cash wraps, electronic cash registers)
Cash drawers	Internet ordering capabilities	Printers
Checkout lanes (self-service)	Kiosks (self-service)	Security systems
Credit card readers	Labeling and tagging systems	Touch screens
Data warehouses	Laptops	Uninterruptible power supplies (UPS)
Electronic article surveillance	Line (queue) management	Visual merchandising systems
Electronic scales	Local area networks	
E-mail systems	Pin pad devices	

Designing a Supply Chain System

In Chapter 8, we went through the process of creating a conceptual system design—the high-level framework or skeleton of the project. Now it's time to put some meat on the bones, so to speak, by creating more detailed system specifications. In so doing, we will create a detailed project plan and budget, which are the last steps before actually building the system.

This is the point at which the people who will work on the project get to take a look at what senior management wants and figure out how they will do it. In a system used by multiple companies in a supply chain, cross-functional teams from different companies may divide up the work and share the results. This is also where adjustments and refinements are made to the project objectives, as the people who will build the system consider the realities of the job before them.

By the end of the design step, it is usually possible to predict the success or failure of any project. If the people on the project finish this phase with a clear set of system design specifications and confidence in their ability to build a system to these specifications, then the project will succeed. If the opposite occurs— the design specifications are vague, incomplete, or hard to understand and if people are ambivalent about their chances for success or uncertain about the accuracy of their cost estimates—then the project will likely fail.

The phase begins with the project leader reviewing the project goal, the conceptual system design, and the objectives with the project work group. The work group is composed of business and technical people who have the necessary mix of business and technical skills and experience needed to do the detailed system design. It is important for the people to understand senior management's intentions and the project's goal. Specific issues relating to the project objectives and budget can be investigated during this phase. This is the time when the work group should make adjustments, if necessary— before starting on the design details.

Once the people on the project work group understand the goal and the objectives, they (and their project leader) lay out a work plan for this phase, in which two major steps are accomplished:

1. Create detailed process flow diagrams for the new system.
2. Build and test the system prototype (i.e., the user interface and technical architecture).

The technique called **time boxing** introduced in Chapter 8 may be used to create a work schedule and stick to it, by breaking each activity into a set of tasks and assigning times to complete them. (Avoid the temptation here to spend extra time doing excessive amounts of analysis and checking and rechecking the results that come out of each activity.)

These two steps should take somewhere from one to three months to complete. For the most part, work on each of the two activities can proceed simultaneously. In some cases this phase could be finished in less than one month, but in no case should it drag on any longer than three months. If it does, this indicates a lack of clear focus or of effective organization (or both) on the project.

Creating a System Prototype

The project team should review the system performance criteria as described in the "define" phase. The criteria will be some mix of performance targets from the four categories, which should be familiar by now: customer service, internal efficiency, demand flexibility, and product development.

It's almost time to sketch process flows again, an activity known as **process mapping**. But before doing so, the project leader must brainstorm with the teams on ways to meet the criteria, encouraging a free flow of ideas. It is important to guide the group away from the common trap of premature criticism and dwelling on why things cannot be done and instead shift the focus to how things *might* be done. Generate as many ideas as possible, since these are the raw material to be blended together to create the designs for the new system process flows. Then, start mapping, much as we discussed in Chapter 8.

Once the process flows have been sketched out, the design sessions can begin to focus on how technology will be used to support this process. The design team must envision how people who use the system will interact with the technology supporting it. In this area, look for ways to automate the rote and repetitive work, and to empower the problem-solving and decision-making tasks. People (and those using computers are no exception) usually don't like to do the rote, repetitive work because it is boring; but they do like problem solving and decision making because it is creative and interactive.

Remember that people are the spark that animates a business process, not computers. Design systems that will be a rewarding experience for people— where they feel they are in control, not the computers. A good system—a retail Web site, for example—empowers people with access to information so that they can find things faster and make better decisions.

A **system prototype** is the model or sample system that the team believes will effectively support these new processes. The **process decomposition diagrams** (the fancy term for the maps or flowcharts) provide the processing logic and sequences to be used and indicate the kinds and volumes of data that the new system needs to handle.

