A CASE OF GREAT EXPECTATIONS

Ivan Aaen
Aalborg University, Denmark, IvanAaen@emailaddressnotknown

Carsten Sørensen
University of Aalborg, CarstenSorensen@emailaddressnotknown

Follow this and additional works at: http://aisel.aisnet.org/sjis

Recommended Citation
Available at: http://aisel.aisnet.org/sjis/vol3/iss1/6

This material is brought to you by the Journals at AIS Electronic Library (AISeL). It has been accepted for inclusion in Scandinavian Journal of Information Systems by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
A CASE OF GREAT EXPECTATIONS

IVAN AAEN & CARSTEN SØRENSEN
Department of Computer Science at University of Aalborg
Frederik Bajers Vej 7, DK-9220 Aalborg Ø, Denmark.

Abstract

One of the current buzz-words in the field of software engineering is Computer Aided Software Engineering (CASE). CASE is mainly seen as a means to improve software quality, speed up the development process and support maintenance. Despite great expectations and aspirations for its practical use, the actual diffusion process of CASE seems nevertheless to be surprisingly slow. In this article we view CASE as a technological innovation and focus on factors affecting its diffusion, based on a survey of contemporary literature. The process of introducing CASE in enterprises is characterized, and major factors affecting this process are outlined. The article concludes by pointing to some discrepancies in the literature between what is generally believed to be certain and the documented empirical foundation of this certainty. There seems to be a case of “great expectations”, in the sense that many writings are based more on expectations than empirical evidence.

Keywords: CASE, software engineering, technology diffusion, diffusion factors.
1 Introduction

According to extensive Information Systems and Software Engineering research, software development seems flooded with grave problems. Computer Aided Software Engineering (CASE) has been promoted as a means to counter or solve some of these problems, in that vendors of CASE have stated productivity gains, quality improvements and enhanced maintenance as outcomes from using their tools.

Despite the grave problems of software development and the great expectations for CASE, this new technology seems to be spreading at a fairly low pace. Most innovations seem to follow well-defined patterns of diffusion. Gurbaxani and Mendelson (1990) and (Jaakkola et al. 1991) describe the diffusion of information systems as following a S-shaped curve. This curve describes a saturation process over time, starting with none of the prospective tool users having adopted a technology as the lower limit, and with all prospective users having adopted the technology as the upper limit. In broad terms, the S-curve describes the diffusion process in three steps: (1) innovators and early adopters introduce the technology at a fairly low but increasing pace, (2) the diffusion process speeds up, reaches maximum speed, and subsequently the speed drops off as the majority of prospective users have adopted the technology, and (3) the technology diffuses at an ever decreasing pace as the technological laggards eventually adopt it.

If we assume that the diffusion of CASE is following an S-curve, then an interesting question is where we are on this curve. Are we in the early adoption phase, and if we are, how can we speed up the diffusion process? We can unfortunately not provide answers to such questions in this article. We can however present some of the major factors affecting the rate of diffusion of CASE, and discuss their implications for the process of introducing CASE into an enterprise.

In this article we view CASE as an established fact. In the real world of enterprises and practitioners CASE is causing change: A lot of vendors are promoting a number of different tools supporting different methods and techniques, and software engineering organizations around the world are buying this new technology (Parkinson 1990). The introduction of CASE tools in organizations doing software engineering affects both the development process and the resulting products. We primarily see the introduction of CASE as a process innovation, since the introduction of new tools imply changes in the work procedures as well as in the final products. This article adopts the perspective of the user organization, having a fairly positive stance towards CASE, and assuming that this technology has something to offer to many enterprises.

The term CASE as used in the literature can denote rather different things, so it is appropriate at this point to clarify how we use it in this article. A CASE tool can support a given software engineering method, or perhaps a number of methods, used in the software engineering process, but we need some more or less rigid criteria for discerning CASE tools from other tools supporting the software engineering process. Gane (1990) suggests a definition — which is sufficient for
our purpose — stating that: "... the distinguishing characteristic of a CASE product is that it builds within itself a design database, at a higher level than code statements or physical data element definitions". This implies for example that debuggers and test data generators would not qualify as CASE tools (Gane 1990).

The following section outlines some major problems in software engineering and difficulties in actually applying structured techniques as a remedy. In the light of this, the promises of CASE are presented and its apparently low diffusion rate discussed. Section 3 depicts some central activities in the introduction of CASE: purchasing a tool, modifying, adapting, and putting it into use. Section 4 presents — based on innovation theory — some general perspectives on factors affecting the diffusion of new technology. Factors affecting the diffusion of CASE are derived from contemporary CASE literature and classified according to these categories. Section 5 investigates implications for the introduction process, and Section 6 draws some conclusions.

