Presented at Workshop on Psychological
Research on User On-Line Interaction,
Boston, MA, October 26, 1975

LBL-4297
c.2

OBSERVATIONAL RESEARCH IN USER-COMPUTER INTERACTION

Daniel F. Kane, Jr.

October 1975

Prepared for the U. S. Energy Research and
Development Administration under Contract W-7405-ENG-48

TWO-WEEK LOAN COPY

This is a Library Circulating Copy which may be borrowed for two weeks. For a personal retention copy, call Tech. Info. Division, Ext. 5545

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY

LBL-4297
c.2

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-4297

OBSERVATIONAL RESEARCH IN USER-COMPUTER INTERACTION

Daniel F. Kane, Jr.

Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

TABLE OF CONTENTS

1. OBSERVATIONAL RESEARCH IN USER-COMPUTER INTERACTION 1
1.1 INTRODUCTION 1
1.2 THE DIFFICULTIES OF THEORY AND EXPERIMENT 1
TYPES OF RESEARCH 2
INFORMATION SYSTEMS 3
USER-COMPUTER INTERACTION 5
1.3 THE HEURISTIC METHODOLOGY 6
TEAMWORK 7
EVOLUTIONARY METHOD 9
DEVELOPMENT AS RESEARCH 12
1.4 THE CONTEXT OF LBL-MIS 13
RESEARCH ENVIRONMENT 13
NATURE OF THE USERS 14
MANAGEMENT INFORMATION SYSTEMS 15
1.5 SOME MEASURES OF SUCCESS 16
RECOVERY OF COSTS 17
EXPANSION OF USER POPULATION 18
CONCENTRATION ON PRODUCTIVE TASK 20
1.6 SUMMARY 22
1.7 ACKNOWLEDGMENTS 23

OBSERVATIONAL RESEARCH IN USER-COMPUTER INTERACTION

1.1 INTRODUCTION

There are three phases to science--observation, experimentation, and theorization. The psychological aspects of user-computer interaction are extremely complex, making theory and even experiment difficult if not suspect. Research on user-computer interaction is not carried out in a vacuum, however; it is applied research, directed to improving the mind-computer symbiosis in practical applications. So, while user-computer interaction is not a naturally occurring phenomenon, observation may take the form of heuristic development of a system with close cooperation from users, evolving the interactive interface along the patterns that seem most naturally useful. The environment at the Lawrence Berkeley Laboratory has been especially conducive to this type of research. An interactive Budget Management System, developed in this manner within the Physics Division, has been successfully adopted and used by other divisions.

This paper is primarily concerned with methodology rather than with justification of this field of research or with results from the author's work. The next chapter describes the nature of user-computer interaction in information systems and suggests what types of research are viable. The third chapter presents the observational methodology of heuristic development, discussing the value both of teamwork and of evolutionary development. The fourth chapter describes the management context at the Lawrence Berkeley Laboratory. Finally, the fifth chapter discusses some measures of success, as exemplified by the expanding use of the Budget Management System in the Lawrence Berkeley Laboratory Management Information System (LBL-MIS).

1.2 THE DIFFICULTIES OF THEORY AND EXPERIMENT

A basic characteristic of scientific research is open-minded skepticism. Simple observation is more conducive to this aspect than either experiment, the controlled testing of hypotheses, or theory, the attempt at generalized explanation. It is all too human to want to be "right", and even scientists are not fully immune. The problem is not serious when definitive results can be established. Unfortunately, information systems, even those that involve interaction between the user and a computer, are far too complex to allow the luxury of definitive results. Martin (19) suggests that the areas of interactive searching and interactive planning are not yet advanced to a point where a common core of

essential features is obvious. Bennett (1) points out that controlled experiments must be carefully interpreted. This does not deny the value of well designed experiments, even when they cannot be conclusive. But theories, hypotheses even, attempt to isolate the fundamental characteristics of a phenomenon, while in cognitive phenomena, such as information systems and interactive communication, it may be that the complex of interrelationships is the essential characteristic. The danger of oversimplification cannot be ignored. For example, the experiment reported by Walther and O'Neil (28) is interesting but of limited value.

TYPES OF RESEARCH

My academic experience included both the hard and the soft sciences. I did all of my degree work in physics, and I wrote my thesis in the area of strong interaction high energy physics. This field of research has certainly had the lion's share of definitive results. Physics is particularly susceptible to precise concepts, mathematical formulations, and experimental analysis. My research dealt with particle tracks from high energy bombardment, their angles and curvatures, the well determined interpretation as energy and momentum, high statistics, and least squares fitting to mathematical models.

On the other hand I have long had a keen interest in the study of the mind. While I have had little formal training, I have read extensively in cognitive psychology and neurophysiology. I am particularly intrigued by the extreme complexity of the phenomenon, both in the functional structure of the brain and in the multileveled processing of the mind. When Skinner (23) would like to deny the concept of freedom, I believe he is in danger of oversimplification. Skinner is certainly an experimentalist, and his results are fascinating and undoubtedly correct within the context he establishes. But I appreciate more the work of Lorenz (17) or of Piaget (8, 22), who attempt to observe psychological phenomena in a natural setting. Piaget's work with Inhelder involves experimental design, but it is still primarily an observational methodology.

From my background in physics I bring an appreciation for one aspect of the Heisenberg Uncertainty Principle--that observation affects the phenomenon observed. The more precise the observation, the more the effect, and controlled experimentation has a much greater effect than unobtrusive observation. This latter type does not lend itself to repeatability except in a very general sense,

but the same may be true of the phenomena being studied.

Different fields of research are more or less susceptible to different types of research. For the scientist interested in psychological phenomena in their natural environment, I suggest that observation is the better methodology.