There are two kinds of system prototypes: user interface prototypes and technical architecture prototypes. You might compare it to designing a building—you'd have to create two kinds of designs:

- The first is the floor plan and façade of the building to show what the building will look like.

- The second is the design of the structural, electrical, and plumbing components needed to support the specified floor plan and façade. This design shows how the building will be built.

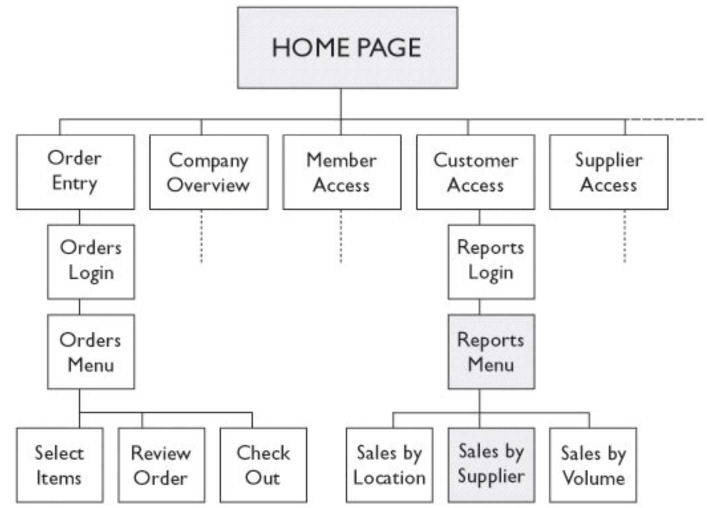
In system design, the **user interface** can be thought of as the floor plan and façade. It shows what the system will look like and how a person would move through the system. The structural engineering for a building is called the **technical architecture** of a system—the hardware, operating system, and database software that will be used to support the user interface.

Both the user interface and the technical architecture designs are created in parallel. It is an iterative process that makes trade-offs among the user interface, the system functionality, and the underlying technical architecture. The aim is to find an overall design that provides a good balance between system functionality and ease of use, minimizing the complexity of the underlying technical architecture. The key is to find ways to use relatively simple technical architectures to creatively support a wide variety of user interfaces and system features.

Prototype the User Interface

Entire books have written about the design of user interfaces for computer systems. One of the most basic but important concepts is that the user interface should be "intuitively obvious." This means a person who already knows how to do the activity for which the system was designed should be able to figure out the basics of the new system within about 20 minutes of playing around with it, just trying things out. The better a system interface design, the less the amount of training is required to teach a person to use it.

The process flows become the model for the prototype of the user interface, as shown in the diagram. The process decomposition diagram (Figure 9-1) shows what activities are performed in what sequence and what data is needed to support these activities. The sequence that leads a person from one computer screen to another should be designed to map closely to the process flows and should allow the user to manipulate the data involved in the process.



This screen map shows a partial layout of the Network Services' Web site. The shaded screens are shown in greater detail with screen layouts below

Figure 9-1. Network Services Company Web site user interface.

Prototype the Technical Architecture

As the user interface design is progressing, a parallel design effort is under way to select and test the technical components that will be used to build the system. Decisions should be made on the computer hardware and software to be used. How will remote users be able to connect to the system? How secure must the data be to satisfy the users? The database and other packages must be specified and a programming language chosen if there is custom coding to be done.

All of these components must be assembled in a test environment and tried out to see if they work as advertised. Sometimes it's as simple as connecting the pieces and making sure they actually do work the way the vendors said they would.

If there's one thing you will notice in the IT end of retail, it is that everyone is trying to sell new technology to stores—systems that are faster, bigger, more convenient—and of course, *brand-new*, probably on sale, and so on. (You're retailers, so you know the buy-now-or-miss-out-forever sales pitch!) But until a technical component has been in use for at least two years, it is not wise to take any of the vendors' published performance statistics at face value. There's just not a wide enough base of experience with it to provide a well-balanced assessment, and it will be unclear just how the performance statistics were derived. There is no substitute for running various performance tests to generate benchmarking data. The technical design team cannot be shy about changing the component, or the technical architecture design itself, if certain components prove to be incapable of performing as desired.