2 Problems and Promises in Software Engineering

2.1 Software Engineering Problems

The costs of software development and maintenance have for years gone up relative to the total costs of computer systems (Boehm 1976, Boehm 1981, Boehm 1987, Boehm et al. 1984, Boehm & Papaccio 1988). While hardware costs measured in relative as well as in absolute prices have dropped, this is far from the case for software. Whereas the demand for software is increasing as a result of the widespread use of computers in society, the capacity for developing software is not expanding accordingly. Nejmeh (1988) reports a 12% annual growth in the demand for software while the resources for its development only increased by 4% a year. The supply of personnel lags behind the demand for software, and programmer efficiency must be increased if the gap between software needs and production is ever to be closed (Norman et al. 1989). This is why the so-called software crisis and software backlog in recent years has received increased attention in industry and academia.

The large software costs originate from a number of related problems in development and maintenance: low productivity, insufficient quality assurance, failure to reuse software components, low documentation standards and configuration management shortcomings. In response to these problems a number of efforts have been made to develop standards and tools for more effective software development and maintenance. Key elements in these efforts have been analysis, design, and documentation, with the goal of combining efficient development of software specifications and design with the development of corresponding documentation for the produced software system.

Structured techniques — structured programming, analysis and design — emerged in the 1970s as a part of these efforts. The aim of these highly celebrated
techniques was to bring some discipline and efficiency to software development, but to date their impact on practical software development has been far from impressive. According to one of the founding fathers of these techniques — Ed Yourdon — only about 10% of the data processing organizations in North America practice structured techniques in a disciplined fashion (Yourdon 1986). A number of inconveniences in using structured techniques can be mentioned as partial explanations, as findings from Schweitzer & Teel (1989), and Aaen (1990) suggest: (1) models and documentation are very time-consuming to develop; (2) inconsistencies, errors and ambiguities in and between different products are often difficult to detect; (3) it is difficult for any individual to establish an overview of all relevant documents except in small projects; and (4) updating and changing products is difficult and time-consuming.

Thus, authors like McClure (1988), Yourdon (1986), and others see the tedious, slow, and manual work that occurs when using structured techniques as major reasons for their limited practical use. This may represent a major problem to the software industry, since for instance; Capers Jones (1986) reports structured methodologies to have a markedly positive impact on productivity regarding development schedule (calendar months), development effort (person-months), and 5-year maintenance effort.

2.2 Promises of CASE

Until recently, the structured techniques have primarily been practiced manually using paper and pencil. This mode of operation accounts for much of the tedium and slowness in using them, and CASE has been advanced as a means for making structured techniques more practicable. Software developers are expected to be more positive towards using specific techniques when these are supported by computer tools for reducing the tedium of modeling, checking, and documenting. Consequently, the tools can be expected to: (1) lead to more disciplined and uniform work procedures; (2) improve consistency and completeness in the models via extensive checking facilities; (3) enhance quality control and assurance; (4) support reviewing procedures by eliminating a host of basic checks, leaving more room to concentrate on the essentials of the deliverables; (5) enhance project management by facilitating a better monitoring of status and progress; (6) improve project planning and estimation by providing more tangible empirical data from other projects; (7) improved possibilities for reverse engineering via automated generation of structure diagrams from source code; (8) bridge the gap from design to implementation by offering either code generation facilities or interfaces to programming environments; and (9) support the production of high quality documentation (Bromell & Preston 1989, Case Jr. 1986, Chen et al. 1989, Gane 1990, Gibson 1989b, McClure 1988, McClure 1989b, McClure 1989a, Norman & Nunamaker Jr. 1989, Orlikowski 1988, Winsberg 1989, Yourdon 1989).

CASE may thus bring about major changes to future application develop-
ment by making the use of structured techniques more effective and efficient. Productivity increases and quality improvements are predicted to follow the introduction of CASE. Although the promises of CASE are great, the diffusion of this innovation into industry is going slowly. Yourdon (1989) reports that only 2% of the systems analysts in the United States had CASE tools available to them in 1987, and that estimates indicate that only 10% will have them in 1990. Similarly Danziger & Haynes (1989) report that less than 10% of the mainframe users in North America have implemented CASE by 1989, and of those, more than 80% are using CASE only on an experimental basis.

3 Introducing CASE in Organizations

If we consider an enterprise introducing CASE, how then can this process be described? The introduction of CASE results, among other things, in a gap between the existing manual work procedures and those supported by the CASE tool. As a result of this, organizations introducing CASE may face a pressing need to mutually adapt the a priori work procedures and tools to one another (Smolander et al. 1990).