INFORMATION SYSTEMS

The cognitive school of thought about mental phenomena, which so dominates the field that it is called cognitive psychology, treats thinking, learning, memory, and so on as processing phenomena. One of Piaget's chief observations is that the mind develops through the interaction of innate structures with the environment. I think of information as the result of processing, so it seems not unreasonable to consider information systems among cognitive phenomena.

Many people, including information scientists, have a tendency to treat "data" and "information" as synonyms. And while words serve primarily for communication, they may also be used privately to sharpen one's own thinking. A clear distinction between these two particular words has proven very beneficial to me, and Feltham (4) and Kennevan (15) have developed similar distinctions. A datum is an isolated fact, so data are many isolated facts. Information is a complex of interrelated data, which is quite a different beast. The whole is greater than the sum of its parts, if only because it also includes all the interrelationships among the parts. The confusion between the words is understandable in human terms; the moment a datum is received, the mind correlates it rather thoroughly with many other data (memories and other perceptions). The mind deals instinctively in information. But now that we are more and more including computers in our information systems, the distinction is important. The computer more naturally deals in data, shoving bits around with complete indifference. The computer must be told precisely how to develop the interrelationships among data in order to provide information, which is what the user wants. Information is sometimes called "useful data"--and certainly that is a distinction. What I have called data, with their isolation stressed, is completely useless per se. Still I would insist that the difference is not merely in attribute, but rather in kind. A datum is a unit; information is a complex formed of more or less processing.

Among computer scientists there is a parallel tendency to refer to data management systems and information systems interchangeably. Again I find a clear distinction useful, even though the two functions may somewhat overlap within the same code. I consider data management to include creating, deleting, or altering single data elements, fetching and replacing specified records. The information function includes selecting, ordering, comparing, and calculating. These two functions are invariably coordinated; even in data entry systems several data elements would be associated in a designated record. But the distinction is useful if only because the users, at least professionals and managers, prefer to deal in information--coordinated presentations, updates in context, and so on. The user wants to see interrelationships. Let the computer manage the data.

It is probably already clear that I do not, in fact, think of an information system as something that resides in a computer system. In the first place, information systems have been around much longer than computers. And even in computer based information systems, the user should be considered the most important component.

A system is a complex organization of interrelated components (sound familiar?), and it is best to keep the picture as complete as possible. The distinctions made by Penniman et.al.(21) among information systems, communication systems, and computer systems were quite instructive; but the full system involves all of these subsystems. Having made the point, I too shall normally drop the qualifier. In effectively modularized systems, many subsystems can be considered independently as systems in their own right. I think the communications function (the electronic networking, not the syntactic/ semantic interaction described below) can well be isolated. I deal with my terminal, and as the programmer I even worry about the computer and backing storage; but I hardly ever consider the communication network. To a lesser extent data management can be isolated. But it is not at all reasonable to consider the computer's information processing separately from the user's (mental) information processing.

The components in LBL-MIS are roughly depicted in Fig. 1--hardware, software, and personnel. I wish particularly to emphasize two components, the users and the programmers. The system presumably functions to serve the user's practical

requirements. The system should be attuned, especially for professionals and managers, to reduce the strain on the user (see further discussion in Kane (11)). This approach most appeals to me because I am not in the final analysis interested in user-computer interaction. I am really interested in the mind itself. But the mind cannot be studied directly. Most modern psychologists study it by observing the behavior it controls. Linguists may study it, in particular, by the verbal behavior of communication. I choose to study the mind using the computer as my primary modeling tool. That is, by matching "impedences" between the information processing in the computer and the mental processing in the user, by making the computer mesh so smoothly with the user's own spontaneous and multileveled thinking that he can become completely absorbed in the problem, I hope to study the computer's procedures, their structure and dynamics, as somehow analogous to the user's mental processes. While the perfect system and the perfect model is unattainable, at least in any forseeable future, it is quite reasonable to start with the crudest of models.

To follow up on the notion of the personnel as components of the system, I suggest that the proper function of the programmer is that of "transducer". A transducer reformulates the output of one component into equivalent but acceptable input to another component--for example, a gauge which converts pressure into electrical pulses. It is now widely recognized that computer systems cannot be completely designed in advance, but must be revised in response to unanticipated user requirements. This is particularly true for matching the computer's information processing to the user's mental processing. The programmer must take the feedback from the users and reformulate it as changes in the computing components, normally the interactive software.

USER-COMPUTER INTERACTION

The discussion here is restricted to user-computer interaction in the context of information systems. Excluded, for example, are action systems--where worker and computer cooperate in controlling a lunar landing module, a rapid transit system, or a chemical plant; and data gathering systems--where the computer simply absorbs the worker's clocking in and out, job completions, and so on. So the phenomenon is essentially one of two way communication--dialog. As pointed out by Kane (11) and Martin (18), in most systems there is only one active participant--depending on the system, either the user or the computer is

normally passive and merely responsive. The more exciting systems involve active participation of both computer, reminding and prompting, and the user, giving spontaneous commands. In this form, user-computer interaction is much more akin to the interpersonal communication of natural language.

In a sense it is somewhat easier to talk to a computer than to another person, since it is always possible to determine, with more or less effort, precisely what the computer "means". On the other hand, the mind does not naturally deal with such precise meanings. Too much precision overloads the circuits. This has led Zadeh (30) to develop the concept of fuzzy sets, enabling the computer to handle the imprecision of psychological and social phenomena. I am more intrigued by the potential of dynamic processes, rather than structural attributes, for facilitating the attunement of the computer to the user. This involves letting the user view the information and manipulate it from as many aspects as possible, as arbitrarily and spontaneously as desired. Chomsky (3) distinguishes linguistic competence from actual performance; the former is the potential inherent in the grammar (broadly defined) while the latter depends on the individual's experience, mood, other perceptions, and so on. This is to a certain extent a distinction between structure and dynamics. And while the structural nature supports the dynamics and is certainly far easier to study, the natural occurrence of communication, either interpersonal or user-computer, is essentially dynamic. This has been discussed, for interpersonal communication, by Chapanis (2).

