ISRA-90 Precedings - volume I

The IFIP TC 8 WG 8.2 Working Conference on

THE INFORMATION SYSTEMS RESEARCH ARENA OF THE 90'S:

CHALLENGES, PERCEPTIONS, AND ALTERNATIVE APPROACHES

Copenhagen, Denmark, December 14 - 16, 1990

Edited by: Hans-Erik Nissen, University of Lund, Sweden
Heinz K. Klein, SUNY Binghamton, USA
Rudy Hirschheim, ISRC Univ. of Houston, USA

Lund, Sweden, October 1990

A REVIEW OF COMPUTER AIDED SOFTWARE ENGINEERING RESEARCH METHODS

Judy L. Wynekoop
Sue A. Conger

Georgia State University
Atlanta, Georgia

ABSTRACT

This paper examines computer-aided software engineering (CASE) research to understand current CASE knowledge and to propose future research. An applied research paradigm consensus in CASE identifies a bias toward building environments rather than evaluating them. The general lack of research using action research, case studies, field studies, and experiments is identified as stalling more rapid development of a cumulative body of knowledge. This applied research bias limits what can be learned for future development. More balance in the type and methods of research with respect to CASE is needed. Researchers need to become more self-conscious in selecting research methods for CASE research and more diligent in evaluating applied development of CASE tools and environments.

1. INTRODUCTION

Computer aided software engineering (CASE)[1] is one of the most important and frequently discussed topics in information systems today. The number of papers discussing the topic grew from tens of articles in the early 1980's to hundreds in the last two years. Much CASE research appears to have little regard for development of a cumulative tradition, to ignore the software development context, and to have very different manifestations of the same components across different environments. In the quest to determine why these situations exist, CASE research is evaluated within the framework of research method and purpose.

Historically, CASE tools have differed markedly in scope and focus. Scope varies in how many application development activities are incorporated into the product. For example, some systems concentrate on program development [cf. 27] while others integrate the entire project life cycle [cf. 6]. Emphasis within an environment differs. Some environments focus on minimizing redundancy of tools and tool fragments, others on the user interface, and still others, on the amount of functional integration provided. While some facilities of CASE environments seem universally

recognized as desirable (e.g. editor, documentation aids, graphic presentation methods), there is little agreement on the design, integration, scope or focus of individual tools [41]. With these differences, one would expect to find areas of consensus with a growing body of cumulative knowledge. Yet, what we find is that we are no closer to consensus in basic areas, e.g. interface design, tools, etc., than we were ten years ago [cf. 25].

This paper attempts to understand what we, as researchers, have done to propose what we should be doing to shape the CASE future. First, several frameworks classifying research are evaluated to select the dimensions -- research method and purpose -- appropriate to CASE research classification. Next, the different types of research methods are discussed, including how each might be applicable to CASE. Then, a sample of CASE research from the last three years is evaluated within the framework of research method used and research motivation. The discussion of results addresses each of the apparent shortcomings mentioned above. Finally, the conclusion identifies future directions for researchers in CASE.

2. RESEARCH FRAMEWORKS

There are many frameworks that could be used to facilitate our understanding of CASE research and to identify areas for research [c.f. 1, 2, 14, 16, 19, 21, 22, 24, 28]. Four of these frameworks specifically identify research method as one dimension [cf. 1, 2, 21, 28]. Only one framework of CASE research [16] includes the category of research method. Its two categories -- random or systematic research -- are too coarse-grained to clearly reveal shortcomings of existing research. While several other frameworks of CASE environments exist, they focus on outcomes of the research such as functionality and life cycle phases supported [cf. 12], and are not considered here.

Scott Morton [28] presents the finest grain scale of research methods. The methods included in his evaluation of management support systems research are prototype, methodology, theory, concept, empirical laboratory test, empirical real-world test, survey, case and truth. These methods, while expressed in slightly different terms than used in this report, cover the breadth of research methods available. Thus, Scott Morton's categories for research method are adapted slightly and used in this classification. In addition, action research, which can take place in field or laboratory settings, is added to this list. The research methods are defined and discussed in the next section.

The second dimension of the classification scheme was determined by considering the dimensions of research classifications in the research cited above. Only Basili et al. [1] in their multi-dimensional framework for evaluating software engineering research include research purpose. Basili's categories are adapted for use here. The categories used in this classification are defined and examples given in the section on Research Purpose.

Although these two dimensions are used for this classification scheme, it is not meant to be a complete framework, but rather, the first

two dimensions of an evolving framework. In the next section, the research methods are defined, and examples of their use in CASE research are discussed. Then research purpose is identified and sub-categories are defined.

3. RESEARCH METHODS

In this section, ten research methods are reviewed: case study, laboratory experiment, simulation, field experiment, field study, action research, sample survey, applied, basic, and normative research. For each method, underlying assumptions and goals are identified. Then, the applicability of the method to CASE research is discussed, including examples of published and potential research. The discussion is summarized in Table 1.

Benbasat's [2] framework discusses research environments of management support system research (MSS), including natural, artificial, and environment-independent. Since his classification is consistent with others from organization research [29] and IS research [28], it is used in this report to simplify the discussion.

3.1 Natural Setting Research

The four natural setting methods -- case studies, field studies, field experiments, and action research -- are used to study some aspect of organizational life in real organizations. Each of these methods are discussed in this section. The common feature of these methods is that they are all conducted in actual business, government, or non-profit organizations. Action research is included in this section because, although it can be conducted in either natural and artificial settings, it is most often conducted in a natural setting.

3.1.1 Case Study

Case studies are intensive evaluations of small samples of entities such as groups, organizations, individuals, systems, or tools. Data are collected by multiple means that may include interviews, observation, questionnaires, computerized data, and written materials. No controls of experimental or statistical phenomena are attempted. The main purposes of case studies are explanation, description, and hypothesis generation (See Table 1).

The major advantage of case studies is that changes and processes over time can be analyzed. The best case studies use a grounded theory approach to guide the research. Findings consistent and inconsistent with the theory are considered and interpreted during the analysis.

