Draftings In

Volume 7
Number 1 *Draftings In Management*

Article 7

1992

Implementing CASE: More than the Purchase of Software Packages

Claudia Ruggless
University of Northern Iowa

Implementing CASE: More Than The Purchase of Software Packages

Claudia Ruggless

•

Society and businesses depend on computer software. Retail stores use software to track inventory. Banks employ software to balance customer accounts. Since most businesses are so dependent on software, managers become extremely frustrated when software errors occur. Poor quality software result in frequent maintenance. When software developers are forced to spend a large portion of their time on maintenance, they have little time to work on new software projects. The result is a backlog of software to be developed and a reluctance on management's part to generate new software project ideas because of extended development time. Even when systems are functioning well it can be equally frustrating for managers when software is continually updated—just as they are becoming familiar with it.

Computer Aided Software Engineering (CASE) is one of today's hottest new technologies. It has the potential to solve many software problems. In general, use of CASE tools leads to higher quality computer systems with fewer errors. This means less management frustration with errors and fewer updated versions. Clearly, relatively error-free programs provide a competitive advantage over competitors' lower quality systems. After initial orientation, use of CASE tools may also result in decreased development time. As project backlog is decreased, managers are more likely to submit new project ideas.

CASE tools seem to be a manager's software dream come true. Unfortunately, many corporations which have implemented CASE are not experiencing the returns they expected. This is largely due to the fact that CASE has been sold as a "quick fix" to software problems. As a result, many corporations have failed to realize that use of Computer Aided Software Engineering involves more than simply purchasing CASE tools. CASE tools can be beneficial and indeed provide a competitive advantage. This article looks at how to implement Computer Aided Software Engineering effectively to produce the expected

returns and avoid some failures common to organizations who have utilized CASE in the past.

DEFINITION

John Teresko (1990) defines Computer Aided Software Engineering as "a methodology for developing computer software which incorporates the use of computer tools to automate various tasks in software design, coding, and maintenance" (p. 82). CASE is not a software package itself, but an approach to system development which entails using computer tools to aid in the development of a system. CASE tools include:

- Graphic tools for diagramming the components of a system;
- Code generators to produce error-free program code;
- Automatic quality checking to ensure consistency and completeness;
- Program documentation assemblers which interface with word processors;
- Prototyping tools used to develop small-scale operable models of the system which can be further developed according to users' reactions;
- Dictionary tools to record, maintain, and report on system details;
- Cost-benefit analysis tools to diagnose cost and benefits of the system; *and*
- Project management tools for organizing and regulating steps in the system's development. (Whitten, Bentley & Barlow, 1989, pp. 126, 61)

A large variety of Computer Aided Software Engineering products are on the market today. Individual tools suit different purposes. Digital Equipment's product DECdesign assists in analysis and design for VAX/VMS computer programs. IBM also has a line of CASE tools. Its AD/Cycle provides a framework for CASE use; its Developmate handles modeling and prototyping, and its AD/Cycle COBOL/360 includes a COBOL compiler (Lindholm, 1992, p. 77). KnowledgeWare Inc. sells a line of products as well. Its ADW/Documentation Workstation is a documentation tool for OS/2. Analysis and design assistance are provided by ADW/MVS, also by KnowledgeWare. KnowledgeWare's IEW Construction Station furnishes a code generator. Also popular is Intersolv's Excelerator line (Lindholm, 1992, p. 78). These tools can be used to assist in all or part of system development.

Principal Financial Group in Des Moines, Iowa, is using KnowledgeWare products to develop a Management Information System (MIS) for agency

support, according to Angela Cruse, senior systems analyst. This system allows agents to evaluate expenses and diagnose why certain aspects of their expenses are not meeting certain limits (personal communication, December, 1991).

BENEFITS OF CASE USE

Many benefits result from the use of CASE tools. As the definition of Computer Aided Software Engineering states, software design, coding, and maintenance are automated. This automation leads to gains in documentation because it becomes simpler both to generate and to revise than without this technology (Hayley, 1990, p. 19). Assembling documentation to explain computer program code is often considered a tedious task. CASE tools may include assemblers which provide, at the programmer's fingertips, a means to generate documentation—and ease of generation makes programmers more likely to develop full program documentation.