The concern among computer scientists has been too exclusively devoted to syntax, namely command languages and parsing, instead of dynamics. Research in user-computer interaction, especially with its orientation to practical results, cannot neglect dynamics. But the study of all possible time sequences of exchanges is not easily approached by experimental analysis. A better approach to such complex phenomena is intuitive synthesis, based on observations, using the Gestalt/pattern processing faculties of the researcher's mind (see, for example, Ornstein (20)). This may not be particularly subject to control or repetition. But Gell-mann (5) has stressed its value in the hard sciences, such as physics.

1.3 THE HEURISTIC METHODOLOGY

Given the complexity of user-computer interaction in information systems, observation is a better research approach than controlled experimentation.

But user-computer interaction is not a naturally occurring phenomenon. The interactions between man and man, or between man and animal, evolved over hundreds of millions of years and are certainly widespread. The interaction between users and computers (direct, on-line, interactive work by non-programmers) has been going on for only a few years within a very limited population. The phenomenon is far from steady state or significant statistical samples.

Clearly intensive, rather than extensive, observation is required. And the rapid evolution of the phenomenon naturally calls for a heuristic methodology, noting what works best and pushing the systems in such directions. These two aspects will be discussed as the teamwork method and the evolutionary method. And since these methods seem more useful for development rather than research, a final section will emphasize the research value of the method.

TEAMWORK

In the management information systems literature of the sixties, there was widespread recognition that management had to be directly involved in the design and implementation of such systems. But was it ever given more than lip-service? I suspect that more often than not management helped set up the criteria, left the systems analysts and programmers alone to work out the details, and expected a complete system, working per specification, delivered by a certain date. The trouble with this approach is that management can, at the very best, get what is specified; but that is not necessarily what is really wanted or needed. Information systems are very complex, and experience is generally a better guide than analysis. It is now widely recognized that systems must be designed so that they are easily modified, later, to satisfy unanticipated user requirements.

Such adjustment to the unanticipated is best accomplished by close cooperation between the users and those implementing the software. This is especially true in developing systems involving user-computer interaction, as discussed by Hinckley et.al. (7) concerning the development of LBL-MIS. Such teamwork is best done intensively with a few programmers working intimately with a few critical users.

Critical users are those who do not just accept what they are given. They actively, and constructively, criticize. They ask, "Why can't it do this, or that?". The preferred critical user does not think, at least initially, about

the cost of implementation. (For any good suggestions, savings during use will often offset costs of implementation). Many procedures that seem straight forward in mental processing are very difficult to implement in computer processing. If we ever hope to attune the system to mental processing modes, we cannot trust the impulses of the programmer, who has to do the hard work. The user, who (at least in his role as user) is only interested in getting his own job done most effectively, is a more unbiased director of the heuristic method. The user is like the harsh environment in the evolutionary process of natural selection. If the user is not the funder, of course, the latter must be brought in to balance the environment.

The programming members of the team should work in intimate proximity to the few initial users, such as in adjacent offices. The team should coordinate closely on a day to day basis. This gives the programmer the opportunity to observe, not only how the user is interacting with the computer, but also the procedures that the user is still performing by other means. Off-hand, informal conversation can often lead to fuller understanding of requirements, not just in content but also in style. This certainly demands a significant investment from the users of more than just money. But this investment is really necessary to achieve the best system.

A greater than normal investment is required from the programmers as well. This type of research and development requires "clever programmers" who are both dedicated to serving the users' real needs and interested in the underlying psychological principles. As discussed earlier, the programmer closes the cybernetic loop, transducing the user's experience into changes in software. For anyone not aware of current trends in computer science, "clever programmer" is a pejorative reference. Such a programmer likes to do "neat tricks", which normally no one else can understand or maintain without great difficulty. The programming manager prefers programmers who use straightforward, standard procedures. And, given that manpower costs more than any potential savings in computer time, this is certainly most cost effective for most applications. But while I was designing and implementing LBL-MIS, I could not have kept up the pace if I had not used every short cut I could think up. This requires a commitment, of course, to convert to standard procedures in more leisurely times. Another consideration, however, is that in information systems involving

direct user-computer interaction, the major cost consideration is probably neither computing resources nor implementation effort, but rather the users' effort. The research aspect of such development may be viewed as discovering ways to save the users' effort. If the techniques and procedures that best accomplish this are not immediately understood by other programmers, then the developer should be prepared to educate his colleagues.

EVOLUTIONARY METHOD

Since everyone generally agrees that systems must accommodate subsequent change, I wish to stress an early and intentional shift into the iterative process. Martin (19) indicates that good instructions (not just exhaustive instructions, but good instructions) cannot compensate for inadequate design. There must be subsequent adjustment. Martin (18) describes the reiteration of implementation stages in design and development. Thompson (26) goes further to urge getting the system into early operation to find the problems and iterate. I would like to point out that the extremely complex interaction of interpersonal communication evolved over hundreds of millions of years, and I suggest that in trying to effect user to computer communication we use the same evolutionary, experiential approach. While our rational, analytic capacities will hopefully shorten the time required to get practical results, we cannot expect to anticipate a complex reality. Instead of trying to do too much advance design, let us observe and adjust along the way. This is an example of Teilhard's directed evolution (25).

What evolutionary stages are involved when developing user-computer interaction in information systems? My approach to the generation of the supportive software is loosely based on my understanding of the evolution of the human mind. This phenomenon did not spring forth full blown to reality. It started with the single cell (to pick a point with behavioral capabilities), passed through more and more complex living and thinking organisms, to reach the current levels of self-reflective thought (25). Along the way, many evolutionary branches were attempted, randomly directed by mutation, and harshly selected by survival of the fittest (17). Hopefully the evolution of user-computer interaction can proceed more effectively, directed by rational analysis; but I think we would be most unwise to vaunt ourselves too highly above Lorenz' great constructors. I am content to use my analytic powers to observe and adjust, rather than to outguess and anticipate.