The database package must be installed on the hardware and operating system platform on which it is intended to run. Any packaged application software that will be used must be installed. Then test data needs to be loaded into the database and performance trials conducted to test the operation of the whole architecture. Simple code should be written to pass data from one component to the next, to test out the data interfaces and the speed of LAN, WAN, and Internet connections. By the end of the prototyping activities, the technical architecture must be shown to perform up to the requirements of the new system that it will support. If a prototype cannot be created that performs well, then there is no sense in trying to build the real "production version" of the system using this same technical architecture.

If the decision is made to purchase a packaged software application developed by an outside vendor, it should be brought in and installed in a test environment. Realistic usage scenarios need to be scripted out, including loading the databases used by the package with a sampling of real data. Security concerns about who can and cannot obtain access to the data should be addressed. People who will both use and support the package need to evaluate it by working through the usage scenarios.

Final Planning and Budgeting

Toward the end of the design phase, as the detailed design specifications are produced, everyone involved will have a clear idea of the work they need to do and how long the tasks will take in the "build" phase. The project leader is now able to oversee the creation of a detailed project plan and budget for building the system.

Project teams are assigned responsibility for specific objectives, and the people on these teams can then lay out the sequence of tasks they will perform to achieve each objective assigned to them. Working with project office staff, each team lays out the plan for its work. Each task has time and resource requirements assigned to it so a cost for each task can be calculated. Respect the seven guidelines for running projects that were introduced in Chapter 8.

As the project teams are each creating their specific task plans to achieve the objectives assigned to them, the project leader is combining these plans into the overall project plan. It's smart to segment the "big" project plan by objective. Devote one section of the project plan to each objective. The project leader determines the necessary sequence for achieving the objectives and arranges them in this order on the plan. He or she looks for opportunities to run activities in parallel, which will allow the flexibility to, when one activity is finished, perhaps shift manpower or other resources over to help with another activity that was delayed.

Delays are inevitable on a project. A plan that does not account for them or provide the flexibility needed to effectively respond to them will almost surely have its timetable and budget thrown into unnecessary disarray and confusion.

When the project plans are in place, a detailed project budget can be derived, which is a fine-tuning of the cost/benefit analysis shown in Figure 8.6. If the project tasks have undergone a lot of change in the planning process, the budget should be adjusted accordingly. The bottom line on this budget document should be the final, accurate cost of the project.

The Decision to Proceed . . . or Not

One of the things that may be hardest for beginning management students to grasp is that, even after all the intensive designing, planning, and budgeting activities described in Chapters 8 and 9, the project team is still not absolutely certain the project will move forward. Theirs is still an exercise in patience. In most companies (and/or supply chains), the detailed system design and prototype, as well as the updated project plan and budget, are presented to a senior management steering committee or an executive sponsor of the project. If there are doubts about the viability of the project or if the revised budget has gotten too big, some tough decisions must be made—like whether to reduce the scope of the project or cancel it altogether.

At this point, the company or partners have already spent between 20 and 40 percent of the total project cost! How could they not move forward? In most businesses this is one of the accepted norms of research and development. Sometimes it is best to drop a project rather than commit further funds to it if its prospects for success do not seem clear.

And why now? Because once the project moves into the “build” phase, it will be very hard to make significant design changes without negative impact on the budget, the completion date, and the organization of the project. That commitment of effort and funds must not be made if there are continuing questions and changes in the basic design of the system that will throw the whole project into confusion.

How does this happen? We may be dealing with computer systems here, but we are also dealing with human nature. It is all too common for companies to run the design phase as a poorly defined research project. Much time is spent in detailed analysis of what already exists, but only sketchy design work is done on the specifics of the new system. Debates ensue about various aspects of the system design, but nobody sees them through to a clear conclusion. Excitement about the idea obscures the tough questions. Preliminary design flaws are brushed aside, and people figure they’ll fix them in the “build” phase. Senior managers who conceived of the project in the “define” phase may still be enthusiastic about it, but the folks who actually have to build the system have their doubts. Rumors and grumbling begin and disillusionment sets in.