Perhaps one of the most commonly assumed benefits of CASE is that it can improve productivity. A Deloitte & Touche survey of 2,200 private sector information system departments found, however, that most projects which utilized CASE realized only a marginal improvement in productivity (Hayley, 1990, p. 21). Marcus Loh and R. Ryan Nelson (1989) have noted that productivity improvements "depend on the proficiency in a particular tool, size of the development project and the degree of tool integration" (p. 31). Productivity gains may become evident as the developer becomes more familiar with CASE.

Since initial productivity gains may be marginal, the focus of CASE tool implementation should be on quality, the major benefit of CASE tool use (Hayley, 1990, p. 19). Gains in the quality of the system arise from several sources. More complete and detailed documentation certainly improves quality. Project management tools increase quality as well (Loh, 1989, p. 31). A savings of $5 million in error reduction is cited by BDM International, Inc. of Kettering, Ohio, from its use of CASE to develop a spare parts Management Information System (MIS) for the U. S. Air Force (Teresko, 1990, p. 85). In addition, the auto quality checking system which examines programming code for errors results in higher quality program code. CASE also tends to elicit more involvement from users. Prototypes help to reveal if the true needs of users are being met. Therefore, clearer communication with users and improved user satisfaction often result (Hayley, 1990, p. 19). The outcome is all around improvement of the quality of the system, which in turn provides the organization with a competitive advantage.

Timeliness is another CASE benefit. Since code generators produce relatively error-free program code, less time is spent on testing and debugging programs, resulting in decreased system development time. Maintenance is also needed less often and is easier with a system implemented through Computer Aided Software Engineering since program code is relatively error-free (Hayley, 1990, p. 19). Improved documentation also makes system maintenance easier and less time consuming than on those systems developed without CASE. In addition, fewer people are needed on projects implemented with CASE. BDM International reported a decrease from 280 to 180 workers after employing CASE (Teresko, 1990, p. 85). Another benefit cited by Hayley (1990) is an "accelerated development life cycle" (p. 19). Acceleration of project development time and a smaller staff clearly result in financial savings which can also promote a competitive advantage.

DISADVANTAGES OF CASE USE

One of the major disadvantages of CASE tool implementation is the cost. Average budgets for CASE tool users, according to a Deloitte and Touche survey, are nearly triple those of non-CASE tool users: $47.7 million for CASE users versus $16.8 million for nonusers (Hayley, 1990, p. 19). Furthermore, CASE requires time initially to integrate (Hayley, 1990, p. 21). Shifting from an emphasis on programming to analysis and design takes time and training. While programming involves generating code based on system specifications, analysis includes investigating the situation to define the problem; design entails creating and outlining how the system will solve the problem specified in analysis. Clearly, analysis and design represent different worlds for programmers. Therefore training and time are necessary to orient programmers to these new worlds. Learning how to operate the CASE tools themselves also requires time. A University of Houston study found that in order to learn how to use CASE, 86 hours of classroom, instructor-led training are required — along with 69 individual hours of training without an instructor (Loh, 1990, p. 31).

Another primary challenge of CASE use is that its benefits are not easily quantified. This is largely because system quality improvement, the major benefit of CASE, is itself difficult to quantify—especially since the benefits are spread throughout the company (Teresko, 1990, p. 85). Computer Aided Software Engineering involves a major change in the way systems are developed. This change, like any major change, involves organizational cultural issues. In

fact, Cruse feels that the major barrier to CASE tool implemention is cultural. Ironically, information systems personnel, the ones who normally institute change, may be quite resistant to a change in their own work.

CHANGES NECESSARY FOR SUCCESSFUL CASE IMPLEMENTATION

In the Deloitte & Touche study noted earlier, only one third of the companies surveyed have used Computer Aided Software Engineering (Hayley, 1990, p. 19). If CASE has the potential to provide a competitive edge, why isn't everyone using it? As has been illustrated, the benefits from CASE use are numerous. Although there are some disadvantages to Computer Aided Software Engineering implementation, most of these disadvantages diminish after initial orientation to CASE tools. However, adopting CASE requires structural changes in the information systems department. Successful implementation involves confronting the most common reasons for failure of CASE tools. Results of the University of Houston study cited earlier suggest the three most common problems include:

- Failure of staff to grasp the approaches to systems development which CASE enforces;
- Failure of the project to gain the backing of top management;
- Failure to train the staff adequately. (Loh, 1989, p. 32)

Before managers introduce CASE tools into the development of a system, it is necessary for them to realize that the introduction of Computer Aided Software Engineering involves more than simply purchasing software packages. According to Margaret Tomczak, director of the software engineering technology unit of Eastman Kodak's Kodak Park manufacturing arm, "CASE, like CIM (Computer Integrated Manufacturing), is not something that's bought off the shelf. . ." (Teresko, 1990, p. 85). Before a CASE tool is introduced, managers must make sure their present information system is implementing the approaches to systems development which CASE enforces. The information system needs to be using structured methods, data modeling, and project management techniques. Furthermore, the context in which CASE tools will be implemented should emphasize quality (Boone & Merlyn, 1991, p. 77).