The major drawbacks of case studies are high cost, time, and limited generalizability of the findings. Replicated case studies may be used to generalize results. Cost and time are inherent in the method.

Case studies are appropriate classification for CASE research, since CASE research and theory are in an early stage of development [3, 30]. The strength of case studies is that they provide rich, explanatory evidence that can be used to explain why and how phenomena occur. Extensive case

Method	Strengths	Weaknesses	Major Uses
Case study	Process understanding Demonstrate causality Natural Setting Rich data	Cost, time Limited generalizability	Explanation Descriptions Hypothesis development
Field Study (Quasi- or real Experiment)	Natural Setting Control Ind. Variables Replicable	Hard to find sites Experiments may lose naturalness Unknown sample bias No guarantee of IV variation Recall/self report reliability	Post hoc study of processes & outcomes in practice Hypothesis development & testing
Action Research	First Hand Experience Apply theory to practice Closer relationship with subjects	Researcher bias Time Ethical Issues Unknown generalizbility	Theory, hypothesis testing Generate hypotheses Identify variables
Lab experiment	High reliability Greatest variable control Ind. Variable manipulation Replicable Precise measures	Unknown generalizability to real settings or real people Artificial Setting Assumes real-world not important	Experiments requiring high degree of researcher control over setting, variables, and subjects Theory testing
Field surveys	Easy, Low Cost Can reduce sample bias	Context insensitive Static picture No variable manipulation	Large samples Hypothesis development & testing Collect descriptive data
Applied research	Goal is a product which may be evaluated	Solution may be constrained by goals May need further design to make product general, adaptive, context-free	Product development Hypothesis testing Proof of Concept
Basic research	No restrictions on solution Solve new problems	Cost, Time May be no solution	Theory building
Normative writings	Insight of firsthand experience Basis for other forms of research	Opinion	Descriptions of practice Frameworks Inform

Table 1.
Summary of Research Methods

studies at Xerox PARC identified the "office" model on which the Star system, Macintosh and other object oriented environments are based. Facets of project work, such as user interactions and group communication were studied to develop that model [cf. 13]. Case studies could be used in CASE research to study, for instance, tool appropriateness in a given organizational context, the effect of individual differences in use of a tool, or impacts of a tool on organizational members [cf. 63]. Case studies would be particularly useful for studying trends, overall group methods, and a combination of individual and group processes that comprise the 'how' of project development.

3.1.2 Field Study

Field studies are an *ex post facto* method of evaluating changes in organizational systems based on recall and self-report of participants. Field studies are non-experimental in nature since no specific manipulation of independent variables occurs. These studies evaluate strength of relationships for hypothesized models. Such studies are high on realism and low on obtrusiveness. The major disadvantages of field studies are unknown sampling biases and no guarantee of variation in independent variables. Field studies may be used to describe how a system works, or to generate and test hypotheses.

Field studies can yield results in a relatively short period of time. Experiences of different organizations with the same tool can be studied in detail through interviews. The tool preferences of different user groups can be studied to identify the set of characteristics the "ideal" environment should have. This might be contrasted with the collective wisdom of normative CASE research.

An example of a field study in CASE research is, for instance, a work-in-progress evaluating CASE implementation strategies in organizations [33]. The research can determine the success of different methods used to implement the same CASE tool. When different software implementations use the same strategies successfully, normative practices can be developed for managers.

3.1.3 Field experiment

Field experiments are characterized by experimental manipulation of one or more independent variables while contaminating variables are controlled to observe the effect on dependent variables, all in a natural setting [29]. They may test hypotheses based on a grounding theory. The major advantages of field experiments are first, increased reliability of results over field studies because of increased control over variables. and , second, support for testing of complex social interactions and processes including measurement of change. The major disadvantage is that as experimental manipulation increases, the naturalness of the experiment decreases. In addition, the researcher has to guard against insufficient control over possible contaminating effects.

A field experiment relating to CASE might evaluate the introduction of the same CASE tool into several organizations which use it to develop

identical applications. The impact of CASE on development time, productivity, the quality of the resulting system, and on individual behavior and motivation could all be evaluated. A second type of experiment might evaluate differences in the same application developed using different CASE tools. The major hurdle to this type of research is not inherent research difficulty, but finding and coordinating multiple sites.

3.1.4 Action Research

In action research, the researcher conducts research while participating in the intervention or activity studied, simultaneously evaluating the results. Such participation can take place as a sub-context for case, field study, or laboratory study research. An action field experiment might analyze the differences in process and product using the different methodologies. The researcher might conduct the training in application development methodologies for different analysis groups, then act as an analyst/consultant on one (or more) project(s).

There are many advantages to action research. The researcher, as part of the team, is required to be present and gains a first-hand understanding of the situation. The researcher is not viewed as interfering with the process, and can get closer to the other participants. In any field setting, this is particularly useful because the subjects forget they are subjects, forget the researcher is present for a different reason, and are more honest and straightforward in their dealings with the researcher. Finally, having a researcher/participant provides a particularly rich way of applying theory to practice and evaluating its worth.

One problem with action research experiments is ethics. If one truly believes that a particular treatment has a more desirable effect, then for ethical reasons, all groups should have that treatment.

Since the researcher has a stake in the success of the outcome, another problem of action research is lack of objectivity. Interpretation of results is necessarily filtered through the eyes of the researcher participant, who may lose his/her objectivity. Finally, it may be difficult to generalize the results of action research beyond the context in which it occurs [3].

Although no examples action research for studying CASE could be found, action research would be useful in the same situations in which case, lab or field studies are appropriate. A case study involving a participant for the development and implementation of an application would provide useful insights on how and why different features of a product are used (or not used). From a group work perspective, we might learn more about how group work is hindered, facilitated or changed through the use of CASE tools. A field study in which the researcher participates comparing two groups: one with a CASE tool and one without, could provide insights into process and product differences that might not be identified through a case study, e.g.

3.2 Artificial Setting Research

Research conducted in artificial settings includes laboratory experiments and simulations. In these methods, the environment is established by the researcher specifically to bound the experiment. All artificial methods may be used to test hypotheses, but the generalizability to non-artificial settings is difficult to justify without additional research using other methods.