Zaltman, Duncan and Holbek review several different models for describing the process of innovation in general (Zaltman et al. 1973). Based on this, they present a process approach to the study of innovation, viewing it as "...a set of stages ordered along the temporal dimensions of their anticipated sequence". One major virtue of this model is that it views the introduction of new technology as an unfolding process reaching far beyond the point of decision upon whether or not to adopt an innovation. However, this model also suffers from two major drawbacks. The first is that it tends to portray the innovation process as a set of stages, rather than activities. This can lead to misrepresentations of the innovation process, as the process may well unfold in a circular or concurrent fashion. The second drawback is that it tends to downgrade the issue of mutual adaptation by viewing the technology part of the process as more or less fixed.

In an attempt to highlight the mutual adaptation of CASE tools and organizations and focus on time-independent activities, we will make an explicit analogy to standard application packages viewing CASE tools as a new group of standard applications. The important point in the analogy is the activities involved in the introduction process, rather than the specific characteristics of the technology. There are of course significant limitations to this analogy, as each group of standard applications has different technical and application area characteristics. However, using the analogy with caution offers considerable advantages, in that we can use experiences from a number of related studies (Anveskog et al. 1983, Anveskog et al. 1984, Sørensen 1989, von Hellens 1990).

For the purpose of clarity, we choose to analyze the diffusion of CASE at the level of the enterprise, taking a rational stance towards the introduction process. Others have analyzed introduction processes from a sociological point.
of view; e.g., Orlikowski (1988, 1989) has analyzed the organizational impacts of CASE technology, and Sørensen (1989) has analyzed the introduction of standard application packages using a framework based on Franz and Robey (1984).

The process of introducing standard application packages in organizations can be characterized by four main activities, as suggested by Anveskog et al. (1983, 1984) and Sørensen (1989). This implies the following four main activities, when introducing CASE tools: (1) purchasing the tool, (2) modifying the tool, (3) adapting the organization to the use of the tool, and (4) putting the tool into use. The sequencing of these four activities may vary in actual situations, and likewise the extent to which each of the activities does in fact occur.

**Purchasing the tool** covers a number of possible activities. It involves both surveying the market and analyzing the enterprise in order to identify possible choices. It also involves demonstrations, tests and negotiations internally in the enterprise, as well as externally, concerning which product might be bought. Finally, this activity involves deciding whether or not a CASE tool should be bought. When standard application packages are chosen as an alternative to dedicated systems, the choice process is critical. This is also true when CASE tools are evaluated and chosen. Anveskog et al. (1984) present a 15 step method for choosing standard application packages, including elements such as: Analyzing needs, eliciting company foundation, analyzing market, evaluating vendors, matching company needs and system properties, negotiating, and making decisions. This approach can also facilitate the process when CASE tools are to be purchased.

**Modifying the tool** implies adjusting the tool to the relevant work procedures of the enterprise. This activity may seem an unexpected one, as CASE tools generally cannot be modified by the tool-using enterprise. Such direct modification of CASE tools is very unusual (Smolander et al. 1990), and according to a survey of 237 CASE users in Holland, the users are dissatisfied with the limited possibilities for adapting the tools to their own standards (Wijers & van Dort 1990). In time, we may expect new tools conforming to the notions of flexible tools and CASE shells (Smolander et al. 1990). According to Yourdon (1989), modification of CASE tools is already now a topic of interest to purchasers of large CASE installations.

**Adapting the organization to the use of the tool** involves adjusting the relevant work procedures of the enterprise to the selected tool. There may for example be discrepancies between the work procedures of the enterprise and the tool, even though the tool supports the methods officially subscribed to by the enterprise. Furthermore, there may be programmers unwilling to use the new tool for one reason or another. When a CASE tool is introduced into an enterprise, cultural norms, values, beliefs, and vested interests are challenged. Surveys done by Smolander et al. (1990) and Wijers & van Dort (1990) treat the issue of organizational adaptation to the use of CASE tools.
Putting the tool into use concerns the actual use of the tool. Putting a tool into use implies an initial learning process and subsequently some maintenance of the tool as well as of the new qualifications required for using it. Possible options for learning how to use the tool include experiments, critical vs. non-critical pilot projects, full scale projects, courses, and asking colleagues. In practice the learning process may comprise a combination of these, and several papers discuss learning aspects of CASE in the software engineering curriculum (Aaen 1990, Amadio 1989, Mynatt & Leventhal 1989, Weiss 1987, Wilcox 1988).

4 Diffusion of CASE tools in Organizations

When seen from the perspective of innovation theory, a number of essential factors affecting the adoption of CASE can be highlighted. Freeman (1981) presents five categories of factors affecting innovation diffusion processes in general: (1) profitability, (2) scale of investment, (3) technical characteristics, (4) acceptability, (5) change agents. To this list we would like to add one more: (6) tool user qualifications.

This section presents these six categories. Based primarily on a survey of contemporary literature, factors affecting the diffusion of CASE we identify, and we discuss the implications of these factors for the innovation process. Each category of factors is presented via statements based on conclusions and opinions found in the literature. The findings are summarized in each subsection.