Another key element necessary for successful CASE tool implementation is management support. According to Cruse, in order for implementation of Computer Aided Software Engineering to be a success, there must be a "champion." Cruse also insists that top-down implementation is necessary

(personal communication, December, 1991). Several factors in CASE implementation make management support essential. First, as earlier noted, benefits are difficult to quantify since they are spread throughout the company. Furthermore, the initial costs of CASE are great. Without hard, quantifiable benefits to offset these initial costs, management support is essential to the continued use of the technology. Tomczak states that "Since the MIS [Management Information System] problem and CASE solution reach across functional boundaries, it can be resolved only by a management focus and a management sponsor" (Teresko, 1990, p. 85). Furthermore, it is essential for management to give ongoing support. Many of the issues surrounding CASE implementation are human issues. Tomczak observes that "In CASE, technology is only 10% of the issue; 90% is the people factor, the area where the biggest opportunities for failure lurk" (Teresko, 1990, p. 85). Only with the support of top management can these human issues be dealt with adequately. In order to maintain the support of top management, it is necessary to show continuing progress of CASE tool implementation (Teresko, 1990, p. 85).

CASE tools are relatively new technologies. Therefore few professionals have the training to implement them effectively in an organization. As a result, Computer Aided Software Engineering implementation requires an extensive amount of training. One key ingredient to achieve successful training is an accepting environment. This environment is one in which the focus is on learning. Attempts should be made to avoid conflict and opposition. Instead, patience is advocated (Hayley, 1990, p. 22). As always, there will be resistance to learning a new way of developing a system. This is why it is essential to show how CASE will assist the system developer and what benefits derive from its use. Indeed, it is necessary for training to show how CASE will improve each system developer's work (Gillies, 1991, p. 646).

According to A. C. Gillies (1991), education should focus on answering the following questions: What is CASE about? Why is it being introduced? What is the likely impact on my job? (p. 641). Since many systems developers already have some knowledge of what CASE is, the latter two questions should be the major focus of training. These questions address the greatest worries among system developers (Gillies, 1991, p. 641).

Most experts seem to agree that a few pilot projects should be implemented initially before full CASE implementation is begun. However, there is some disagreement as to which employees should be the first to use Computer Aided Software Engineering. Some experts believe the brightest and best workers who are open-minded and accepting of a new approach to system development

should be the first to use CASE. Here, CASE use is seen as a bonus for those who do their jobs well. Following this initial use, CASE would then be applied to projects to which it is most productive and to projects which are difficult to solve (Hayley, 1990, p. 22).

According to Cruse, the Principal Financial Group chose the project which would best fit the Computer Aided Software Engineering approach (personal communication, December, 1991). CASE tools were used to develop a Management Information System for field support of agency managers. The people who worked on that project were the first to use CASE tools. All in all, choosing the project which best fits CASE use seems most productive. CASE tools can be used to some extent to develop most computer systems. Remember, CASE's benefits are mainly qualitative. If implementation of the technology is to be seen as a good investment, it is necessary to use the technology first where it will be most productive. This way some quantifiable benefits will result. When working with CASE tools is seen as a bonus for some personnel, animosity might result between coworkers.

As for the training itself, it is essential to include a great deal of "hands-on" training. Most corporations have found that the training provided by the vendor is inadequate (Zagorsky, 1990, p. 26). New York Life's training program included a three day in-house workshop. The trainees used the tools hands-on from the first day. Additionally, projects similar to the actual projects for which CASE would be used were developed for the training sessions. Such simulations help to make the users feel "comfortable" with this new approach so that it can be more easily used on actual projects (Zagorsky, 1990, p. 26). New York Life also maintained an in-house expert. This person had technical expertise and an understanding of the human issues surrounding the changeover to CASE. This person enjoyed assisting others in learning how to use the new tools successfully (Zagorsky, 1990, p. 28).