The design phase is the opportunity for a company to reduce the risk on a project before committing large amounts of time and money to it. The more detailed the design specifications, the better the chances for building or refurbishing the system on time and on budget. The broader the understanding of and support for the system among both business and technical people, the greater the likelihood that it will be used effectively and produce the desired results.

So at the end of the design phase, the executive sponsor and the project leader must pause and take stock of the project. Is there an air of understanding and confidence among the team members? Are they “good to go”? The answers will be evident, for those who want to hear them.

If the design phase has not produced clear design specifications, if the strategic design guidelines have been ignored, and if people on the project team seem to have doubts about their abilities or the eventual outcome, then there will be no success. The project leader can only fail in these situations, no matter how sincere or heroic the use of leadership skills.

The biggest eye-opener here is the 75 percent failure rate on IT system development projects. It’s not so high because we are incapable as people. It is because we make fundamental mistakes in our system designs and our plans for building them.

The Build Phase

This is the “go for it!” phase. If the project has gotten this far, the project leader should be sure everyone sticks to the aim and resists the temptation to change direction. There is a time frame that, at this point, should be realistic. Activity must be tightly focused on the completion of specific sequences of tasks. This is the step where good design and planning pay off handsomely.

In this phase, the project effort really ramps up. The full complement of people is brought on to fill out the project teams. Because of this, the weekly cost or “burn rate” on the project also rises significantly in the “build” phase. So, unlike the previous two phases, the cost of false starts and wrong turns now adds up very quickly.

The Project Office

The project leader and the team leaders are fully engaged in leading the project, and they must depend on the office or administrative staff to keep up on the planning and budgeting updates as milestones are approached, reached, and passed. It is startling how quickly the plan and budget can become outdated as coordinating and decision-making tools, and keeping them up-to-date to reflect reality is in itself a full-time job. The plan is the map of where the project is going and the progress made to date.

That said, there is a pervasive tendency for people to hide bad news such as delays and cost overruns. Unless the project leader takes active steps to counter this tendency, the project will run into trouble. People need to see that they will not be penalized for reporting bad news. On the contrary, they must be shown that by reporting delays and potential cost overruns as soon as they detect them, they can improve their chances of success. Early reporting gives everyone more time to respond effectively.

The team members must also understand that the project office staff is there to help them keep track of what is really going on and make timely decisions based on realistic data. If bad news is hidden on a project, when the truth finally does come out, there is usually very little (if any) time to respond effectively to the situation.

System Test and Rollout

The first step in taking a system from development into production is to perform a system test with all the system components in place. If competent load and volume testing was done on the system prototype during the design phase, then there should be no surprises about whether the system technically works and can handle the workload expected of it. The purpose of system testing is to work through a series of test scripts that subject the system to the kinds of uses it is designed for and to exercise various features and logic of the system. Some embarrassing logic mistakes may well emerge during system test, and this is okay. That's what a test is for, to flush out and fix these kinds of errors before the system goes into **beta test**.

The beta test is the second system test—this time with a pilot group of business users, typically a few long-term customers or other members of the supply chain. The pilot group should have been involved in some way in the design phase of the project, so they already have a basic understanding and acceptance of the need for the new system and are ready to help the test team work out the bugs. Many minor adjustments are made, both to system architecture and the user interface, during the beta test, as the operating parameters are fine-tuned to get the best response time and stability from the system. The people who designed the user interface spend time with the pilot group of business users, listening to their ideas for improvements.

The business user who works with a system day in and day out will have a different perspective on the system's features than the people who designed and built the system. Minor inconveniences in the system's operation can become major irritants to the people who have to use the system on a daily basis. No matter how "minor," these inconveniences should be fixed.

As the pilot group makes suggestions for adjustments, some will emerge as advocates for the system. They feel a personal connection to the success of the system, especially if it has taken on a "look and feel" that was influenced by their suggestions. These are the people who will sell the benefits of the system to the rest of their company, and who may even be the ones who train their coworkers in the use of the system.

One of the deliverables that is key to the success of a system in the field is documentation. The people who operate a system need to know how to boot the system up, bring it down, and do performance tuning, troubleshooting, and operating maintenance. The project team is responsible for documenting these procedures clearly and understandably, and like the system itself, these documents will undergo a lot of fine-tuning.