4.1 Profitability

With respect to profitability, CASE is assumed to remedy the software crisis (Nejmeh 1988), and economic benefits are advanced as driving forces for changing to CASE (Norman et al. 1989). Managers tend to view CASE as a key to productivity and success in exploiting marketing opportunities (McClure 1988), or as a means to countering a software backlog, and in time possibly a fix for rising programming costs (Francis 1988, Chen et al. 1989). Danziger & Haynes (1989) refer to a 1988 report indicating a 25% productivity improvement for analysis, 20% for requirements definition, and 30% for maintenance, and Coolidge (1988) reports surveys showing productivity increases from using CASE ranging from 30% to 600%. Coolidge refers to evidence that CASE pays immediate dividends in higher productivity and better software as anecdotal. This anecdotal character of productivity assertions reflects a number of problems in measuring the effects directly (Kemerer 1989, Coolidge 1988).

The issue of profitability of CASE is not a simple one, and not surprisingly a number of studies seek to circumvent these problems by measuring the perceptions of software engineers (Burkhard & Jenster 1989, Necco et al. 1989, Norman & Nunamaker Jr. 1988, Parkinson 1990, Smolander et al. 1990, Wijers & van Dort 1990). Most studies of this kind report that software engineers perceive CASE technology to improve their productivity (Necco et al. 1989,
Norman & Nunamaker Jr. 1988). In contrast Wijers & van Dort (1990) report that Dutch users emphasize another tool characteristic — quality improvement — as the major reason for purchasing CASE. Burkhard & Jenster (1989) asked prospective and current users of CASE for their ranking of the most important outcomes of CASE. Prospective users ranked speed of development as the most important, and ranked cost savings as relatively unimportant. Current users, on the other hand, ranked improved accuracy on top, and ranked cost savings high as well.

A number of authors focus on intermediate and long term profitability, emphasizing the quality improvements gained from using CASE. Some argue that CASE helps control the number of bugs in software (Foremski 1988), and some claim that CASE can improve quality and quantity at the same time (Burkhard 1989, Necco et al. 1989). Burkhard predicts significant reductions in maintenance activities, and a shift in available resources from maintenance activities to development of new systems (Burkhard 1989).

Finally, some authors describe the productivity gains as contingent on other factors. Some argue that productivity gains will increase the longer a company employs CASE (Gibson et al. 1989). Others argue that CASE tools must be used properly in order to simplify project development (Miller 1989). Bromell & Preston state that the tools must be introduced and managed carefully, and that the projects must have clearly defined, restricted scopes and limited objectives to give the best results in quality, reliability and productivity (Bromell & Preston 1989).

It is almost impossible to assess the benefits from using CASE technology. The real productivity gains probably lie in systems, rather than in particular sub-processes such as analysis or design per se. Many of the benefits of CASE will be felt downstream in development, maintenance and management, and are difficult to quantify. Furthermore, many of the benefits will presumably appear in the quality of the software, and the speed with which new and modified software can be launched. Such improvements are difficult to measure, and may vary in importance between enterprises and business areas. Finally, despite strenuous efforts in software engineering, much of software development is still an art form, highly dependent upon the particular individuals involved, and consequently open to variable levels of performance. The benefits of using CASE therefore largely depend upon the particular circumstances in which it is used.

Beyond this, problems arise when deciding on the scope of impact of CASE tools. At least three alternative scopes can be defined: (1) productivity improvements in diagramming and document production. This scope allows for the most measurable, but probably also the least informative, observations; (2) tool impact on the software life cycle. Besides speeding up analysis and design activities, the tools may improve design, e.g., by reducing the error rate. Profitability may either be measured relative to the costs of the software development process, or relative to the total life cycle costs of the developed software, since quality improvements in the design can be expected to reduce subsequent costs significantly.
(Boehm 1987); (3) downstream benefits. Introducing CASE in most cases seem to warrant the introduction of new methods and work procedures, e.g., project management, configuration management, and quality assurance. These spin-off changes may contribute significantly to the profitability of CASE.

In summary, there seems to be almost universal agreement on the short or long term profitability, although precise empirical observations are far from abundant. The effects are described as direct (productivity gains) or indirect (quality gains). If this is true, CASE should be expected to diffuse rapidly. Freeman (1981) however argues that an optimistic view of the profitability of any new technology must be qualified with the findings of innovation research. A number of studies of early adopters of new technology have conclusively shown that in the early stages of a diffusion process, great uncertainty about actual and future profitability is the rule rather than the exception. Moreover, even when high profitability has been convincingly demonstrated, large numbers in the potential adopter population may continue to behave “irrationally” by refusing to invest in the innovation. Freeman concludes that purely economic analysis cannot be divorced from the social and cultural context.