Implementation of Computer Aided Software Engineering involves many structural changes. Staff may be working with an entirely new methodology and new technologies. A new approach must also be learned. In addition, CASE involves a change to more front-end system development. According to Loh and Nelson (1989), use of Computer Aided Software Engineering results in the following time devoted to the stages of system development (p. 31).

Table 1:
Percent of Time Devoted
to Each Stage of the Systems Development Life Cycle

	Planning	Analysis/Design	Implementation	Maintenance
Without CASE	10%	34%	40%	16%
With CASE	12%	43%	30%	15%

As can be seen, those organizations using CASE spend a greater percentage of their time on planning and analysis/design and a smaller percentage on implementation and maintenance than those who do not use CASE. Employees who possess programming experience exclusively will be required to assume the new role of system developer. Under the old approach, an analyst would define the problem and meet with the users. Then, a system design would be generated. Programmers next would write the code. With CASE, however, programmers must also become analysts and designers (Souza, 1990, p. 23). This often creates job stress. Cruse notes that programmers may have a problem with CASE because they currently measure their progress by the amount of programming code they generate (personal communication, December, 1991). Accordingly, a new measure of progress must be developed.

An approach to help avoid the negative repercussions of CASE implementation would include several elements. First, it is important to realize that changing to CASE will cause job stress. Workers must change both their outlook and their measures of success. Also, since CASE will not be implemented overnight, programmers will be using the new approach, yet still have to program as well (Cruse, personal communication, December, 1991). A schedule of CASE implementation should be developed and shared with employees so they can have a better sense of the time frame for the changeover to the different tasks CASE requires. Second, employees need to be assured this new technology will not cause them to lose their jobs. The biggest fear among workers involved in any change is the possibility of job loss (Gillies, 1991 p. 646). Employees should be assured the change is only a change in the focus of their jobs. Since projects will require fewer information systems personnel, more projects can be developed. Education must demonstrate to system developers that CASE is there to assist, not to replace them (Gillies, 1991, p. 646). Once anxieties are overcome, many employees will find this side of the system development process more exciting than generating code. However, this realization is not likely to be readily apparent.

CONCLUSION

Implementation of CASE tools for system development requires a focus on the long term results. Although there are some initial disadvantages to the use of CASE tools, most of these diminish in the long run. The benefits of CASE are clear. Better documentation, improved software quality, and improved timeliness all produce a competitive edge for an organization.

Nevertheless, management must remember CASE is not a "quick fix" to systems development problems. To be successful, the approaches which Computer Aided Software Engineering enforces must be in place in the organization. Management needs to give full support to implementation efforts and it is essential that staff be adequately trained. Without these three key elements, CASE is doomed to failure. Management may wonder if CASE is worth all the effort. They should remember, however, that when computers were first introduced into the work place, management was also unsure. Those who used the computer forged ahead to leave competitors behind. So it is with CASE. Higher quality software is needed, and, properly implemented, use of CASE can deliver. Organizations need to jump on the CASE wagon or be left behind in the dust.

References

BOONE, G., & MERYLN, V. (1991, June 10). Getting The Process Right. *ComputerWorld, 25,* 75-77.

GILLIES, A. C. (1991, November). Humanization of the Software Factory. *Information and Software Technology, 33,* 641-646.

HAYLEY, K., & LYMAN, H. T. (1990, Summer). The Realities of CASE. *Journal of Information Systems Management, 7,* 18-23.

LINDHOLM, E. (1992, March 1). A World of CASE Tools. *Datamation, 38,* 75-79.

LOH, M., & NELSON, R. R. (1989, July 1). Reaping CASE Harvest. *Datamation, 35,* 31-34.

SOUZA, E. (1991, Winter). The Impact of CASE on Software Development. *Journal of Information Systems Management, 8,* 17-24.

TERESKO, J. (1990, April 2). What MIS Should Be Telling. *Industry Week, 239,* 82-85.

WHITTEN, JEFFREY L. BENTLEY, LONNIE, & BARLOW, VICTOR M. (1989). *Systems Analysis & Design Methods.* Homewood, IL: Irwin.

ZAGAROSKY, C. (1990, Summer). Case Study: Managing the Change to CASE. *Journal of Information Systems Management, 7,* 24-32.