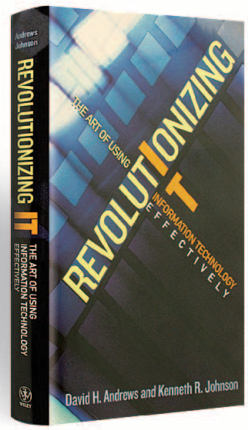
SOUNDVIEW Executive Book Summaries®

FILE: HANDS-ON MANAGEMENT



By David H. Andrews and Kenneth R. Johnson

CONTENTS

The Art of Using Information Technology Effectively

REVOLUTIONIZING IT

THE SUMMARY IN BRIEF

Why is it so hard to gain concrete business benefits from Information Technology (IT)? Why do so many IT projects overrun budgets, miss schedules and fail to meet expectations? How can your organization ensure the success of its next IT undertaking? Revolutionizing IT *provides a simple yet profound set of management principles that will completely change the way you think about the management of technology. Most managers instinctively know that IT projects shouldn't be as complex as they always seem to become. This summary will help guide you through your next IT project, showing you how to* ***R****evolutionize* ***IT E****ffectively — the* ***RITE*** *way.*

What You'll Learn In This Summary

✓ **How to let time determine the scope of your project.** You will see why setting a time limit early will save the project from becoming bloated and irrelevant.

✓ **How to succeed through continual incremental improvements by letting projects mimic evolution.** Change is evolutionary, and trying to make too large a leap into the future at once is doomed to failure. Improving parts of the system quickly is often better than overhauling the entire system at once.

✓ **How to recycle proven concepts instead of reinventing them.** You will see why borrowing from others who have successfully solved a problem is often better than developing the same or another solution yourself.

✓ **How to demand progress, not perfection.** You will learn why it's best to expect progress but not perfection. There are no perfect solutions but all good solutions bring progress.

✓ **Why to make end users fully accountable.** It does no good to make consultants or the IT staff accountable for the success of a project. It is the end users who should be responsible since it is their problem that is presumably being solved by the project.

Published by Soundview Executive Book Summaries, P.O. Box 1053, Concordville, Pennsylvania 19331 USA

REVOLUTIONIZING IT

by David H. Andrews and Kenneth R. Johnson

— THE *COMPLETE* SUMMARY

The Nature of Projects

The pace of technology-driven change is not likely to slow. Each innovation cycle opens up opportunities for improving existing products or services or inventing new ones. Putting technology to use is an important management challenge. Managers must pay attention to information technology. Web sites must be created and updated, data must be collected and analyzed, and software must be built linking it all together.

Unfortunately, advancing technology is creating opportunities at a faster rate than businesses can recognize and exploit them. This is because there is a large gap between recognizing an opportunity and taking advantage of it. The bridge across that gap is a project. And that project is most likely to succeed if you take the **RITE** (**R**evolutionizing **IT E**ffectively) Approach. The **RITE** Approach is a collection of observations and principles designed to provide guidelines to those who make decisions regarding IT projects. Some of the most powerful principles of the RITE Approach are:

1. Let time determine the scope of your project. Setting a time limit early on will save the project from becoming bloated and irrelevant.

2. Succeed through continual incremental improvements by letting projects mimic evolution. Change is evolutionary, and trying to make too large a leap into the future at once is doomed to failure. Improving parts of the system quickly is often better than overhauling the entire system at once.

3. Recycle proven concepts instead of reinventing them. Borrowing from others who have successfully solved a problem is often better than developing the same or another solution yourself.

4. Demand progress, not perfection. There are no perfect solutions, but all good solutions bring progress.

5. Make end users fully accountable. It does no good to make consultants or the IT staff accountable for the success of a project. It is the end users who should be responsible since it is their problem that is presumably being solved by the project.

Terminology

In this summary, the following definitions will be used to describe the work of information technology:

- *IT projects* are those projects that are launched and monitored through a formal process, involve a number of people over months or longer, use software to play a major role in solving the problem, and have as the goal putting a working improvement in place. The end result is a change in the *business process.*
- *System maintenance* involves minor changes or enhancements to existing software applications carried out by a small group of people over weeks rather than months.
- *Projects* are involved when complex systems are created, such as an airline reservation system. These normally evolve over time and involve individual efforts that create specific workable parts of the ultimate system.
- *Products* are the systems created through a series of related projects.
- *Software releases* are the software created during a project. To test whether the software will do what is intended, *prototype software* may be created. Incomplete software given to users for testing is called a *software increment.*
- *Stages* are the efforts taken to evaluate the problem, create the plan or design the solution.
- *Scope* of the project is the determination made as to exactly what the project will accomplish.