3.2.1 Laboratory experiment

Laboratory experiments are characterized by a researcher-created setting with experimenter control over assignment of subjects to experimental treatments, treatment variables, and the manipulation of treatment variables [29]. In the CASE context, a lab experiment might entail systematic variation of tools/methods to minimize outside interference from environmental interactions. Advantages of laboratory settings are more precise measurement of the phenomena of interest when compared to field methods, and enhanced replicability.

There are several major disadvantages of the method. First, the method assumes that real-world interference is not important to the events being modeled. In a CASE context, this assumption may cause disregard for such phenomena as implementation politics, development processes and user interactions taking place over time. Second, generality is limited to the sample population. The enhanced replicability of lab experiments can be used to make findings more general by replication with multiple, different samples.

In computer-oriented research, there are other inherent disadvantages to this method. If users are novices with the tool/method, the method assumes that newness is not a factor in use. If users are experienced with the tool or method, this research method assumes that past knowledge is equal across participants and is applied the same by all participants. Consideration of such individual differences becomes an important part of planning for CASE research using laboratory experimentation.

Laboratory experiments might be used to evaluate productivity impacts of tool use or non-use, ease of tool use, interface design alternatives, or other tool features. In addressing what we do not know about CASE environments, laboratory experiments could be used to determine, for instance, exactly what "seamless" integration means to users.

While traditional statistical methods of evaluating laboratory data can be used, one unique method of analyzing laboratory experiments is protocol analysis. In protocol analysis, a subject or group is asked to 'think aloud' during the performance of the experimental task. A coding scheme is devised to identify the possible events of interest to the research. Then, both counts of the events and individual events themselves are analyzed to interpret the data. Protocol analysis can be used to study, for instance, underlying mental models present in software development, or group communication during the development process. Results of such studies

could be used to determine what features should be included in a new CASE tool, to evaluate tool features that facilitate or hinder thought processes, or to define features supporting collaborative application development for inclusion in a CASE tool.

3.2.2 Simulation

There are two types of simulation research -- organizational and programmed. Organizational simulations use created settings to replicate the attributes of real situations. Participants are exposed to researcher-designed "real world events" and react freely within the constraints of the simulated situation. Role playing is one form of simulation. Advantages of this type of simulation are that realism and subject involvement tend to be high. The major disadvantages are high cost and difficulty in identifying variation in variables.

Programmed simulation automates some aspect of a real world situation and allows automated sensitivity analysis of perturbations in the simulated environment. The advantages and disadvantages of automated simulation are the opposite of those above for organizational simulation. Automated simulations are lower in cost with easily identified, user-controlled variations. The major disadvantage of automated simulation is that it requires parameterization of context that may narrow the problem domain so much that its "real world" qualities disappear.

Both methods of simulation are applicable to CASE research. Organizational simulation can be used for determining the extent to which multi-user, multi-level collaborative work is necessary and useful when automated. It is assumed, from organization communication research, that some communications will always be needed in face-to-face meetings, telephone conversations, and telephone conferences. The usefulness of automated tools for walk-throughs, requirements presentations, and meetings between geographically dispersed participants is assumed but should be demonstrated before prototypes are developed. Other outcomes of such simulations might be characteristics of different forms of information presentation.

Automated simulation has been used in testing real-time systems, e.g., evaluating message queuing efficiency. It could also be useful in identifying how much automation of application testing is possible and how difficult automating test processing would be. Another possible application of the method is the simulation of AI routines' learning capabilities with respect to application history and the use of, for example, automated testing.

3.3 Environment-Independent Research

Three methods in the environment-independent category assume that participants' responses, the process studied, and the product of the research are not affected by the setting. The research in this category includes surveys, basic and applied methodologies.

3.3.1 Sample Surveys

Sample surveys use information from a known population gathered via some systematic technique, usually questionnaires or interviews. No independent variables are manipulated. Responses are collected directly from respondents and assumed to be unaffected by context. Confounding influences are measured and controlled statistically. The advantages of sample research are that large amounts of data can be collected relatively easily and cheaply in a variety of settings. If properly chosen, samples can be selected to reduce bias and allow for generalization of results to broader populations. The major disadvantage of surveys is the assumption that snap-shots of differences in processes across a number of settings is analogous to differences in processes over time within settings. To eliminate this assumption requires triangulation of results with different methods, or longitudinal data gathering from the same sample. A less critical disadvantage is respondent unwillingness. By making response voluntary, the risk of sample bias increases. Involuntary compliance raises the risk of inaccurate answers.

Sample surveys are useful in CASE research, but are more limited in their usefulness than other methods because they ignore processes and time. Such research might identify industries adopting CASE first, and the problems, plans, and tactics used in those industries that facilitated the use of the tools in their environments. From a pedagogical perspective, research identifying which CASE tools are most popular in industry is helpful in determining what to teach and/or demonstrate in information systems courses.

Surveys in CASE research evaluate the relative usage of commercial CASE tools, preferences for tools and tool features, and the perceptions of productivity impacts of various tools or tool features [cf. 62, 72]. These surveys are discussed further in the Results section.

3.3.2 Applied Research

In applied research, intuition, experience, deduction and induction are used to analyze a specific research problem for which the desired outcome, but not the method of obtaining the outcome, is known [4]. There are frequently time or budget pressures in finding the desired outcome. The approach to solution finding, like basic research below, is trial and error based on the expertise and reasoning capabilities of the researcher. The difference is that a specific goal is known. The major disadvantage of this method is that an initial solution may not be elegant, generalized, easily adapted, or even context-independent. The major advantage of applied research is that, while giving free rein to the researchers, the research is goal-directed and the usefulness of an end product is able to be evaluated.

Applied research for the development of environments, environment fragments, and enhancements to existing environments has been the main focus of CASE research [cf. 43, 47, 50, 51]. An example of applied research is the development of an adequate database architecture for storing, retrieving, and reusing the meta-information of application development.