So my prescription is to get started with a rudimentary system, a skeletal system that supplies only the most basic features. Forget about all the great, sophisticated procedures for which computers are so well suited; they are often beyond the user's immediate comprehension and the results must be accepted on faith. I am often discouraged by a common tendency of people to believe what is on a computer printout just because the computer did it. I have worked with so many programs by now that I know I cannot really trust computer output until I have been able to crosscheck it. Presumably, after an initial shakedown, the results will be correct, but enough data should usually be presented for a crosscheck, if only so the user can periodically and by active processing assure himself that he is correctly apprehending the semantic and informational content based on interrelationships. The highly sophisticated processing is certainly a most valid use of computers; often such processing cannot be done otherwise. But this is better left to batch mode, where the user has more leisure. To be really effective, the interactive mode depends heavily on a facility for reassurance. This can be done by extension. Whenever I pick up a hand calculator, either novel or after long disuse, I always check 4 + 4 and a few other simple calculations. Here it is not because I distrust the hard-wired logic, but because I want to be sure I have the order right for pushing the buttons. Then I am happy to accept complex results. Similarly, any scientist sitting down to a bibliographic retrieval system will invariably ask for selection on his own name, not (only) as ego reinforcement, but because he knows the answer. Or he will try boolean combinations to get a field that is completely familiar to him (a good system just might provide him with an unknown reference, but it had better not leave out any known ones). A system for user-computer interaction must allow for and even facilitate healthy distrust.

Another reason for starting with a rudimentary system, and then implementing more complex or specialized procedures as they are needed, is to develop emphasis. Many would argue that if the designer does not get the whole pattern organized in advance, it will be difficult to add the options later. But I am very wary of preconceived, generalized systems that do everything equally well (13). That is not the way the mind works. Some things are easier to remember because they are needed and used more often. Habits, which can be practically automatic, greatly reduce mental effort, as James (9) has pointed out. Similarly, in user-computer interaction the procedures that are most often needed should be most

readily usable. And, as a first approximation, the procedures that come up for addition soonest will be the ones most often used. The programmer cannot at the start ignore the later difficulties of adding options, but this is best resolved by generally modular organization very susceptible to arbitrary modification.

Many evolutionary branches fail to survive in the case of living organisms. We should be no less harsh on software procedures. Any facility that is little used, even if used on rare occasions, should be excised. I am here assuming a rich redundancy in the system, so that the particular function could still be accomplished by other, although in specific cases less efficient, methods. The mind is quite competent at weaving a pattern from a set of basic procedures to achieve a more complex objective. It is, of course, important that all of the procedures be spontaneously available. But keeping track of all the internal details of many specific procedures can be quite a task. In the text editor that I use, I know there are some commands that would accomplish in one step what I do in several if I knew how to use them. But I am not about to overload my memory capacity; I get by. The software designer must recognize that different users will have different needs. But I think there are valid cases in which one can eliminate--or even not implement to begin with. I strongly advocate following user-originated directives; but sometimes my expertise in cognitive phenomena must override. Sometimes I am convinced that a request would not work as expected and I will not implement it. The user certainly knows his job better than I, in his direct ability to perform it. But in some ways I may understand it better, in the analysis of the cognitive processes. So natural selection may be encouraged by the programmer.

Finally, the development team, through the programmer, should see to the survival of the fittest techniques. In implementing different procedures in the user-computer interaction, different styles, phrasings, and dynamics of dialog should naturally come to mind. It is unwise to decide in advance on one consistent pattern for dialog. That is just not the case with interpersonal communication, which is primarily characterized by its flexibility. The mind can handle a great deal of variation, much more than the computer (or its programmer). Still, with all its variety, human speech follows certain basic types of patterns (3). Similarly, certain basic types of techniques should emerge as most comfortable for user-computer interaction. These cannot be anticipated in advance, so it is

best to implement whatever seems most appropriate in a given context. After the fact some techniques will stand out as the more successful, and then the programmer should not hesitate to rewrite the software--transducing the user's experience--using the better techniques where possible. Ego-involvement in previously written code could prove fatal to an embryonic or immature system. This emphasizes the value of starting with a small team where only a few users go through the changes. But even in a more mature system with many users, the adjustment may not be difficult in a truly interactive system. In a command system, where the computer is passive-responsive, this would be very difficult. But in sophisticated dialog, command language plays only the spontaneous, initiating role. The computer can pitch in with questions and comments; in a word, prompting. In this milieu of pleasant repartee, the user will easily recognize the unchanged semantic content even though the computer's syntax has changed (and presumably improved).

I think computer scientists are far too concerned about the structure of command languages and parsing rather than about the dynamics of real communication. Children do not learn to speak grammatical sentences; they start with many short phrases. Why not let computers do the same? Of course, compilers have to be mature, since they must interpret "dead" communication coming from a file. This is what Chomsky means when he says formal computer language can be adequately represented by its "deep structure" (3). That is not true and need not be expected of interactive dialog. Keep it simple and keep it dynamic. I maintain that any batch program put on-line, by using the terminal as input and output files, does not deserve the appellation "interactive". It is on-line batch, where the batches contain single entries.

DEVELOPMENT AS RESEARCH

In concluding this chapter I want to reemphasize that although the methods described are in fact methods for developing information systems which support user-computer interaction, still these methods are the best way to do the research. This is a field of applied research primarily--how can the mutually reinforcing, symbiotic relationship between the user and the computer best get a job done? But it is certainly possible to do both research and serve the user's needs. In many instances of applied research, doing relevant research best serves the user's needs in the long run. Most pure research is eventually applied. But in such a complex phenomenon as user-computer interaction, I submit that the

best way to do the research is to serve the critical user in the ways I have described.