The Vital Early Stages

IT projects create value through business process improvements if the result is a permanent improvement

(continued on page 3)

The authors: David H. Andrews has overseen information system development efforts at SmithKline, Citibank and other major corporations. Kenneth R. Johnson has spent more than 30 years overseeing the development of advanced software systems.

For Additional Information on the authors, go to: http://my.summary.com

Published by Soundview Executive Book Summaries (ISSN 0747-2196), P.O. Box 1053, Concordville, PA 19331 USA, a division of Concentrated Knowledge Corporation. Published monthly. Subscriptions: $195 per year in U.S., Canada & Mexico, and $275 to all other countries. Periodicals postage paid at Concordville, PA and additional offices.

Postmaster: Send address changes to Soundview, P.O. Box 1053, Concordville, PA 19331. Copyright © 2003 by Soundview Executive Book Summaries.

Available formats: Summaries are available in print, audio and electronic formats. To subscribe, call us at 1-800-521-1227 (1-610-558-9495 outside U.S. & Canada), or order on the Internet at www.summary.com. Multiple-subscription discounts and Corporate Site Licenses are also available.

The Nature of Projects

(continued from page 2)

in the way a function is performed. Keep that goal in mind when evaluating project proposals. Ask yourself:

- Exactly how will the project change the way the organization functions?
- What benefits will the change bring?
- Are other options available to achieve the same benefit?

Many projects fail because they don't answer these fundamental questions. If you look back at a failed project, you will likely see the signs early on. The fate of a project can be determined early on. How an organization handles the earliest stages has enormous impact on the result. Many projects are doomed to failure before they have begun because of attitudes, assumptions and approached that are seriously flawed. These include:

- Vague or contradictory mandates.
- Lack of people with the right skills and knowledge.
- Inability to get vital information on time.
- Inability of the organization to absorb change at the rate needed.
- The problem changes before the solution can be implemented.
- Inability to provide management with the precise plan it desires.

Why Scope Takes Off

Once a project is initiated, predictable things happen. For one, the end users have two conflicting forces pulling at them. They want to ask for everything they could possibly want but they also want to limit the disruption they will face. For another, the very people who best understand exactly how things work now are the ones who have the least time to spare. The temptation is also great to dump troublesome individuals into the project to get them out of the way. Finally, the technical people assigned to the project typically want to make use of their skills and will create more ambitious changes than may be necessary.

All these factors combine to create growth in the scope of the project. Those representing the end user community are encouraged to define their needs broadly. They fear criticism for leaving something important out of the specifications, leading to the creation of highly ambitious system requirements. The IT professionals involved in the projects are supposed to translate these ambitious goals into a system that solves the problem. Couple that with the natural IT desire for challenge, and you have an upward spiral of scope as the user's desire for functionality feeds the IT professional's desire for challenge and excitement.

The Classic View of Project Management Theory

The first generation of formal IT methodologies was viewed as a series of logical steps carried out one after another. Projects were born when an organization identified a problem or saw an opportunity. People were assigned to define what the project was supposed to accomplish. A project plan was developed and presented to management for approval with an estimate of the costs, time and resources required.

The project plan then served as the starting point. IT professionals would interview people familiar with the affected functions. Their requirements formed the basis for custom-designed software or selection of a software package. Once approved, the design was turned into a working system and then implemented. This orderly process was far more useful than simply putting a few programmers together and having them knock out the code needed to make computers do what was needed.

The expanded scope then runs headlong into another reality — limited resources. Competition is stronger, margins tighter, buyers more demanding, staffs smaller and market conditions more unstable. Most organizations today have precious little capacity to undertake efforts that aren't directly related to the daily operation of the enterprise. Problems severe enough to demand immediate attention are arising at an ever increasing rate. Undertaking an IT project is often the correct response, but finding the resources is a challenge. It is therefore essential that IT projects undertaken be assured the best possible chance of success. There is no time or money for wasted efforts.

The Anatomy of a Project

Before you embark on a major IT project, you must learn to guard against five unrealistic assumptions. These are:

1. The environment will remain stable during the project. Unfortunately, long periods of stability are a thing of the past. Change is a constant. The organization will experience management turnover, product introduction, new competition, government regulation, lawsuits, natural disasters and even terrorism. Technology will change. The longer the project, the more likely instability will enter the picture.