Bigelow [36] discusses the application of hypertext in the development of CASE repositories, while Hull and King [17] develop a specialized form of relational databases.

3.3.3 Basic Research

Investigators doing basic research develop new theories or perform research in which the problem is known but no specific method for a solution and no specific solution are known. The approach to solution finding is trial and error based on the expertise and reasoning capabilities of the researcher. The major advantage of basic research is that there are no preconceptions about how a solution should come about. Typically, there are no time pressures. The disadvantage of basic research is that it is a slow, costly process that may not lead to any solution. Most basic research tends to be isolated with little information or idea sharing between different research organizations or, even, researchers.

Basic research could be applied to some fundamental CASE and application development problems, for instance, understanding the dynamics of application development to develop a theory of software engineering.

3.4 Normative Writings

There is a significant non-research body of writing about the nature of CASE environments. Scott Morton [28] describes such writing as either concept development or "truth". Concept development writing attempts to organize ideas through frameworks and/or present new perspectives to existing ideas to stimulate research and action. This paper would fall in the conceptual development category. The "Truth" category identifies suggestions and presentations of ideas, not based in theory, that intuitively seem correct. Osterweil's [25] definition of five minimal characteristics of software development environments, McClure's [23] discussion of CASE experiences and ideal CASE environment components, and Pressman's [26] list of ideal CASE requirements are all examples of Truth normative writing. We collapse both these features into a category called "Normative".

In this category, we also include what Benbasat [3] terms "application descriptions". These are narratives written by practitioners describing the author's description of a situation, generally describing "what worked for us". The author's original intention is not to conduct research; the description of what was done is written up after the fact. An example of this in CASE research is Davis' [9] discussion of the introduction of PSL/PSA into an Army installation.

A second application description details the experiences of one organization in the use of a CASE tool [52]. The study is interesting because it describes management doing all the "right" things and the subsequent failure of the tool for that organization because of the tool's technical difficulties. While the generalizability of these two cases is limited, they provide an insight into the notion that not all benefits from CASE are positive. They both indicate a need for future research to determine the

organizational, tool and user factors that contribute to CASE success and failure.

3.5 Summary

Ten types of research methods, examples of CASE research using each method, and their applicability to CASE research needs were discussed in this section. Initial attempts to classify research demonstrated many empty cells. Two sets of research methods are collapsed to avoid numerous empty cells in the classification: field experiments and field studies are combined to create Field Research; laboratory experiments and simulations are combined in Laboratory Research. The final research categories in the framework include Case, Field, Laboratory, Action, Survey, Applied, Basic, and Normative.

In the next section, research purpose is defined as the second dimension of the classification scheme.

4. RESEARCH PURPOSE

The goal or purpose of a research effort both determines, and is determined by, the research method used. Because these two notions are so closely coupled, we use purpose as the second dimension for classifying CASE research. Borrowing from Basili, et. al [1], the categories in research purpose include understanding, engineering, re-engineering, evaluation, and description. Each of these categories is discussed in this section.

Understanding describes a category for definitions that allow one to grasp the meaning of the entity studied, for instance, CASE. Thus, frameworks, which attempt to categorize for the purpose of providing understanding, are in this category.

Two categories are used to classify the development of CASE tools, while allowing identification of evaluation efforts which should separate the two activities. Engineering is the original development of a prototype or CASE environment. Re-engineering is the re-development of an existing environment or environment fragment. Ideally, evaluation takes place between the engineering and re-engineering efforts, and is the basis for changes during the re-engineering effort.

Evaluation, which includes assessment, validation, and assurance actions, is the systematic study of a CASE product(s) to determine its usefulness and/or affect. Evaluations can be individual studies to determine if some feature works; can compare a product with another product; or can study large groups of users to determine if some feature(s) have some properties such as usefulness, ease of use, etc.

The description category refers to research that defines and/or describes features of "ideal" CASE environments. Features discussed in such research include integration across phases of a life cycle, coupling to a development methodology, and human interface design [cf. 40, 73].

Intentions to manage, learn, or improve an environment imply improvement of the software engineering process as a result of use of the CASE environment under discussion. Improvement of the work process

would seem to be an underlying assumption of all CASE development. This category in Basili [1] is not included in this study because it requires a value judgment about whether or not improvement is really intended.

The remaining research purpose categories in Basili are prediction and motivation. No research to date has been done with the intent to specifically predict or motivate action, so those categories are omitted in this discussion. While they might be useful categories in a future framework, they are not relevant here.

In this section, we defined the categories selected from Basili [1] to describe CASE research purpose, including understanding, engineering, re-engineering, evaluation, and characterization. These categories, along with the research methods, are used in the next section as a basis for classifying and discussing CASE research from the last three years.

5. CASE RESEARCH CLASSIFICATION

This section evaluates research on CASE environments which support most or all of the life cycle activities, including requirements definition, conceptual design, detailed design, programming and unit testing, and system testing. We specifically omit programming environments that address only program implementation activities. To combine both categories of environments in one paper skews the results toward applied research without adding additional insight.

This paper is an initial application of the framework, therefore it does not represent an exhaustive review of the literature. Research articles were randomly selected from CASE conference, IEEE, and ACM publications between 1987-1989. Forty of 160 articles identified are classified. Research publications on the development of commercial products were specifically omitted from this classification. However, several research evaluations of CASE products in this survey compare or evaluate commercial products.

In Table 2 the eight research categories identify the columns. As the table shows, 47.5% (19 of 40 studies) of the research is accounted for by the applied method. The second most populated research method is normative, accounting for 32.5% (12) of the total. The remaining 20% is spread over case (10%), survey (7.5%), and laboratory (2.5%) research. There are no entries for field studies, action research, or basic research. Clearly, some research using these methods is required. An increase in research in the case, survey, and laboratory categories also seems warranted.

The five categories identifying research purposes describe the rows of Table 2. Again, while no rows are empty, 45% of the work is some form of engineering or re-engineering. All of the re-engineering entrants discuss shortcomings of their original environments to be alleviated with the re-engineering effort, but none include any evaluation of the environment by people other than the designers.