Are significant results possible, as opposed to good, specific applications? Bennett (1) suggests that theory in this area is rudimentary, and that the resulting technology takes the form of procedural rules. In this sense, general results are certainly possible. And while this paper is devoted to methodology, some tentative discussion of results of this author's research is already available (11).

1.4 THE CONTEXT OF LBL-MIS

Before discussing the success of the observational methodology in the development of the Budget Management System, it would be well to describe the rather specialized context of LBL-MIS. This will be presented briefly in terms of the research environment, the nature of the users, and the functions performed.

RESEARCH ENVIRONMENT

The observational methodology described in this paper may seem hardly a practical way to develop a major system. And although the design team was convinced from the start that this was the best approach to take, we recognized both that it was highly irregular and that we had no reasonable assurance of success. The environment at the Lawrence Berkeley Laboratory was especially conducive to our work, however.

The management members of the team were prepared to take a calculated gamble on the development of the system. And it should be noted that, although the senior manager had the broad responsibility for management reports, the initial development was not coordinated on a lab-wide basis but was focused within the Physics Division. This was the natural place, since the Physics Division by historical and practical coincidence had responsibility for the computer center and for Math and Computing research and development. This is shown in Fig. 2. Also, the division had by far the most complex management problem, in desperate need of better computing support. The developing system was treated as primarily experimental. As described by Hinckley et.al.(7), the new procedures were normally run in parallel with the existing manual and batch procedures. Although the interactive procedures became the standard methods as soon as they were

shown to agree, as they were very much faster, the managers were always prepared to fall back on old methods. This was because so much of the "documentation" and understanding of the software was locked up in the head of the one design programmer.

I certainly risked the possibility that my accidental death would leave a working system unsecured and unmaintainable through changes in computing or management environment. I was not about to waste my time on documentation of a rapidly changing system. This was the agreed approach during the early period of experimental development. Now that the Budget Management System has many users, and the system is being considered as the start of a coordinated lab-wide system (12), the documentation and the education of fellow programmers assume high priority. Probably the general research atmosphere at the Lawrence Berkeley Laboratory was critical to the early research approach. I strongly felt the need that there should be no hesitation if massive rewrites of the software were necessary. I could not expect management to accept this on faith, and I willingly invested my own time to do so because of my research interest. I have rough records for the first two years showing a sixty hour work week minimum. Management cannot, of course, expect that level of effort; it derives only from a research commitment. But such commitment flourishes, and is amply recognized, in a research environment such as the Lawrence Berkeley Laboratory.

Perhaps the circumstances of the development of LBL-MIS were unusual, but hopefully the resulting success, probably not of just this one system but cumulatively, will encourage other managers and other computer scientists to utilize this observational methodology.

NATURE OF THE USERS

There is some closet humor among computer scientists that, if a system can be used by children, then probably it is ready for the managers. Kidding aside, this has a serious point. As everyone knows (see, for example, Martin (18)) managers cannot waste their valuable time learning a difficult system. They would instead work through intermediaries. At conferences I am met with a certain incredulity that managers are using LBL-MIS directly. In fact, the managers, both those who initiated and participated in the development as well as those who have become subsequent users (and participate in the continuing improvements), have all been highly intelligent. It is important to note, however, that they

do not all come out of scientific or technical backgrounds. Also, outside their role as "critical users", these managers have been most tolerant of and interested in the research aspects of the project. Still, as best suited my methodology, getting their job done always took precedence. These managers all have broad responsibilities, each coordinating budgets of the order of five to ten million dollars.

The managers using LBL-MIS may be classified as middle management. I am essentially using the three levels suggested by Kennevan (15). The highest level, which I shall call executive, is primarily concerned with strategic planning, or policy making. At the laboratory this is the scientific leadership, concerned broadly with the direction of the current and future course of research. Executives deal in matters too intangible for computerized information systems (unless possibly text management of research descriptions). The third level involves day to day operations, directed in scientific areas by group leaders and in technical areas by supervisors. These operations are widely computerized but are not the focus of LBL-MIS. The managers, who are in the middle level (no one should misconstrue that group leaders report to managers), are primarily concerned with resource allocation and scheduling in support of the scientific purpose.

MANAGEMENT INFORMATION SYSTEMS

Rather than furnishing general purpose information processing and report generation, LBL-MIS concentrates on very specific aspects of management problems, primarily aimed at lifting from the manager the load of data management and many, often repeated calculations. All of the calculations could be carried out by the manager on a hand calculator, given all of the data, but at a much greater expenditure of the manager's time and with greater chance of human error. The manager is very familiar with the operations and can also do manual checks of the system to maintain confidence in his approach. Another very important reason for keeping the range of procedures small is to allow building a high degree of responsive intelligence into the software. The user does not have to remember much detail. First the list of spontaneously applicable commands is quickly produced when desired. But more important, in specific contexts the computer is primed to ask in turn for needed details or to make appropriate comments. The dynamic mixture of command and prompting processing, representing

the active participation by the user and the computer respectively, is the fundamental principle of these interactive systems (11).

In the Budget Management System in particular, the computer facilitates allocation and scheduling of resources in terms of expense (dollars), effort (manmonths), or people (full time equivalents). Conversions among different representations are supported by general or specific parameters (as appropriate and under control of individual managers)--rates (dollars/man month), average salaries and fringe benefit factors, and leave pattern factors. Definite accounting algorithms, such as application of support burden or overhead, are always performed automatically--and adjusted automatically whenever the manager changes the rates involved. This requires a very specific and complex data structure, which could in no way be supported by a general data base management system. In LBL-MIS data management is simply the fetching and replacing of data elements and records. All of the rest is information processing highly specific to the application. This means that when the manager makes a basic change, he need not go through a lot of other follow up procedures. The system knows enough to cut through to the full implications automatically. The system probably could not have been successful without this time saving specificity. The Budget Management System also maintains historical and current accounting data, both for comparison to current plans and for abstracting patterns for (modifiable) use as current plans. The users' manual, of course, supplies much more detail (10).