2. End users can define in advance exactly what will be needed. This, too, is untrue. The first attempt to

(continued on page 4)

Did Lewis and Clark Succeed or Fail?

One of the most important projects in U.S. history was the 1803 effort by Meriwether Lewis and William Clark to find a Northwest Passage. Their project had a profound effect on the United States we know today. But using the standards some executives would apply today to projects, the expedition was a complete failure. It took longer than expected, cost more than planned, and failed to meet its primary objective — to discover a waterway that led to the Pacific Ocean.

Before Lewis and Clark started their project, they only knew vaguely what lay ahead. They planned carefully but couldn't be expected to know what they would face. Yet they gathered valuable information that accelerated westward expansion. In IT, some projects need to be organized like the Lewis and Clark expedition — organized as voyages of discovery and not as precisely planned trips to a known destination.

The Nature of Projects

(continued from page 3)

design anything complex is rarely very good, even when done by experts. And end users aren't experts with the ability to design complex systems. Design engineers don't ask users to create and approve the technical specifications for cars and airplanes. Prototypes are built and tested instead. The people currently using the system may not be capable of defining appropriate requirements for a better business process.

3. Complex problems can be solved completely on the first attempt. It's not that simple. Most plans are imperfect.

4. Requirements can be precisely defined before packaged software is selected. This, too, is untrue. This expectation arose in a time when most software was custom-designed by in-house programmers. Better to select a package faster than to spend time and money trying to define exactly what the requirements — a futile task at this point.

5. Users will cheerfully accept changes in their work environment. This, too, isn't the case. People fear and resist change.

As the size and complexity of a project increase, a number of bad things can happen. Greater scope means more time, which increases the chances that environmental change will disrupt the effort, maybe even making it irrelevant. Costs also increase as complexity grows. And as time to complete the project increases, the people who will benefit find it difficult to put energy into the project because they don't see any immediate benefits. Organizational resistance increases at the same time as resources are spent to provide interim fixes to the problems the project is expected to fix. By now it should be obvious that the first priority of any project team must be to put limits on what the team will attempt to accomplish. ■

For Additional Information on a project that failed because it didn't take the RITE Approach, go to: http://my.summary.com

Elements of Successful Projects

Not all information technology projects fail. What separates failed projects from successful ones are six things. These are:

Strong leadership. As the talent, motivation and experience of project leaders increase, so does the chance of success.

Time control. Time is a project's worst enemy. Successful projects are those that are manageable but subject to time pressure.

Resource limitation. Successful projects limit the number of people involved.

Scope control. Limit the scope, and chances of success increase.

Staged evolution. Successful projects break what needs to be done into manageable parts. They follow an evolutionary path.

Concept recycling. Successful projects invent as little as possible. A highly innovative system can be constructed from proven components where a small number of totally new ideas are included. Successful projects borrow ideas and building blocks from any legally available source.

The RITE Approach

Changing how IT projects are viewed can help improve the odds of success. Consider this: If the definition of a successful marriage was that the couple never had a disagreement, then the odds of success would be close to zero. It is equally absurd to define the success of projects by how closely they adhere to plans created in the early stages. On the other hand, it would be absurd to define a successful marriage as one in which neither spouse killed the other. Similarly, a project is hardly a success if its barely breathing carcass is dragged across the finish line long after the reason it was initiated has been forgotten.

Managers should view projects as chess games. Good

(continued on page 5)

Elements of Successful Projects

(continued from page 4)

chess players start with a general strategy and a detailed plan for the early stages of the game. After a few moves, the plan needs to be constantly revised in reaction to the opponent's moves. The strategy remains the same, but the moves change. IT projects likewise need to be constantly adapted to changing conditions while remaining faithful to the overall goals. Success at IT projects and chess is defined by the end result and not by the degree to which the original plan was followed.

Organizations are in much better position to succeed when realistic assumptions underlie the mental picture of success. These realistic assumptions include:

- It's impractical to attempt to understand every aspect of a problem.
- Complex problems are made up of an intricate web of smaller ones.
- The nature of the problem will change over time.
- The more time passes, the more change will occur.
- Any complex design will be imperfect.
- There will not be enough human talent available to create the optimum solution.