Such evaluation of environments has been limited. One research team omitted from the table because they were not in the sample, does build and evaluate an environment. Boehm et al, [6] have been developing the SREM and REVS environments for the last decade. Upon completion of

	Research Method								
	Case	Field	Lab	Action	Survey	Applied	Basic	Normative	Totals
Purpose:									
Understand								38, 41, 42, 64	4 10.0%
Engineer						36, 45, 46, 50, 51, 53, 54, 55, 56, 57, 60, 65, 67, 68, 70			15 37.5%
Re-Engineer						37, 43, 71			3 7.5%
Evaluate	35, 39, 59, 63		34		62, 69, 72			52	9 22.5%
Describe						44		40, 47, 48, 49, 58, 61, 66, 73	9 22.5%
Method Totals N	4	0	1	0	3	19	0	13	40
Percent	10.0%	0%	2.5%	0%	7.5%	47.5%	0%	32.5%	100%

Table 2.
Classification of CASE Research

Legend: nn = The sequence number in the Classified Research Bibliography

each major activity, an evaluation of the functionality, usefulness, and design characteristics of the tool/environment is performed [cf. 7]. This work should serve as a guideline to other researchers on how to perform evaluations of completed work.

The evaluation row has 22.5% (8) of the table entries. One of the case studies was a rigorous evaluation of the use of one CASE environment in different development groups in one organization [63]. The research was theory based, evaluating changes accompanying the use of a CASE tool. One contribution of the work is the identification of constraining capabilities of CASE.

Three of the four case studies used some methodology or environment that is not commonly used and/or known [35, 39, 59]. Because of the low probability of the subject methodology or environment ever becoming common, the potential importance of these studies lies in the development of general conclusions about methods or tools. Unfortunately, no conclusions like this were developed, therefore the results of these cases are of limited interest.

The laboratory research describes simulation of environment results to test correctness, and is interesting both for the approach and for its results [34]. The environment being evaluated was developed over six years ago and has undergone a series of refinements. The research is an automated simulation of real-time event synchronization. Both successful and unsuccessful results are analyzed to develop the next iteration of re-engineering changes to the environment.

The survey entries fall into two categories: user perception and tool comparison. Norman and Nunamaker [62] and Wynekoop and Senn [72] survey practitioner perceptions of commercial products. The studies describe CASE features that need redesign based on how real people use the environments. Troy [69] surveys four practitioners to compare features of several tools. The small sample size is a distinct weakness of the research, but it shows how such an evaluation might be performed. With a larger sample size, such feature comparison would be useful in identifying which design aspects of features are actually used and which are not used.

Empty cells account for 77.5% (31 of 40) of Table 2 cells. Only 15%, or six, cells contain three or more entries. Although some of the cells are not likely to be populated, e.g., case study engineering, the results might be interesting. The lack of basic, action and field research is particularly disappointing. Each could provide a valuable contribution when we really do not understand what it is that we are automating.

This section identifies applied engineering as the most frequently used research method, with normative writings second most frequent. The problems with the proliferation of only these two types of research are discussed in the following section.

6. DISCUSSION

The classification in the preceding section highlights the lack of evaluation of CASE research along with the proliferation of both normative descriptions of CASE, and applied research for both original and re-

engineering environments. The most significant problem with CASE research identified by evaluating the studies in the previous section, is that there are significant implicit assumptions in each CASE environment that are not dealt with by research to date. This section discusses these assumptions to determine their potential impact on the field and how to overcome them.

Three assumptions prevail in CASE research: 1) CASE is preferred to alternatives, 2) we know what CASE should do, and 3) methodology does not matter. One major assumption of most CASE research[2] is that some automation is better than none. There is a refusal to assume that the tool or environment may not be useful and may not enhance productivity. This bias is side-stepped in CASE research by concentrating on what the environment does rather than on how well it does it, or on what the environment does not do, or on the unintended effects of its actions. The evaluation research in Table 2, focuses on functional analysis of the specific software under consideration rather than on contextual issues including, for instance, the extent to which the environment facilitates, hinders, and/or changes the work [cf. 34, 35, 39, 52, 59, 69, 72]. To eliminate this pro-innovation bias, evaluative research is needed. Sample studies might compare automated and non-automated development, compare multiple automated CASE tools, analyze one project's use/non-use of a product, or analyze multiple project use of a product.

The second assumption is equally troublesome to accumulation of knowledge about CASE. Researchers do not address the fact that there is no consensus on what is, and what is not, software engineering. Recognized experts [e.g., 5, 10, 18], all have similar, but different definitions for what constitutes a life cycle and the software engineering process within it. If we cannot agree on a definition of the activity, how can we automate it?

Without some consensus on these fundamental activities and processes, comparison of CASE environments is impossible. Without comparison, we reduce our ability to determine what features, functions, activities, and processes are really needed in CASE tools.

Further, there are over 60 methodologies falling into five major classes for analysis of application requirements[6] [8]. A CASE tool can only support one method in one class. Yet, which one is best? Or, do we want customized methodologies? The simple answer is that we do not know.

The applied research that only builds environments and never evaluates them, is getting ahead of our knowledge. Research does a disservice to potential comparative analysis when it fails to define terms, goals of the research, and, if a product is built, degree of coupling to a methodology, user characteristics, phases of development life cycle supported, and the underlying definition of support[3]. To solve these problems, we need more research on what analysts do when they analyze. We need methodology evaluations within and between classes to determine which are truly useful and which can be forgotten. We need consensus on

terminology and definitions. In short, we need a theory of application development.

The final assumption of most academic applied CASE research is that methodology[4] does not matter[5]. This may be appropriate for exploring problems in a narrow area, such as interface design or data storage structures, but it yields weaker application development environments than those coupled to a methodology. Although they may provide tools automating a technique such as data flow diagraming, CASE not coupled to some methodology cannot enforce a management framework, rules, or standards of a methodology, nor can they assist in identifying steps that have been missed or steps that should be done next in development. Thus, although these methodology-free tools automate routine and tedious tasks, they cannot provide process, structure, quality assurance, or guidance to the user.