1.5 SOME MEASURES OF SUCCESS

Any research methodology cannot succeed, cannot even proceed, without some measures of success. Whether these measures are the correct ones or not, and they are certainly subject to change with time, the researcher without them either glides on undeviatingly or flails about at random. Measures of success gear the cybernetic process of directed activity. The intent of this chapter, then, is not to show that LBL-MIS has been successful. Only those who have benefited from its use can have any direct appreciation of that. Rather I wish to present what I (currently) consider to be my measures of success with references to the Budget Management System only to exemplify these measures.

RECOVERY OF COSTS

Without entering into the question of social desirability, it is safe to say that the competition of pure capitalism best supports natural selection in the economic sphere. It is very exciting to me to see on-line information systems actually surviving and making a profit in head-on competition. The field of bibliographic information searching is not purely competitive because of the high investment barrier; and the systems at Lockheed Research Laboratory (16) and at Systems Development Corporation (24) tend to dominate the area. But the network of users, international in scope, normally has access to both systems from the same terminals. Where users spend their dollars is certainly a measure of success for either system. Actually both have succeeded, since the other, unsuccessful systems have dropped out of the market. I have noticed the tendency of both ORBIT and DIALOG to react to this measure of success. Both systems have many advantages, and users press the system developers to make available the facilities from "the other system". Wanger (29) has made the point that this is the best approach to standardization. It is an excellent example of the evolutionary emergence of the basic types of techniques most comfortable for user-computer interaction.

In my development of information systems I am very wary of doing subsidized work. If the direct user is also the funder, in the sense that he could readily apply the resources to other important projects, then he can more instinctively integrate the conflicting values of ease of use and cost effectiveness. I have nothing against pure research, but pure research certainly lacks the driving force of cost recovery as a measure of success. Of course, managers might be motivated by the prestige value of a terminal in the office, although that has never appeared to be a motivation having much persistence. Managers, certainly, have been more concerned with cost recovery on a broader scale, with a primary concern for improved utilization of the organization's resources and only a secondary concern for reducing administrative costs (7). The latter, if done injudiciously, can lead to slip-shod, cut-rate management.

The cost recovery in the case of bibliographic information systems is quite obvious--an on-line session, even at rates up to one hundred dollars per hour connect time, is nominal if it can save a scientist or engineer days in

the library.

EXPANSION OF USER POPULATION

There is a classic proverb--build a better mousetrap and the world will beat a path to your door. And certainly if users specifically choose to work within a given information system, that is an important measure of success.

The proverb says nothing about advertising. And while advertising is a valid means of expanding the population of customers, the unsolicited expansion of the user population is even more impressive. It means that the word is getting around on its own, and that the value of the system is apparent to potential users. This has in fact been the case with the Budget Management System in LBL-MIS. I have been most reluctant to expand the user population--even after I had made it possible by bringing the Math/Computing group on with independent files. (The second user is the major hurdle to expansion in any ad hoc system.) Fig. 3 shows the addition of new users up to the time of this writing. The presentation of the system to the director of the laboratory and the publishing of a major report on the efforts and results, while not intended specifically to attract other users, certainly spread the word. Eventually a management review committee on administrative data processing was constituted to coordinate lab-wide development, with this system as one important point of departure. In this expanded development I shall still insist on the teamwork method, with small teams involving the initial users and the few programmers directly involved; but I certainly recognize a not entirely new but now much more critical challenge of organizational psychology--the overall coordination of many interconnecting projects in a broader but in reality very integrated context.

My reluctance, never resistance, to expanding the user population was based on a desire to maintain the possibility for intensive observation. The expansion of the user population has necessitated the development of new methods. Foremost was the completion of a comprehensive users' manual. The manual available now is complete, detailed, and cross referenced. This, of course, makes it difficult to use and should eventually be supplemented by an instruction manual--organized not by the distinct aspects or commands types of the system, with full detail provided, but organized rather by what few commands should be learned first, which next, and where appropriate details should be introduced. Such an

instruction manual is not yet available but has been simulated by the use of a highlighted hardcopy of a session of direct interest to each individual manager. As each manager began to work on his personal file, I discovered what particular approaches he could use immediately--entering expense allocations directly and converting to effort, entering known full time equivalents and deriving expense, basing a first guess on the prior year's pattern, or whatever. Then, using a printing terminal (high speed CRTs are normally used), I did a few examples directly from the manager's own file. When the input was highlighted with yellow marker and appropriate comments were added, both by myself and by the manager, the result was a very personalized entry into the system. Since the massive data on the file had already been aggregated from the accounting data bases, in terms of designated groupings of accounts, the manager could easily get productive results within weeks, based on the comparison of plans to actual utilization. I certainly consider this a measure of success, and busy managers would not be inclined to use the system otherwise. Because the communication is really interactive dialog, which both makes it easy and comfortable to use and stimulates immediate change of plans due to feedback, the managers strongly prefer direct use over working through intermediaries. In fact the more experienced managers tend to do all detailed planning right at the terminal.

Not quite so obvious as a measure of success is that individual managers can, in fact, use quite different styles in the Budget Management System. One might argue a preference that all managers use all aspects of the system. But different managers work in different ways. There is no point in overloading memory capacity. Emphasis is good. Martin (18) agrees that, for management information systems, allowing different styles is important as long as the system also improves the precision of communication among managers. The computer software, which contains the algorithms for converting one form of expression into another (man months to dollars, say, given rates), can readily facilitate better communication. But each manager can work with whatever approach is most personally comfortable. Such depth in a system is, to me, a valid measure of success. In fact, even when very busy, I have always made time for informal discussions, concerning management style and concerning laboratory policy, both for the user to enhance use and for me to enhance implementation. As the number of users has grown, private discussions have been supplemented by users group meetings and by surveys, which have stimulated group transfer and learning. The

managers, although having their own styles, have picked up techniques from other managers just because the techniques were in the system. And the modular and ongoing design of the system minimized a prejudice one might naturally expect. Namely, a number of procedures desired by new users have been readily implemented and then quickly adopted, with all due appreciation, by users who were on the original design team.