Following Mother Nature's Lead

The foundation for a better approach to projects is ***iterative development***. Mother Nature has been using this design philosophy for billions of years. Evolution has been shown to be a simple and powerful way to develop systems of incredible elegance. The logic of evolution is to create something new, try it in the field, introduce variations, incorporate those that are successful into the design, and reject what doesn't work.

The essence of evolution is incremental improvement. Evolution teaches us that iteration is the best way to develop a great design. Innovations are tested incrementally under real-life conditions, and those that represent a true improvement become part of the design. An iterative and incremental approach to projects incorporates this concept by encouraging the testing of new concepts, one at a time, under real-life conditions and keeping those that actually work.

The way that projects proceed under the **RITE** Approach includes the following steps:

- **Create a long-term vision.** But don't spend a lot of time making it perfect. Expect that no vision is flawless.
- **Evaluate the environment.** Estimate how much time will pass before the next major environmental change will occur.
- **Break down the problem.** What are the most important goals of the effort? Are there opportunities to quickly make high-impact, near-term improvements?
- **Plan the first step.**
- **Iterate activities as needed.** Build and test anything new in manageable increments.
- **Make it happen!** The first operational deployment needs to arrive quickly and create positive momentum.
- **Evaluate and adjust.** Once an increment of the ultimate solution is operational, re-evaluate.
- **Update the vision.** See how it stacks up to what has happened and update as needed.
- **Plan the next phase.**

Using the **RITE** Approach, the first step is to assign a task force to examine the issue and formulate a high-level view of the issue and how it might be solved. The **RITE** Approach is based on the principle that the best plans are made rapidly by a small group of people. Less than five is ideal, fewer than 10 is essential. They should understand that plans will be imperfect regardless of how much time or resources are invested. The task force should be encouraged to come up with solutions that can be put into place immediately. They need to know that a project isn't the only possible outcome. If they do conclude a project is needed, they must first develop a broad view of the ultimate solution — the long-term vision. Then they decide what the first steps toward that vision will be.

Control Project Scope and Establish Accountability

The propensity for projects to grow in scope (see page 3) is a major cause of failure. There is nothing wrong with systems becoming highly complex as long as it occurs in stages. What needs to be resisted is the desire to strive for a high level of sophistication all at once.

Let time determine the project's scope. The problems and opportunities that need solving refuse to remain

(continued on page 6)

Why the Largest IT Project Ever Undertaken Succeeded

During the late 1990s, the IT industry faced its greatest challenge. By the end of 1999, software that had been developed over 30 years had to be checked to make sure it wouldn't fail when the millennium dawned. But the so-called Y2K problem turned out to be a non-event. Only a handful of applications failed, and none affected anyone critically. The reason so few failures occurred was that the IT community knew the consequences of not completing its projects on time. The Y2K problem was unique because the end date was fixed and non-negotiable, the definition of success was clear and never changed, and the consequences of failure were dire. There was no scope creep, no negotiation with end-users about specifications.

Elements of Successful Projects

(continued from page 5)

constant while we fix them. Economic conditions change, customer preferences evolve and competitors disrupt the market. None of this can be controlled or even accurately predicted. The only thing you can do is to limit the damage by shortening the time between concept and realization.

Let resource availability drive project scope. Just as time is an enemy, so is lack of resources. Instead of looking for the optimal solution, find the one that fits the resources available. That doesn't mean you should expect too little from your people. Most do their best work when challenged.

Limit the size of the design team. The ideal size of a design team is three to seven people.

You must also gauge the organization's ability to absorb change. Only a few organizations have established a culture where frequent change is the norm and stability is considered boring. In most organizations, change is feared. Today you have no choice but to make constant, constructive change an accepted part of the culture. Just don't do it all at once. The right question is not how much change the organization will comfortably accept but rather how much it will reasonably tolerate. Slowly increasing tolerance for absorbing positive change needs to become a goal for every organization.

Imitate, don't invent. That doesn't mean violate copyrights or patents. It does mean you should take into account best practices and use available technology, such as packaged software, to your benefit.

A Single Point of Accountability

Create a single point of accountability. In an effectively managed entity, all participants have a clear understanding of their role and what is expected of them. Those who are accountable cannot make excuses. If things go well, they are rewarded, if not, they pay a price. It works because it motivates a single person to see that key goals are met even in the face of adversity.