The research needed to determine how to integrate methodology into CASE includes all possible research paradigms. Sample research might include basic and applied research on meta-data definition that supports customized methodology definition, action research to determine how methodologies are used (and not used) in real projects, surveys to determine the state of the art in business and the magnitude of training problems, lab studies to analyze how methodologies can and cannot support analysis and design work.

This brings back the question of why, if we need all of this research, do normative and applied research proliferate? One simple answer is that these types of writing are easiest. Normative writing is based on personal opinion, observation, and summarization of others' thoughts. Applied research attempts to derive a new approach to solving a known problem. Neither set of writings requires theory or theory development, although both could benefit from it.

Applied research is not the most appropriate way to alleviate our lack of knowledge on computer aid or software engineering or CASE. From a cognitive psychology perspective, applied research is viewed as a weak problem-solving method [15, 20]. The use of trial and error, generate and test, and means-end analysis are all considered weak methods of problem solving that are "very taxing cognitively and ... often translate into poor performance because they require search of a large space of possibilities" [15].

Research need needed to eliminate these assumptions and the accompanying problems include all types. Table 3 shows one topic that could be addressed by each research method. There are infinitely more studies possible in each cell. And, of course, it is desirable to replicate and triangulate results using multiple methods. Case study and action research to analyze details of processes over time; field studies to compare methods, tools, techniques; lab experiments to simulate and analyze learning curves, domain knowledge importance, and individual differences between novices and experts; field surveys to determine what features, functions, and interface designs real users *want* in a CASE tool; applied research to continue to build *and evaluate* tools, tool fragments, and new

Purpose:	Research Method: Case	Field	Lab	Action	Survey	Applied	Basic	Normative
Understand	Process of introducing CASE in one organization	Importance of communication and training in CASE implementation	The importance of domain knowledge in use of CASE	Same as case study with researcher active in implementation	Industry adoption differences in CASE	Prototype a new interface based on theory of communication processes	Develop theory of how analysts analyze	Develop framework of CASE research
Engineer	Study engineering *process*; compare to other software work		Simulate productivity and payback (time and cost) for different CASE features	Compare multiple implementation techniques e.g. object vs. rule-based	Survey engineering practices across CASE industry, e.g. methods used	Develop CASE tool	Prototype of developed theory on DBMS relating to group analysis	
Re-Engineer			Compare multiple implementation techniques	Compare multiple implementation techniques	Study industry design changes to determine convergence on *ideal* CASE criteria	Change feature(s) of existing CASE tool	Alter existing environment to add new AI, ESS, database, etc. findings	Develop guidelines for re-engineering to meet ideal CASE criteria
Evaluate	Study usage to determine quality, usefulness, etc. of one CASE tool	Evaluate Innovation Theory as it applies to CASE	Evaluate novice/ expert differences in using CASE tool	Evaluate, while participating in, different training techniques for CASE users	Study features of CASE used across different companies	Use expert system to evaluate usage of CASE tool for collaborative work	Build prototype to test theory	Unscientific evaluation of what worked/ failed in one company with prescriptions for others
Describe	Description of what worked/ failed in one company in CASE use		Problems in learning to use a CASE tool ... relate to learning theory	Systematic description of personal experience in using CASE tool	Which CASE tools are used in industry	Description of decisions and design of engineered CASE tool	Lab notes	Ideal characteristics of CASE

Table 3.
Sample Research Projects Using Different Research Methods

methods; basic research to develop domain independent methods, technology independent methods, or other novel perspectives on building applications; normative writing to provide a basis for discussion, exploration of concepts, evaluation of potential components, and derivation of consistent definitions -- all are needed.

7. CONCLUSION

This paper examines the paradigms of CASE research by classifying a sample of of CASE publications from the last three years. The classification, by research methodology and purpose, identifies the applied research method as most frequently used with normative writing as the second most frequently used method. The most common goal of research is to build or re-build CASE environments.

There is little evaluation of applied research efforts, and most evaluation research is narrow in scope. Other research methods are not frequently used. Basic, action and field research are not found at all in this sample from the last three years. There is very little case study, laboratory, or field survey research. Few evaluative research studies are in the sample; most CASE developed through the applied method is never formally evaluated or compared to possible alternatives environments.

Researchers must overcome the applied bias in CASE research to gain more cumulative knowledge. As researchers we can do several things. First, we must be more cognizant of the research paradigm, and be specific in addressing limitations due to selection of a specific research paradigm. Second, we need to evaluate every engineering and re-engineering effort to determine what, if any, progress toward understanding computer-aided software engineering is made. Third, we need to conduct a richer body of research using varied methodologies and multiple methodologies to allow triangulation of results. Finally, we need to conduct research in both software engineering and CASE to move toward a theory of software engineering that would inform all future research and CASE development.

FOOTNOTES

1. Computer aided software engineering is the automation of the software engineering discipline. Software engineering is the application of "principles, skills and art to the economical development of programs that run reliably and efficiently on real machines" [31].

2. Notable exceptions, such as Orlikowski [63] are rare.

3. Support might mean 'is able to build an *xyz* diagram'; it may also mean 'is able to build an *xyz* diagram that is validated, leveled, consistency and completeness checked automatically'.

4. Methodology refers to a specific set of procedures, graphic representation forms, and documentation standards for application development. e.g., Structured Methods of DeMarco [11].

5. Again, there are exceptions. Writing on SREM [5], for example, consistently incorporates that methodology in automated tool development.

6. Process, data, object, social and semantic [8].

REFERENCES

1. Basili, V.R., Selby, R.W., & Hutchins, D.H. (1986). Experimentation in software engineering. IEEE Transactions on Software Engineering, SE-12, 733-743.

2. Benbasat, I. (1985). An analysis of research methodologies. In F.W. McFarlan (Ed.), The Information Systems Research Challenge (pp. 47-85). Boston: Harvard Business School Press.