4.2 Scale of Investment

CASE requires considerable hardware and software investments compared to the traditionally low investments per employee of system developers. Burkhard (1989) reports that the costs for hardware, software and initial training for a user to be approximately $30,000 per workstation, and Gibson (1989a) estimates an average expenditures of $3,000–$6,000 for consulting, training and coaching one manager, systems analyst or programmer/analyst. Spurr (1989) points to the dual overheads of training and management: training in CASE software and associated methods is essential but time consuming, and existing project management techniques will have to be reviewed and changed, consuming additional time.

Burkhard (1989) acknowledges that the costs per workstation are relatively high, but urges providing enough workstation access so as to not discourage or hinder the use of CASE. Gibson (1989a) argues that the productivity gains warrant the investment. On the other hand, Chen et al. (1989) describe the costs of implementing and using CASE as extremely high, whereas the benefits will not be realized in the short term.

Apparently, elaborate cost/benefit-analyses of CASE have not yet been published. As was the case above, the overall assessment found in the literature is that CASE will pay — at least in the long run. Specific assessments on how well it will pay, and how soon costs and benefits will break even are, however, not to be found. From a “rational” perspective, costly innovations should ceteris paribus be expected to be adopted more slowly than those requiring a small scale of investment (Freeman 1981). If this is true CASE, should be expected to diffuse slowly. Based on Freeman, this again must be qualified with the observa-
tion that marketing considerations, vested interests etc. may be influences which outweigh the simple issue of scale of investment outlay. Such influences seem to have been important in the case of Computer Aided Design (CAD), where advertisements, brochures, and annual reports from enterprises in the 80's frequently used CAD systems as a marketing tool for advertising a leading-edge position in high-technology.

4.3 Technical Characteristics

CASE tools are designed to counter the problems in using structured techniques mentioned in Section 2. Some describe CASE as the automation of step-by-step methodologies designed to automate the drudgery of development and free the developer to solve problems (McClure 1988). Others describe CASE as taking the pain out of using graphical methods, as providing more effective ways of employing methodologies and techniques, and as facilitating the integration of structured methodologies and prototyping (Gibson et al. 1989). Finally some point to the improved maintenance of documentation due to the instant registration of changes (Berman 1989).

These positive effects should warrant a rapid diffusion of CASE. As Freeman (1981) reports, much innovation in machinery has been concerned with the achievement of greater precision and reliability, higher speeds, reduced maintenance, and other superior performance characteristics. This pursuit may sometimes contradict the goal of increased profitability, at least in the short run. Users of a new technology may, however, be ready to pay a higher price for improved performance, if increased long-term profitability and convenience seem likely to offset reduced short-term profitability.

Still, the technical characteristics of CASE tools are not all that impressive, and some tools have perhaps been over-sold. Parkinson (1990) describe early tools as limited in the functions and features they offered, and in the size and type of project they could support, and even worse: these limitations were unknown to both buyers and vendors. Furthermore, the tools were far from bug-free, which meant that the design database of a project could become corrupted, causing severe problems. Parkinson also ascribe the slow adoption of CASE to early problems in sharing data and having multiple users on a single project. In addition, the methodological rigidity of most CASE products can also discourage their use (Parkinson 1990): studies show that CASE users are dissatisfied with the inflexibility and rigidity of the tools (Smolander et al. 1990, Wijers & van Dort 1990). If CASE tools are based on a single methodology, they may fail to meet market demands (Berman 1989, Coolidge 1988). There are, however, signs on a shift towards CASE producers providing more flexible means of integrating methodologies (Berman 1989).

Most of the CASE tools on the market support either data-flow oriented or information engineering-oriented methods (Berman 1989, Gane 1990). One can speculate that the slow adoption of these tools to be caused by the fact that they
do not properly support object-oriented methodologies. We do not, however, have any indications that suggest a wide-spread use of the latter methodologies in industry. Furthermore, it has been shown that for example Structured Analysis and Structured Design can be used in an object-oriented manner (Sandvad 1990). Consequently, we do not consider this factor to be of paramount importance.

CASE technology is still in its infancy and potential users seem to be waiting for new tool generations and, not least, de facto standards, to emerge. As an example many enterprises for a long time postponed the introduction of CASE technology awaiting IBM's announcement of a common standard-repository.

4.4 Acceptability

CASE tools can offer benefits to developers with respect to autonomy, responsibility, pride in work, and motivation (Danziger & Haynes 1989). Kemerer notes that the growth in CASE sales is due to a demand from developers for some means to improve quality and productivity (Kemerer 1989). The ease of making and changing descriptions with CASE could be a major factor in motivating developers, and some developers find the process of systems analysis and design more enjoyable when the process is less tedious (Necco et al. 1989).