When the system first goes into production, the rollout for a "big" system (one that affects more than one area of the business, or many people in a single area) may last a while, from six months to a year. There is not a lot of new development going on during this time, but a steady stream of minor enhancements and "bug fixes." The project team can be slimmed down, but the project leader needs to stay involved during this time to facilitate the rollout and respond quickly if some unexpected obstacle arises.

Supply Chain Skills—Creating a Proprietary System Continued

Network Services Company built the first version of its "Web-enabled supply chain" in 2000. The system has been enhanced and new features added since then as market needs have evolved. Through the strategic planning process, Network saw the biggest benefit to be gained was from using Internet technology to electronically connect all the different computer systems of its member companies. This would allow passing files such as purchase orders, invoices, and product masters quickly and accurately among customers, members, and manufacturers. With this type of system, Network would be able to plug into whatever electronic trading networks were evolving in the markets it served. The system to connect Network with its members, customers, and manufacturers was named NetLink-NSC.™

The second opportunity the company saw and acted on was to make sales history data available via its Web site. Since Network is the hub of the NetLink-NSC™ system, a handy by-product is that Network gathers a lot of valuable information that can be used by customers, members, and manufacturers. The company decided to build a data warehouse that could be accessed through report generation screens on its Web site.

Many of Network's customers already had, or were building, their own order entry systems. They wanted to send the company purchase orders directly from their systems using either EDI or the Internet. Network decided to lease the use of a Web-based order entry package from a supply chain service provider named Tibersoft.

As Network's CIO, I was given the responsibility for building this e-business systems infrastructure. It was very important to develop it quickly and cost-effectively, so I used the time-boxing guidelines suggested in the define-design-build process. A team of business and technical people was assembled; they identified the company's most pressing business issues and ideas for how technology could help. These ideas were translated into a conceptual system design (shown as Figure 8-4 in Chapter 8).Within six weeks, the define phase was complete. The conceptual design and a proposed budget to develop the infrastructure were presented to the board of directors. They gave their approval to proceed with development.

Even before the define phase was finished, work began on a design-and-build sequence to produce a beta test version of the data warehouse that stored sales history data. This was completed quickly and provided the business with a valuable tool as well as proof that the technical architecture was viable. It also gave everyone a clear indication that the system development effort was off to a good start and would live up to expectations.

A select group of consultants was brought in to work with company IT staff. Four project teams were created—one team to design and build each of the four components of the e-business infrastructure. Two of the project team leaders were from the Network IT staff, and two were consultants. There was also a two-person project office team headed by one of Network's IT managers.

The hardware and software components chosen for the systems architecture were assembled and tested by the teams that would use them. Data was passed between the components to make sure they could work together. Response times were tested under different data volumes to verify that the system could handle the expected amounts of data. When testing was finished, we had a solid design and there wasn't any talk of, "We'll figure this out after we get into the build phase..."

As each team finished the design phase of its work, it was well positioned to launch into the build phase. Each team had a clear set of design specifications and had been able to test and verify that the hardware and software would meet their expectations. This allowed each team leader to work confidently with the project office manager to create very accurate project plans and budgets for the build phase.

The first versions of all the system components were finished within nine months and demonstrated to member companies and suppliers at the Network annual trade show. The systems were well received by Network members, and some members asked if they could use these systems to support their own local business as well as to handle national account customers. So a set of enhancements to NetLink-NSC™ and the data warehouse were quickly designed and built to let members use them for their local customers. In the last three months of 2000, the project team worked with Network members to roll out version 1.0 of these systems.

Going into 2001, Network assessed its business situation and the market conditions, and defined a set of major enhancements to add to the systems it had just rolled out. These enhancements were again designed and built within nine months and were demonstrated in the fall at the annual trade show. Every Friday afternoon, the team leaders and I met to discuss the project. We spent several hours and reviewed the progress and the issues as each team encountered them. The project office staff provided accurate and updated plans and budgets for these meetings. We could all see the most current time and cost estimates to finish each objective. We could see if work on an objective threatened to push beyond its time box or if it was likely to overrun its budget.