3. Benbasat, I., Goldstein, D.R., Mead, M. (1987). The case research strategy in studies of information systems. MIS Quarterly , 7(3), 369-386.

4. Beveridge, W.I.B. (1957). The Art of Scientific Investigation. New York: Vintage Books.

5. Boehm, B. W., (1981), Software Engineering Economics, Englewood Cliffs, NJ: Prentice-Hall.

6. Boehm, B. W., Penedo, M. H., Stuckly, E. D., Williams, R. D., & Pyster, A. B. (1984). A software development environment for improving productivity. Computer, 17(6), 30-44.

7. Childs, D., Whalen, D., & Keith, L. (1983). U.S. Army Intelligence Center and School (USAICS) Software Analysis and Management System (USAMS) evaluation of SREM/REVS/RSL and PSL/PSA. In Proceedings of the Symposium on Application and Assessment of Automated Tools for Software Development, pp. 89-98.

8. Conger, S.A, Russo, N. (1990) A Taxonomy of Applications: A Framework for Selecting and Evaluating Methodologies, Georgia State University Working Paper #90-001.

9. Davis, J. (1983), Transfer of Automated Requirements Test Technology, Proceedings of the IEEE Workshops on Software Engineering Technology Transfer, Silver Springs, MD: IEEE Press, pp 30-33.

10. Davis, G. B., and Olson, M. H. (1985), Management Information Systems: Conceptual Foundations , Structure and Development, NY:McGraw-Hill.

11. DeMarco, T. (1978). Structured Analysis and System Specifications. New York: Yourdon Press.

12. Fedchak, E. (1986). An introduction to software engineering environments. In Proceedings of COMPSAC 86 (pp. 456-463). Washington: Computer Society Press.

13. Goldberg, A. (1984), Smalltalk-80: The Interactive Programming Environment Reading, MA:Addison-Wesley.

14. Gorry, G. A., Scott Morton, M. S. (1971), A Framework for Management Information Systems, Sloan Management Review, Fall, pp 55-70.

15. Guindon, R., Krasner, H. & Curtis, B. (1988). Breakdowns and Processes During the Early Activities of Software Design by Professionals. Austin, Tx:MCC.

16. Hausen, H. & Mullerburg, M. (1981). Conspectus of software engineering environments. In Proceedings: 5th International Conference on Software Engineering (pp. 34-43).Washington: Computer Society Press.

17. Hull R. and R. King, (1987), "Semantic Database Modeling: Survey, Applications, and Research Issues", ACM Computing Surveys, Vol. 12, #1, March, 1987, pp 201-260.

18. IEEE (1983), IEEE Software Engineering Dictionary, Piscataway, NJ:IEEE Press.

19. Ives, B., Hamilton, S. & Davis, G.B. (1980). A framework of research in computer-based management information systems. Management Science, 26, 910-934.

20. Laird, J., Rosenbloom, P., & Newell, A. (1986). Universal Subgoaling and Chunking, Kluwar Academic Publishers.

21. Leitheiser, R.L. (1986). Computer support for knowledge workers.: A review of laboratory experiments. Data Base, 17(1), 17-45.

22. Mason, R.O. & Mitroff, J.I. (1973). A program of research on management information systems. Management Science, 19(5), 475-487.

23. McClure, C. (1989), CASE is Software Automation,Englewood Cliffs, NJ: Prentice-Hall.

24. Nolan, R.L. & Wetherbe, J.C. (1980). Toward a comprehensive framework for MIS research. MIS Quarterly, 14(2), 1-13.

25. Osterweil, L. (1981). Software environment research: Directions for the next five years. Computer, 14(4), 35-42.

26. Pressman, R. (1988), Making Software Engineering Happen: A Guide for Instituting the Technology, NY: McGraw-Hill.

27. Reps, T., and T. Teitelbaum, The Synthesizer Generator.In Software Engineering Notes, Vol. 9, #3, May, 1984.

28. Scott Morton, M. (1985). The state of the art of research. In F.W. McFarlan (Ed.), The Information Systems Research Challenge (pp. 13-41). Boston: Harvard Business School Press.

29. Stone, E. (1981), Research Methods in Organization Behavior, Chicago, IL:Scott-Foresman.

30 Swanson, E.B.& Beath, C.M. (1988). The use of case study data in software management research. The Journal of Systems and Software, 8, 63-71.

31. Wasserman, A.L. (1975), A Top-Down View of Software Engineering, In Proceedings of the 1st National Conference on Software Engineering, Washington, D. C., September 11-12, 1975.

32. Weber, R., (1984), Toward A Theory of Artifacts: A Paradigmatic Base for Information Systems Research, Draft Paper, University of Queensland, April 1984.

33. Wynekoop, J.L. (1989). The Impact of Perceptions and Expectations of CASE Tools on Implementation. Unpublished manuscript. Atlanta: Georgia State University.

CLASSIFIED RESEARCH BIBLIOGRAPHY

34. Aggarwal, S. , Barbara, D., & Meth, K.Z. (1988). A software environment for the specification and analysis of problems of coordination and concurrency. IEEE Transactions on Software Engineering, 14, 280-290. (A1)

35. Amundsen, B. & Christoffersen, B. (1987). Can todays design tools support an integrated design method? In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 327 - 347. (A2)

36. Bigelow, J. (1988). Hypertext and CASE. IEEE Software, 5(2), 23-27. (B1)

37. Blum, B. I. (1987). The Tedium Development Environment for information systems. IEEE Software, 4(1), 25-34. (B2)

38. Bostrom, R., Anson,R. (1988). Using computerized collaborative work support systems to improve the logical systems design process. Proceedings of the 1988 ACM SIGCPR Conference on the Management of Information Systems Personnel, pp. 98-108. (B3)

39. Carasik, B. (1987). CASE tools: Illusion and reality. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 194-199. (C1)

40. Chen, M., Nunamaker, J.F., & Weber, E.S. (1989). Computer-aided software engineering: Present status and future directions. Data Base, 20(1),7-13. (C2)

