Good Planning Means Success in Implementing RAD

New development methodologies like rapid applications development (RAD) require innovative management techniques and the use of integrated computer-aided software engineering tools.

JAMES MARTIN

Most information-systems (IS) executives are under pressure to keep their core applications running and maintain existing systems that are critical to the business. This tends to consume so much of their time and energy that they can barely consider a major change in the IS development culture. They don't have any spare staff with which to take risks.

Because IS has slow development cycles and because maintenance of the old systems is difficult, there are large application backlogs. An IS department with backlogs like those shown in the figure (at right) severely damages the ability of a corporation to adopt new computerized business procedures. To compete effectively, it's critical that a RAD life cycle be introduced.

Rather than attempt to change the entire IS organization, the best way to introduce RAD techniques is to start small. A small group should be responsible for selecting the tools and life-cycle techniques for fast development. Small, specialized construction teams should be established, and their members should be well-trained to use the new tools and techniques.

A goal should be to make the brightest and most capable developers as skilled as possible with the most powerful tools. These special teams of two, three or four people should be put to work on pilot projects that are non-critical but strategically important to the organization.

The teams should use a computerized methodology customized to the corporation and its chosen tool set. As the team gains experience, the methodology can be enhanced to reflect what the developers think works best.

Some organizations have set up high-productivity groups in an otherwise low-productivity IS department. It's important to measure the results from such a group. The goal is to achieve a productivity and cycle time dramatically better than the rest of IS, while creating systems of high quality. If this is achieved, the high-productivity group should describe their successes and train others to use the same techniques.

It's important to choose members of the development team who think positively and are excited about becoming skilled with advanced tools. People who are firmly wedded to old techniques could subtly sabotage RAD efforts.

Reluctant developers can all too easily demonstrate that new tools don't work. They can scuttle new methodologies by

adopting a "work-to-rule" attitude, in which the new methodology and tools are forced to conform to older, much less productive development rules.

The low-productivity figures with CASE tools in many organizations are the result of developers not wanting to use them. These people are comfortable only with existing techniques and don't want to climb a new learning curve.

A skilled and well-motivated construction team can demonstrate the value of the RAD life cycle and can tune the methodology to make it as effective as possible. Once a team has proven the value of the new development process, more teams can be added.

The problem may be an inappropriate choice of a tool. In this case, it's important to find a tool set that does what is required and switch to it. Managers should seek out integrated computer-aided software engineering (CASE) tools that are capable of generating 100 percent of the code for an application from design specifications.

There are many unfortunate examples of IS organizations becoming committed to an inadequate tool set, such as a non-integrated CASE tool. These tools require hand-generation of the application from design specifications and should be avoided. The switch to a different tool set is painful; but if it's needed, it

training should be oriented to fast development.

The first applications to be developed with the RAD methodology should fit comfortably within the capabilities of the tool set. They should be applications that can be accomplished by one small, skilled construction team in three months or so. Within these constraints, the applications should be appropriately complex—between 500 and 1,000 function points. A purpose of the first project is not only to build systems but to demonstrate that the methodology works well.

To develop systems fast, it's essential to bypass bureaucratic delays. As its victims know, bureaucracy insists on formal procedures. It emphasizes filling in the right forms, going through the right channels and getting the right approvals, rather than finding out how to get the job done as quickly and effectively as possible.

Bureaucracy is the enemy of speed. Indeed, it's the enemy of the three main goals of RAD—higher quality, lower cost and rapid development. Many attempts to create new computer applications have quickly ground to a halt because of career bureaucrats or managers motivated by politics rather than efficiency.

Know Your Enemy

In a healthy competitive corporation, the enemy is on the outside. In a bureaucratic organization, however, the enemy tends to be internal, in other departments.

This is one reason why successful entrepreneurs have difficulty transferring their executive skills to large organizations. They expect loyalty from a staff seeking common goals for the enterprise; instead, they find that predominant goals relate to internal politics. Instead of hostile external forces, they find hostile internal forces.

The operation of small, specialized construction teams, with good measurements, can bypass the bureaucratic tendencies in IS organizations. The executive owner and his IS counterpart must agree that rapid action is essential in the user community as well.

If bureaucracy threatens to interfere with the process, the executive owner must be responsible for cutting the red tape. As in IS, the players from the user community should be determined to take action quickly to make the new system work.

In the final article in this series, I will summarize the factors that lead to success in the introduction of new development life cycles. ∎

The Extent Of Application-Development Backlogs

Use of Rapid Application Development Technology, Tools, Can Cut Backlogs
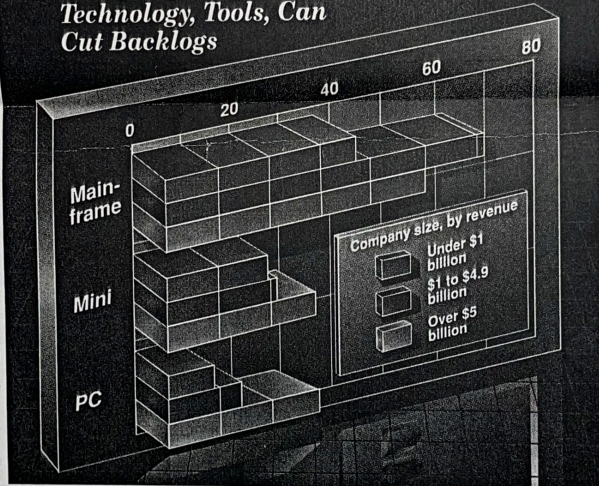
John Avakian

The brightest, most capable developers should be made as skilled as possible with the most powerful tools and put to work on non-critical but strategically important pilot projects.

Productivity, cycle time and quality are measured, and the teams are compared with other developers using older methodologies. It should be made clear that pay raises and promotions relate to measured achievements.

Sometimes a first attempt at RAD fails, despite positive attitudes about the project. When this happens, the cause of failure should be identified, the situation corrected, and another attempt should be made. The developers shouldn't be penalized because of one initial failure; on the contrary, they should be rewarded for a good attempt and encouraged to try again, correcting the problem.

should be done as early as possible, before many people have climbed the learning curve with the inadequate tools.

Sometimes the reason for failure is inadequate training. Many IS organizations have underestimated the training needed for building an integrated CASE environment. These powerful tools are rich in functionality, and the developers have much to learn. Simply providing developers with advanced tool sets and expecting them to become proficient with the new technology in a short period of time is insufficient. The commitment to good training is vital, and the