Despite these observations, resistance to change can also be expected to play a role in the diffusion of CASE. CASE can produce a culture shock in an enterprise, due to changes in development methods and management techniques (Spurr 1989, Wasserman 1988). Adopting new, or changing existing, work procedures following the introduction of CASE means changing the developers' job domain (Burkhard & Jenster 1989, Norman et al. 1989). Introducing CASE may advertently or inadvertently serve as a powerful means of tightening the organizational control via standardization of the work procedures and the division of labor. Furthermore, CASE can be seen as a threat to hard-earned expertise and the quality of work life, and indications are that older developers resist such changes more than younger ones (Kemerer 1989, Norman et al. 1989).

Both Kemerer and Orlikowski emphasize the importance of the social consequences of CASE and suggest that these consequences be made clear to all participants (Kemerer 1989, Orlikowski 1988, Orlikowski 1989). Several sources point out that it may be crucial to the introduction of this technology that perceived threats are removed early in the process, allowing participants to see CASE as an important means to improve planning and development activities (Danziger & Haynes 1989) and therefore to expect the benefits to outweigh the drawbacks (Norman et al. 1989).

Resistance to change seems to be an important impediment to the diffusion of CASE. Perceived or real threats to job content, autonomy and qualifications may create opposition, whereas aspirations for more autonomy, greater responsibility, less tedium and better quality may balance such resistance. There seems to be a parallel between introducing CASE to computer professionals and introducing information systems in user organizations. In both cases resistance to change
may have a great effect, whence Smolander et al. (1990) have investigated how to combine tools and methods, and draw a parallel between methodology adaptation and introduction of information systems in general.

4.5 Change Agents

At the level of the enterprise, the concept of change agent refers to people engaged in paving the way for change. Depending on their position in the enterprise, change agents may engage in: (1) decision making concerning a change; (2) allocating economic or physical resources; (3) formulating and communicating goals and objectives; (4) training in the operation of equipment; (5) consultancy — answering questions, troubleshooting, giving advice; (6) facilitating — preparing users for change, negotiating, providing base for mutual understanding, preparing for changes in professional roles; and (7) communicating enthusiasm and commitment, etc.

Change agents are pronounced to be important to the introduction of CASE. Management’s taking responsibility for managing the introduction process and communicating that commitment to the other participants is, by some authors, mentioned as one important factor (Danziger & Haynes 1989, Norman et al. 1989). Bromell & Preston mention “selling” the investment to top-management level as well as “selling” the new techniques to developers (Bromell & Preston 1989). Both of these factors involve change agents. Training and use of consultants is also seen as important in achieving productivity gains.

Change agents may play a major role in the introduction of CASE. They may be managers or developers opting for CASE for one reason or another. The main role of change agents — apart from taking part in coordinating practical matters and distributing information — is to push the technology into all levels in the organization and to facilitate the change process.

4.6 Tool user Qualifications

As far as drawing diagrams and producing documents are concerned, the skills required to use CASE tools are nearly the same as those involved in manual procedures. Using the fuller range of the facilities of a CASE tool distills the skill component for concentrated attention by the analyst/designer. Focus shifts from making the models right to making the right models. Thus the basic “driving” skills come relatively easy; however, best practice will most probably take considerably longer time to emerge, contingent on the size of the gap between the methodological and theoretical stance prior to adopting CASE on the one side, and the methodological and theoretical requirements for obtaining full benefits from the tools on the other. In line with this, a host of writers assert that understanding and practicing structured methodologies is a major competence factor in using CASE tools (Bromell & Preston 1989, Burkhard 1989, Burkhard & Jenster 1989, McClure 1988, Necco et al. 1989,
Nejmeh 1988). For management, CASE also requires a considerable learning process and time for contemplation with respect to location of equipment, number of workstations, training, work procedures, and adoption of standards.

High learning curves can be a major force against effective introduction of CASE (Norman et al. 1989, Suomi 1990), and it is emphasized that training is more than learning how to use the tool (Burkhard 1989, Gibson 1989a). A systematic monitoring of the training — in order to assess the different types, amount, duration, and depth of training each individual requires — can facilitate the introduction of CASE (Gibson 1989a). Furthermore, putting more effort to the relationship between methodologies, management techniques, and CASE tools seems to be more important than quickly adopting technological enhancements (Wijers & van Dort 1990). The learning curve will largely depend upon the use to which the CASE tool is put. A tool used only for document production and drawing will be easier to assimilate in an enterprise than one used in the full range of the software development process, or used to capture downstream benefits. On the other hand, a limited use may just as well reflect an inability to use CASE effectively in the enterprise.

To summarize, the developers’ general qualifications and tradition for using structured methodologies seem to be of major importance in relation to introducing CASE. High learning curves may be an important obstacle to diffusion, i.e., the organizations may not want to invest the time and resources necessary in order to change the developers’ and managers’ qualifications re structured methods. On the other hand, well-qualified users of the CASE tools may produce new ideas for effective work procedures based on the tools, thereby acting as what von Hippel (1988) terms “sources of innovation.”