You must select one person to be accountable for the project, and that person should not be someone from IT. Instead, make it someone from within one of the functions what will use or benefit from the resulting system. Those who use systems should be totally accountable for them, including their design, creation, cost, quality, functionality and value. No project should be launched before someone appropriate is chosen to be the project leader.

The project leader isn't expected to personally handle the details of project management. An experienced IT professional should perform that function. The role of the IT leader is to support the project leader in a cost-effective and professional way. In other words, IT professionals are resources under the control of those who will use the resulting systems. Those in IT are accountable for satisfying the needs of those who use their services.

Amazon Gets RITE

Amazon.com is a company that understands incremental improvement. Long-time customers have watched the Web site evolve. Something new and better seems to be added every few weeks. When the site was first launched, it was impressive and professionally designed. By today's standards, though, it was primitive. But slowly improvements have been added, each one adding to the customer's experience. The Amazon lesson is simple: Rapid evolution through a never-ending flow of incremental improvements is superior to less frequent attempts at great leaps forward.

Using Packaged Software

In the early days of computing, applications were usually homegrown. Now, most are packaged software; IT projects revolve around the selection, customization, deployment and integration of packaged software. Packages are used because they reduce the time spent trying to conceive of new ways to perform necessary functions. They also reduce the risk that custom designed software carries — the risk it will fail in practice.

The traditional way to select a software package was to develop system requirements and create a Request for Proposal (RFP) documenting those requirements. When responses came back, the organization picked the one that seemed the best match and created enhancements to overcome any package limitations. That may seem logical, but it isn't. There are several traps in the traditional approach, including:

- Asking users to articulate what they want before they have seen what is available.
- Believing that the RFP really represents what's needed.
- Encouraging modifications that may turn out to be unnecessary.
- Taking a long time to reach a decision.

A better approach is to adopt a new attitude about packages. A package contains the combined wisdom of many other organizations. Their collective wisdom is likely to be better than whatever a single organization can develop. Good packages have used an evolutionary process to arrive at their current state.

If many organizations similar to yours use a particular package, it is likely that it will be a good fit. In fact,

(continued on page 7)

Elements of Successful Projects

(continued from page 6)

there should be a good reason to *not* adopt the most widely used approach before you reject it.

The best approach is to visit one or two sites similar to yours to see the software in action before the vendor makes a formal presentation. Once you select a package, encourage its use exactly as intended. Invest heavily in training, especially after the package is online. It will take time before the software's full capacity will be appreciated. ■

The Project Balancing Act

To summarize the issues involved, those approving projects must strike a balance between eight critical factors or they risk being left behind. The eight factors are:

1. Scope — what the project will accomplish.

2. Benefits — the net value the project creates.

3. Time — the period between approval and delivery of benefits.

4. Disruption — the negative impact of the change on the organization.

5. Cost — the profit and cash impact.

6. Risk — the probability that things will not work as expected.

7. Resources — the talent and skills that will need to be dedicated to the effort.

8. Quality — the reliability, availability, performance and user satisfaction of the solution.

Trade-Offs Are Inevitable

Trade-offs always need to be made, so don't try to avoid them entirely. The best way to balance the critical factors is to ask the right questions before the project launches. These include determining what can be spent, who can spare time on the effort, whether outside help is needed, whether similar projects elsewhere were successful, where the next major business disruption is likely to come from, and what other risks can be foreseen. The analysis shouldn't take long. Sometimes all that's needed is an open discussion about them at the very beginning of the planning process.

One of the most difficult trade-offs is time. The best way to deal with how soon a project must be delivered is to set a firm completion date and let that date determine the scope of the project. The most successful organizations are those that consistently produce a steady stream of incremental improvements. The cumulative effect of projects with tight deadlines that deliver incremental improvements is excellence. The first step toward becoming such an organization is to adopt the view that steady, evolutionary progress achieved through an endless succession of small improvement projects must become a permanent part of the organizational culture.

Don't fall into the trap of over-planning. Some planning is a good thing, but too much can destroy the project. Don't let yourself get bogged down to the point where you never get to the doing stage. The best plans are created in a limited time by a small group of people. Never spend more than 90 days on planning. Usually 30 days is enough. Limit the people involved. More than 10 is usually fatal and less than five is ideal. The mandate must be to create a project plan that can be fully implemented in the time allowed. Management's expectation must be progress, not perfection.

You must always keep in mind that the single valid reason for undertaking an IT project is to create benefits. Projects should only be approved after a consensus is reached that anticipated benefits are well-understood and will work.