Finally, the expansion in the number of users has clearly prevented close, direct observation in situ --at least with any regularity. This could be rectified by the capture and study of the electronically transmitted dialog, as suggested by Katter (14) and Treu (27). The system is designed so that all telecommunications pass through a single focus, where capture is trivial. I have not used this method, however, even though it may be unobtrusive observation (barring any overtones of surreptitious snooping). In fact, it places an extra load on the computer resources, and also I feel uncomfortable about even looking at any such statistics isolated out of such a rich psychological context. A colleague has been using the dialog capture version of my focal module (6). But his users are spread out all over the country, and this is his only means of "getting to their offices" whenever problems arise.

CONCENTRATION ON PRODUCTIVE TASK

In the final analysis, my ultimate measure of success is the possibility for the user to completely lose conscious awareness of the computer, of the files, of even the terminal. When the user can concentrate fully on the productive task, absorbed completely in his mental activity, with results, implications, questions, and even critical comments appearing upon the screen as if cross talk from another channel in his own mind, when the user is not distracted by alien technique, then the symbiosis will have reached ultimate success in its practical aspect and penultimate success in the research aspect (leaving analysis and understanding of the structure and dynamics of the software model). This characterization of the measure of success is obviously romantic and absurdly idealistic. But it is the driving force behind my research commitment.

At a more practical level, I judge this measure of success by the extent to which the users stop bothering the programmers. Remember, these are "critical users" with an intentional responsibility to give constructive feedback. Also

"bothering the programmers" is restricted to features already in effect but not easily usable.

There are other reasons (which are more a matter of stimulation than bother) why the users come to the programmers. These concern new features that they want added. The time sequence in Fig. 3 shows in a gross sense the addition of new features to the Budget Management System. It has been continually true that the managers want more and more hands-on control of all features of the system. The budgeting procedures, based on the fiscal year, were the original interactive implementation. Comparisons to accounting data were initially batched off-line. But it was not convenient for the manager, whenever he changed some plans, to get the programmer to run off reports. So the facility was added to instigate report generation from the terminal. Eventually all accounting data were aggregated periodically by the programmer onto the random access file and made available on the terminal. This meant that reports were now less often needed, primarily for historical record or for distribution to group leaders, and also that scheduling of resource utilization could be anticipated in the data base. When the number of users grew, it became desirable to let the managers call as well for the aggregation of historical data off of the accounting data bases onto their own personal files. The users are now strongly pushing for the ability to restructure their project account definitions, on-line and at any time.

This historical perspective clearly shows the evolutionary methodology of implementing the most immediately useful procedures first and then following the emerging pressures for additional features. I certainly do not look forward to the day when the users run out of new ideas. But the measure of success described in this section involves satisfaction with facilities already implemented. When such features are sufficiently tractable, the users will make do, and the programmer ceases to hear about them.

An anecdote may serve here to illustrate the success of the Budget Management System in this regard. When the first really new users from the Chemistry Division started on the system, I spent a couple hours on two succeeding afternoons showing off the system and making a highlighted printout. This was done on their own file, which had previously been aggregated off the accounting data bases. Since I did not want to encourage use, I then left them to proceed as

they chose. But after two weeks I got a little concerned and called them. They claimed they had been using the system all along, quite successfully; and a subsequent check of computer day files confirmed at least that they had been using it. In all fairness, this case involved a manager who had considerable technical background and who was aided throughout by a colleague with considerable programming experience. But still, that anyone could use so complex a system with so little help (the users' manual was also poor and incomplete at that time) impressed even me as an indication of success. I thought the system was good; I did not realize it was that good.

1.6 SUMMARY

An observational methodology is the best approach to research and development for user-computer interaction in information systems. This phenomenon is akin to interpersonal communication, very rich in complex interrelationships, and is not susceptible to controlled experimentation.

There are two primary aspects of the observational methodology as described. The intensive aspect involves the close teamwork among programmers and users in the development of the information system. The heuristic aspect involves the intentional evolution of the system from rudimentary to complex, primarily directed by the users' perceived needs, and harshly selected on the basis of observed ease of use.

Any research methodology must be directed by some measures of success. In applied research, cost recovery is certainly an important measure. For information systems, the expansion of the user population is another measure, especially if new users come without solicitation, are able to get productive results quickly, and can use the system in different ways to suit their own needs. The most important measure of success for systems involving user-computer interaction, however, is that the users can concentrate on doing their job, undistracted by techniques alien to their own mental processing.

1.7 ACKNOWLEDGMENTS

Quite obviously this work could not have been performed without the dedicated support of the users, not only in funding but also in constructive criticism and informal discussions. Bob Hinckley and Wes Weber have had the longest and most active involvement, but I have greatly benefited from the many others using my various systems--Olivia Austria, Igor Blake, Bernard Harvey, Julie Kearney, Jane Kennedy, Ev Magnuson, Eddie Reed, Howard Smith, Mabel Smith, Harold Wilson, and Steve Yabroff.

Among colleagues, I have especially benefited from discussions with Don Austin, Brad Heckman, Marilyn Mantei, Jeannette Mahoney, and Jerry Zimmerman. I am indebted to Jim Carlisle, from the Information Science Institute, for inviting me to present my ideas at The Workshop on Psychological Research on User On-line Interaction. This workshop was held in conjunction with the annual conference of the American Society for Information Science in Boston, Massachusetts. Without Jim Carlisle's encouragement this paper might still be on a back burner in my mind.

This work was performed under the auspices of the U.S. Energy Research and Development Administration.