5 Implications for the Introduction Activities

What can we learn from these factors about the introduction of CASE in organizations? Although only few conclusions emerge from the literature survey presented above, some speculations concerning the introduction activities can nevertheless be offered. This section will relate the major findings from the previous section to the four main introduction activities for the purpose of clarifying the importance of each category for the introduction process. The discussion is summarized in Table 1 (p. 17) and Table 2 (p. 18).

In relation to purchasing a tool, the issue of profitability emerges. Since little information is available on the profitability of CASE, prospective enterprises may face uncertainty with respect not only to the possible benefits from adopting CASE, but also to the risks involved in the adoption. One obvious risk is to make a fruitless investment, another and possibly greater one is to suffer a negative impact, i.e. experiencing major setbacks in critical projects due to failures in the introduction of CASE. In this respect, it is worth noting that there seem to be few examples of clear-cut disasters, where the introduction of CASE has led to
the failure of a software development project. Although there are examples where the objectives for introducing CASE have not been met (Norman et al. 1989), harmful impacts seem to be rare, at least they are rarely reported. Although enterprises may have good reasons for being reluctant to disclose disasters, the scarcity of independent research reports of CASE failures suggests that failures are relatively limited in number and size.

Furthermore, irreparable long-term damage to the organization is not likely to occur, as CASE will not substitute manual procedures to the extent where existing qualifications in the organization wither away, thereby excluding the option of abandoning CASE at a later stage. This indicates to us that the risks involved in putting the tool into use in “real” projects — as opposed to experiments — may be limited. Furthermore, there seem to be few indications of immediate productivity gains; rather productivity gains, if any, are likely to emerge late in, or after the close of, the actual introduction process. In fact, productivity may even decrease during this process as a result of the learning curves involved in putting the tools into use, and the adaptation of the organizations to the use of the tools.

Investments in basic hardware and software procurements following the choice of tool will make up a large part of the total CASE investments. In addition, investments in tool user qualifications must be assessed when purchasing the tool. The qualifications of the users influence the changes in work procedures that may be implemented, and at the same time one of the objectives of introducing CASE may be to change the qualifications of the users (Andersen et al. 1990).

The technical characteristics of the tools are in focus throughout the introduction process. These characteristics will normally be explicitly evaluated prior to the purchase and modification activities, but some technical characteristics — correctness, reliability, efficiency, integrity, usability — can first be thoroughly evaluated as part of adapting the organization to the use of the tool, or even putting the tool into use.

As for modifying the tool, the tool’s flexibility is of paramount importance. Present tools generally offer little flexibility, and enterprises mostly face the choice between abstaining from major modifications or investing large resources in this activity. Flexibility will likely become a major factor in the competition between producers of CASE tools.

The manpower investments in the process of adapting the organization to the use of a tool may amount to a considerable sum, as may also be the case with the possible expenses for hiring consultants to assist in putting the tool into use in the first projects. Large expenses for courses as well as for consulting, training, and coaching are also likely to occur.

When putting a tool into use in an organization, low user qualifications may well introduce major risks of failure in the introduction process, and the costs for consulting, training and coaching in order to counter these low qualifications can be very high. If on the other hand the users are competent in using structured methodologies, the process of putting the tool into use may be greatly alleviated.
<table>
<thead>
<tr>
<th></th>
<th>Purchasing the tool</th>
<th>Modifying the tool</th>
<th>Adapting the organization to the use of the tool</th>
<th>Putting the tool into use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Profitability</strong></td>
<td>Widespread agreement on the direct or indirect profitability of CASE. Profitability will likely depend on the application area and market positioning.</td>
<td>—</td>
<td>—</td>
<td>Profitability will follow a learning curve. Technical benefits will emerge early, and down-stream benefits late. CASE failures seem to be rare, and the introduction process is likely to be reversible.</td>
</tr>
<tr>
<td><strong>Scale of investment</strong></td>
<td>Investments in hardware and software are high. Duration of the payback period is uncertain.</td>
<td>Presumably either low or very high.</td>
<td>Investments in courses, consultants, training, and coaching depending on the <em>a priori</em> methodology in the enterprise. Investments in the development of effective work procedures.</td>
<td>Investments needed for further development of qualifications and working procedures.</td>
</tr>
<tr>
<td><strong>Technical characteris-tics</strong></td>
<td>—</td>
<td>Tool rigidity hinders widespread diffusion.</td>
<td>—</td>
<td>Tool rigidity may be a problem for achieving full benefits in the software life cycle, and for achieving full down-stream benefits.</td>
</tr>
</tbody>
</table>