Remember that you also will be balancing risk as you develop the project. Risk is a very hard thing to quantify, much less control. One of the most effective techniques for managing risk is obvious but infrequently used: Do the hardest thing first.

Using Outsiders Wisely

There was a time when car owners could tune their own engines and do much of the necessary maintenance themselves. But cars have become too sophisticated for that. There was also a time when most organizations handled important IT projects on their own. That era has also ended due to the complexity of the software that most projects depend on. More and more, organizations rely on outside help.

There are many reasons to seek outside help. It nets you an extra pair of hands without a long-term commitment. An outsider can provide skills not available within the company. Experts who have helped other organizations may be able to offer insight into business processes that can work. But these benefits come at a cost. Consultants are also expensive, learn at your expense, may not appreciate your culture, and have a natural bias toward larger projects.

If you must use consultants, the best way to get value is to maximize the talent of the people chosen for your project while minimizing their number. If you do decide to use consultants, hire them the RITE way. You want to maintain control over the project. Shifting responsibility to consultants is tempting, but does nothing positive for you. If consultants accept all risk of failure, they most likely will adopt an approach that bloats the project. The consulting agreement should provide that:

(continued on page 8)

The Project Balancing Act

(continued from page 7)

- Both parties acknowledge that projects are dynamic and subject to change.
- You, the client, will provide a leader responsible for making decisions and trade-offs.
- The service provider will help manage the project but follows directions from the client leader.
- The schedule and resources are fixed, and the scope is adjusted to meet them.
- The client agrees that the service provider is entitled to make a profit.
- The client expects the people assigned to be capable and to work in their interest.
- Service providers aren't responsible for the ultimate cost of the project.
- The project team consists of a limited number of highly experienced and capable people.
- Service providers are entitled to charge high rates for the most capable people.
- The engagement forms the foundation for a long-term relationship.
- The service provider can expect to participate in follow-up efforts.
- An appropriate number of people are assigned with high average skill levels.

Using this approach, clients make key decisions and trade-offs and assume associated risks. Service providers agree to participate in smaller projects than they would prefer. Rates are fair to both parties, there is an opportunity to build a profitable long-term relationship, and the service provider may become a trusted advisor. ■

Project Management

After the scope of a project has been tightly controlled and undivided accountability has been assigned to a leader from the user community, don't relax. As the project unfolds, consider these problems:

- Project schedules are inherently inaccurate. One reason is that schedules are typically developed before the information needed to do so accurately is available.
- Problems tend to surface late.
- Bad news travels slowly.

One way to mitigate the schedule problem is to create a two-tier schedule. The first is highly optimistic and the second contains more buffer time but still meets organizational requirements. If the second deadline is met, the program is a success and whatever rewards you have promised are earned. But additional rewards are available for projects that meet the earlier, more ambitious schedule.

More Speed, Scotty

In the original *Star Trek* television series, Captain Kirk would regularly ask Scotty, the chief engineer, to give him more power from the starship Enterprise's engines. In almost every case, the response involved a trade-off for the captain to consider: "I can give you more speed, Captain, but if I push her too hard the whole thing is going to blow."

Captains of industry face the same dilemma every time a project is presented for their approval. They can blow the whole thing if more speed is requested without making adjustments elsewhere. There is rarely any debate that fast is better than slow. What is hard is making the necessary trade-off between speed and the other factors.

Building Software Yourself

Sometimes you can't rely on the principle, "Imitate, don't invent." If your project requires the development of custom software, you need to recognize right away that the project will be tough. But more and more, companies need specialized software. Consider the example of online auction site eBay. In that case, software is the business.

> ***Sometimes you can't rely on the principle, "Imitate, don't invent."***

Creating custom software is difficult. It isn't a game for amateurs. The specifics are left to the IT professionals, but management's role includes:

- Approving plans.
- Measuring the fit of the software as it's being developed to see if it will do what management expects.
- Overseeing project reviews to resolve problems early.
- Setting an example and selling peers on the value of improvements.
- Ferreting out errors and misunderstandings.

In other words, good development managers need the technical mind of Bill Gates, the organizational skill of Caesar Augustus and the patience of Gandhi.

Once actual software development is underway, a disciplined methodology is essential. There are many methodologies from which to choose, but the choice must be sure to let the project include the stages of planning, design, coding, testing, delivery and service. ■

For Additional Information on a company that successfully used the RITE Approach, go to: http://my.summary.com