41. Conger, S.A & Wynekoop, J.L. (1989) A framework for evaluating computer-aided software engineering research and products. In N.S. Coulter & E. Ellie (Eds.). Proceedings of the 27th Annual Southeast Regional Conference (pp. 201-205). New York: ACM Press. (C3)

42. Dart, S.A., Ellison, R.J., Feiler, P.H., & Habermann, A.N. (1987). Software development environments. Computer, 20(1), 18-28. (D1)

43. Fisher, G. (1988). An overview of a graphical multilanguage applications environment. IEEE Transactions on Software Engineering, 14, 774-786. (F1)

44. Freeman, P. (1987). A conceptual analysis of the Draco approach to constructing software systems. IEEE Transactions on Software Engineering, 13, 830-845. (F2)

45. Giacalone, A. & Smolka, S. A. (1988). Integrated environments for formally well-founded design and simulation of concurrent systems. IEEE Transactions on Software Engineering, 14, 787-801. (G1)

46. Harel, D., Pnueli, A., Politi, M., Sherman, R., & Shtull-Trauring, A. (1987). The AD CAD methodology and STATEMATE1 working environment. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 376-383. (H1)

47. Hecht, A. & Simmons, A. (1988). The automation of structured analysis and structured design. In Proceedings of the 6th Annual International Phoenix Conference on Computer and Communications (pp. 267-271). Washington, D. C.: IEEE Computer Science Press. (H2)

48. Kaminsky, A. (1987). Capabilities for a computer-aided software enginnering environment. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 44-46. (K1)

49. Kearns, M. (1987). Capabilities for CASE tools: now and the future. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 47-49. (K2)

50. Kishida, K., Katayama, T., Matsue, M., Miuamoto, I., Ochimizu, K., Saito, N., Saylor, J. H., Torii, K. &. WIlliams, L. G. (1988). SDA: A novel approach to software environment design and construction. In Proceedings of the 10th International Conference on Software Engineering (pp 69-79). Washington, D. C.: IEEE Computer Science Press. (K3)

51. Kramer, J., Ng, K., Potts, C., & Whitehead, K. (1988). Tool support for requirements analysis. Software Engineering Journal, 86-96. (K4)

52. Kubilus, N. J. (1987). Putting computer aided software engineering to work. In Proceedings of the Second International Conference on Computers and Applications (pp. 528-531). Washington, D. C.: IEEE Computer Science Press. (K5)

53. Kuo, J. H., Tu, H-C. (1987). Prototyping software information base for software engineering environments. In Proceedings of COMPSAC '87 (pp. 38-44). Washington, D. C.: IEEE Computer Science Press. (K6)

54. Lamsweerde, A. V., Delcourt, B., Delor, E., Schayes, & Champagne, R. (1988). Generic lifecycle support in the ALMA environment. IEEE Transactions on Software Engineering, 14, 720-740. (L1)

55. Lempp, P.R. & Rainer, A. (1987). Systems development environments. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 50-55. (L2)

56. Luqi, & Ketabchi, M. (1988). A computer-aided prototyping system, IEEE Software, 5(2), 66-72. (L3)

57. Marca, D., McKenna, N., White, S. (1987). Computer-aided support for coordination technology. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 293-299. (M1)

58. Martin, C.F. (1987). Second generation CASE tools: a challenge to vendors. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 56-62. (M2)

59. Mital, R.M., Kim, M.M., Berg, R.A. (1987). A case study of workstation usage during the early phases of the software development lifecycle. SIGPLAN Notices, 22(1), 71-76. (M3)

60. Moore, D. J., Drew, P. A., Ganti, M. S., Nassif, R. J., Podar, S. & Taenzer, D. H. (1988). Vishnu: An object-oriented database management system supporting software engineering. In G. J. Knafl (Ed.), Proceedings of The Twelfth Annual International Software and Applications Conference (COMPSAC), pp. 186-192. (M4)

61. Nizarri, M. (1987). Ideas for future tools. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 66-67. (N1)

62. Norman, R.J. & Nunamaker, J.F. (1989). CASE productivity perceptions of software engineering professionals. Communications of the ACM, 32, 1102-1108.(N2)

63. Orlikowski, W.J. (1988). Information Technology in Post-Industrial Organizations: An Exploration of the Computer-Mediation of Production Work. Unpublished doctoral dissertation. New York: New York University. (O1)

64. Perry, D.E., Kaiser, G.E. (1988). Models of software development environments. In Proceedings of the 10th International Conference on Software Engineering. (pp. 60-67). Washington: Computer Society Press. (P1)

65. Radhakrishnan, T. &. Jaworski, W. M. (1987). Audiographic teleconferencing and CASE. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 307-317. (R1)

66. Raghavan, S.A. & Chand, D. (1987). CASE: Towards man-machine partnership in software engineering. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 77-79. (R2)

67. Sorenson, P.G., Tremblay, J.P., McAllister, A.J. (1988). The metaview system for many specification environments, IEEE Software, 5(2), 30-38. (S1)

68. Symonds, A.J. (1988). Creating a software engineering knowledge base, IEEE Software, 5(2), 50-57. (S2)

69. Troy, D. A. (1987). An evaluation of CASE tools. In Proceedings of COMPSAC 87 (pp. 124-130). Washington, D. C.: IEEE Computer Science Press. (T1)

70. Umar, A. & Teichroew, D. (1987). Computer aided software engineering and distributed applications. In Proceedings of TENCON 87 (pp. 1197-1201). Washington, D. C.: IEEE Computer Science Press. (U1)

71. Wasserman, A. I. & Pircher, P. A.. (1987). A graphical, extensible integrated environment for software development. ACM Sigplan Notices, 22, (1), 131-140. (W1)

72. Wynekoop, J.L & Senn, J.A. (1989). Software Development Professionals' Perceptions of The Usefulness of CASE Tools (INTEC Working Paper). Atlanta: Georgia State University, Information Technology Management Center. (W2)

73. Yadav, S. B. (1987). Desirable features of a CASE tool. In E. Chikofsky (Ed.), Advance Papers for the First International Workshop on Computer-Aided Software Engineering, pp. 112-113. (Y01)