Table 1: *Profitability, Scale of investment and Technical characteristics related to the main activities in the introduction process.*
<table>
<thead>
<tr>
<th></th>
<th>Purchasing the tool</th>
<th>Modifying the tool</th>
<th>Adapting the organization to the use of the tool</th>
<th>Putting the tool into use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Acceptability</strong></td>
<td>Mixed feelings to be expected. Factors affecting success and failure akin to other information systems.</td>
<td>Same</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td><strong>Change agents</strong></td>
<td>Needed for preparation of the process, provision of resources for acquisition, and for pushing management.</td>
<td>Needed for provision of resources for modification, and for pushing management.</td>
<td>Needed for provision of resources for adaptation, and to facilitate, train, and push users.</td>
<td>Needed for provision of resources for operation, and for appropriate use of the tool.</td>
</tr>
<tr>
<td><strong>Tool user qualifications</strong></td>
<td>A priori qualifications of great importance for the innovation process.</td>
<td>—</td>
<td>Development of adequate qualifications a sine qua non.</td>
<td>User qualifications must be maintained and further developed in order to go from 'driving skills' to proficiency.</td>
</tr>
</tbody>
</table>
Designing the learning process properly with respect to \textit{a priori} and required qualifications can perhaps be a major prerequisite for the successful introduction of CASE.

Last, with respect to the introduction process in general, problems with accepting the tool and other organizational issues may emerge in each of the four introduction activities, since each provides ample opportunities for negotiating the change process. Similar, change agents may play a role in all of the four activities. Different individuals may act as change agents in different activities, but lacking a change agent in one of the activities may threaten the entire process.

6 Conclusion

The CASE literature seems to be based more on expectations than on empirical evidence, a point also made by (Wynkoop & Conger 1990).

There seems to be undisputed agreement in the literature on the profitability of adopting CASE in enterprises, be it with respect to productivity in the production of deliverables, speed of development, or the quality of deliverables. On the other hand, only few accounts have been published on the actual profitability of this technology. It seems unclear whether profitability will be obtained over the short, medium, or long term. Furthermore, it seems unclear whether profitability may be attributed to improvements in diagramming and document production, to life cycle impacts, or to down-stream benefits. Similarly, accounts of profitability with respect to specific application areas and types of tools are needed.

Regarding scale of investment, all indications are that large investments can be expected in hardware, software, adaptation, training, and management. Little is reported in the literature on the duration of the investment and pay-back periods. Cost/benefit analyses of CASE adoption are not to be found.

Technically, the tools are purported to contribute to less tedious and easier maintenance. The functionality of the tools is however limited and error-prone, and the tools generally offer little or no integration of methodologies. Reports on the actual constraints imposed on methodology adaptation in practical settings seem to be rare. The number of features of specific tools being used in practical setting, and the intensity of their use, is largely unknown.

The social consequences of CASE seem to be of great importance. This field of study is however still in its infancy, and conclusive evidence is not yet available. Accounts are needed, e.g., on the changes imposed on developers, on users, and on managers, as are reports on the changes in relationships between these groups in various settings.

Change agents are reported to play a role in decision processes, in the management of the introduction process, in persuading developers, and in communicating commitment to the adoption of CASE. The role of change agents for facilitating, for training, and for consulting has not been treated in any detail.
Regarding tool user qualifications, there seems to be general agreement on the importance of a match between users' competence and methodological demands. There are, on the other hand, few results on the length and shape of the learning curve, and on which qualifications are needed in order to effectively utilize CASE tools.

Returning to the questions raised in Section 1, we seem still to be in the early adoption phase, since the reports quoted in Section 2 indicate a low diffusion rate for this technology. Our study does not point to simple measures for speeding up the diffusion process in industry. On the contrary, it indicates that the diffusion of CASE depends on a complex mix of factors.

Measurements for speeding up the diffusion process can be expected to require a balancing among all six categories of factors. This, in turn, suggests a need for developing a meta-level framework embracing these categories. This line of reasoning bring us in consonance with writers like Danziger & Haynes (1989), Norman et al. (1989), and Orlikowski (1988), describing the diffusion of CASE as contingent on a complex mix of organizational and technological conditions.

Our findings thus point to two areas of further research. The first area is comprised of the lacunae in our body of knowledge indicated in the beginning of this section. The second area concerns the development of conceptual and theoretical contributions, combining innovation theory and organizational change theory. Such contributions could well illuminate the interactions and synergies among the six group of factors identified in this article.

Acknowledgment

Birgitte Gregersen gave us the idea of using Freeman’s categories of technology diffusion factors and suggested adding tool user qualifications as a sixth category. Her contribution is gratefully acknowledged. We also would like to thank the two anonymous reviewers and the participants on the 13th IRIS Conference for constructive comments and suggestions. We also thank Michael J. Manthey for numerous language improvements. All errors in this paper naturally remain the responsibility of the authors.

References