The development sequence was focused and tightly time-boxed. Work ran in parallel during the design and build phases, requiring good planning and coordination. Version 1.0 of the e-business systems infrastructure was created in nine months! Based on positive reception and feedback from version 1.0, enhancements for version 1.1 were created. Further assessment of business needs led to the definition of the next round of major enhancements that created version 2.0 of the e-business infrastructure . . . and so on.

Source: This was written by coauthor Michael Hugos about his own computer design experience as CIO of Network Services Company, Mt. Prospect, Illinois.

Chapter Summary

Designing a computer system for a company to enable data synchronization with others in the supply chain is one of the most collaborative and intensive efforts employees and managers will undertake. After the basic planning process (which was covered in Chapter 8), the designing and building of the system can be broken down into these steps:

"Design" Deliverables

1. A detailed design for the business process flow of the system. Also, agreement among the people who will have to work with the system that it will meet the performance criteria expected of it.

2. A system prototype that specifies both the technical architecture and user interface. The technical architecture must be capable of handling the projected data volumes and user demands that are expected. There must be a complete set of screen layouts, report formats, and specifications for all aspects of the user interface.

3. A detailed project plan and budget that accurately reflects the time, cost, and resources needed to build the system.

"Build" Deliverables

1. A working system that matches the design specifications and meets performance criteria. The building of the system should be scheduled so something of value is delivered to the business every 30 to 90 days. This means certain pieces of the system must be finished and put into use before the entire system is completed.

2. A complete and updated set of technical design documents. The design documentation is analogous to the floor plans and infrastructure plans of a building.

3. A complete set of operating instructions. The people who operate and maintain a system are different from the people who build systems, and they require clear, concise instructions about everything from how to boot up the system to troubleshooting and operating maintenance.

No one in business today works in a vacuum. The design or upgrading of any company's computer system requires extensive input from other supply chain members to ensure that the systems can work together, as well as the companies themselves manage to do so. In retail, business would be impossible without this type of coordination.

Discussion Questions

1. List the basic technology-related information you believe a retail employee should be familiar with in order to advance in a midsize to large store chain today.

2. Explain the difference between the types of process sketches required in Chapter 8 (the plan phase) and the ones required in Chapter 9 (the design phase).

3. Why do costs for each task in a system development project have to be calculated? How would you counter a staff member's assertion that this is "micromanaging"?

4. Describe one "bug" or shortcoming of a computer system you now work with that drives you crazy—you'd change it if you could! Knowing now how a system is designed, why do you think the project team didn't fix it?

5. What should a project manager do when, in beta testing of a system, the critics outnumber the advocates?

The Incredible Journey Continues

OCTOBER 1—PRODUCT MEETS STAFF

In the wee hours of the morning, a CVS truck appears at McGee's store with his order. McGee unloads the tote in the back room and, voilà, there is the bottle of Listerine that originated in the Australian eucalyptus grove back in May. He displays it on the shelf according to the dictates of the store presentation managers at Woonsocket who determine ideal store layouts and product placement.

OCTOBER 2—THE PURCHASE

Now for the last step of our journey—the customer. A man walks into McGee's store to pick up aftershave and a box of chocolates for the date he's planned for the evening. On impulse, he decides to buy a bottle of Listerine. His purchase is recorded by the cash register and sent to E3 and Category Map that night for future forecasting. In a few days, McGee will place an order for a replacement bottle of Listerine. And so begins the cycle again.

How do store managers know how much product to stock?

CVS management frowns upon stores keeping extra inventory in the back room because it ties up capital. So instead of having each store manager project demand, once a year, category managers at headquarters analyze annual point-of-sale data generated in each store and calculate the weekly sales rate for every product by store.Then they multiply that number by 4 to get the minimum shelf quantity (MSQ). In other words, at any given time, CVS carries four weeks' worth of supplies on the shelves. At Mike McGee's store in Framingham, Massachusetts, the MSQ for the 500-milliliter bottle of the best-selling Listerine flavor, Cool Mint, is 20 because the store sells an average of five bottles per week.