REFERENCES

1. John L. Bennett, "The User Interface in Interactive Systems," Annual Review of Information Science and Technology, Vol. 7, American Society for Information Science (Washington, D. C., 1972).

2. Alphonse Chapanis, "Interactive Human Communication," Scientific American Vol. 232, Number 3, March 1975.

3. Noam Chomsky, "Language and Mind," (Harcourt Brace Jovanovich, Inc. New York, 1972 [enlarged edition]).

4. G. Feltham, "Information Evaluation," (American Accounting Association, Sarasota, Fla. 1972).

5. M. Gell-Mann, "The Value of Schemes in Scientific Theory", Proceedings of Convegno Mendeleeviano, Academico Scientifico (Turin, Italy 1970).

6. Bradford Heckman, "RGNLMIS Users' Guide", Regional Management Information System Project (RMIS) Report on First Year's Activities, September 1975 (U.S. Department of Labor Manpower Administration, Region IX San Francisco, Ca., Lawrence Berkeley Laboratory U.C. Berkeley)

7. R. L. Hinckley, D. F. Kane, J. Kearney-Wright, P. W. Weber, J. L. Zimmerman, LBL-MIS, A Computer Aided System for Management of Research, Lawrence Berkeley Laboratory report LBL-3089, September 1974.

8. Bärbel Inhelder and Jean Piaget, "The Growth of Logical Thinking from Childhood to Adolescence", translated by Anne Parsons and Stanley Milgram, (Basic Books, Inc. translation published 1958 New York).

9. William James, The Principles of Psychology, Vol. I, Henry Holt and Company, New York, 1907.

10. D. Kane, The Budget Management Facility, Lawrence Berkeley Laboratory Report UCID-3670, Revision 1, Latest Revision December 1974.

11. D. Kane, Subsets on the PSS Library NTRACT, Lawrence Berkeley Laboratory

report UCID-3668, Revision 1. Latest Revision January 1975.

12. D. Kane, A Concept of Self Service Information Retrieval, Lawrence Berkeley Laboratory, report UCID-3787. Latest revision, July 1975.

13. D. Kane, A Personal View on Documentation, Lawrence Berkeley Laboratory Report UCID-3784. Latest Revision October 1975.

14. Robert V. Katter, "Insights in Implementing the Redesign Cycle", in Interactive Bibliographic Systems, AEC Symposium Series 28 Madeline B. Henderson, editor, U.S. Atomic Energy Commission Office of Information Services, Washington, D.C., April 1973.

15. Walter J. Kennevan, "Management Information Systems", in Management of Information Handling Systems, Paul W. Howerton, editor, Hayden Book Company, Inc. Rochelle Park, N.J., 1974.

16. Lockheed Palo Alto Research Laboratory. DIALOG--Terminal Users Reference Manual, 2 vols. Lockheed Retrieval Service Information Systems Laboratory, Palo Alto, California, no date.

17. Konrad Lorenz, On Aggression, translated by Marjorie Kerr Wilson, Bantam Books, Toronto, translation copywrite 1966 (original 1963).

18. James Martin, Design of Man-Computer Dialogues, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1973.

19. Thomas H. Martin, "The User Interface in Interactive Systems", in Annual Review of Information Science and Technology Vol. 8, Carlos A. Cuadra, editor, American Society for Information Science, Washington, D.C. 1973.

20. Robert E. Ornstein, The Psychology of Consciousness, W. H. Freeman and Company, San Francisco, 1972.

21. W. David Penniman, Richard E. Krohn, Gabor J. Kovacs, "A Framework for the Study of Emerging Network Technology", in Journal of the American Society

for Information Science, Vol. 25, Num. 6, Nov.-Dec. 1974, Am. Soc. for Inf. Sci., Washington, D. C., 1974.

22. Jean Piaget, The Construction of Reality in the Child, translated by Margaret Cook, Basic Books, Inc. translation published 1954, New York.

23. B. F. Skinner, Beyond Freedom and Dignity, Bantam/Vintage, Toronto/New York, 1971.

24. System Development Corporation. SDC Search Service: ORBIT Users Manual. System Development Corporation, Santa Monica, California, April 1975, TM-5511/000/00.

25. Pierre Teilhard de Chardin, The Phenomenon of Man, translated by Bernard Wall, Harper & Row, New York, translation copywrite 1959 (orig. copywrite 1955).

26. Fred Thompson, "The Need for a Science of Information:, in Multi-access Computing Paul H. Rosenthal & Russell K. Mish, editors, Hayden Book Company, Inc. Rochelle Park, N.J., 1974.

27. Siegfried Treu, "Techniques and Tools for Improving the Interactive System Interface", in Interactive Bibliographic Systems AEC Symposium Series 28, Madeline B. Henderson, editor, U.S. Atomic Energy Commission Office of Information Services, Washington, D.C., April 1973.

28. George H. Walther and Harold F. O'Neil, Jr., "The User-Computer Interface in an Information Utility Delivery System: An Empirical Approach to User-Centered Design", in Information Utilities, Proceedings of the ASIS Annual Meeting, Volume 11, Pranas Zunde, editor, American Society for Information Science, Washington, D. C., 1974.

29. Judy Wanger, Systems Development Corporation, Santa Monica, Ca., Private Communication.

30. L. A. Zadeh, Fuzzy Languages and their Relation to Human and Machine Intelligence, in Man and Computer, Proceedings of the International Conference Bordeaux, 1970, pp. 130-165, S. Karger, New York, 1972.

Figure 1. An information system is a complex organization of many interrelated components--hardware, software, and personnel. The users and programmers are stressed here.

Figure 2. The original design team of LBL-MIS included members from the Physics Division Office and the Math and Computing research and development group.

Figure 3. The Budget Management System in LBL-MIS has evolved with emphasis on determining how to facilitate ease of use. New facilities were added in response to developing needs. Although the development was reported in various forums, the new users came without solicitation.

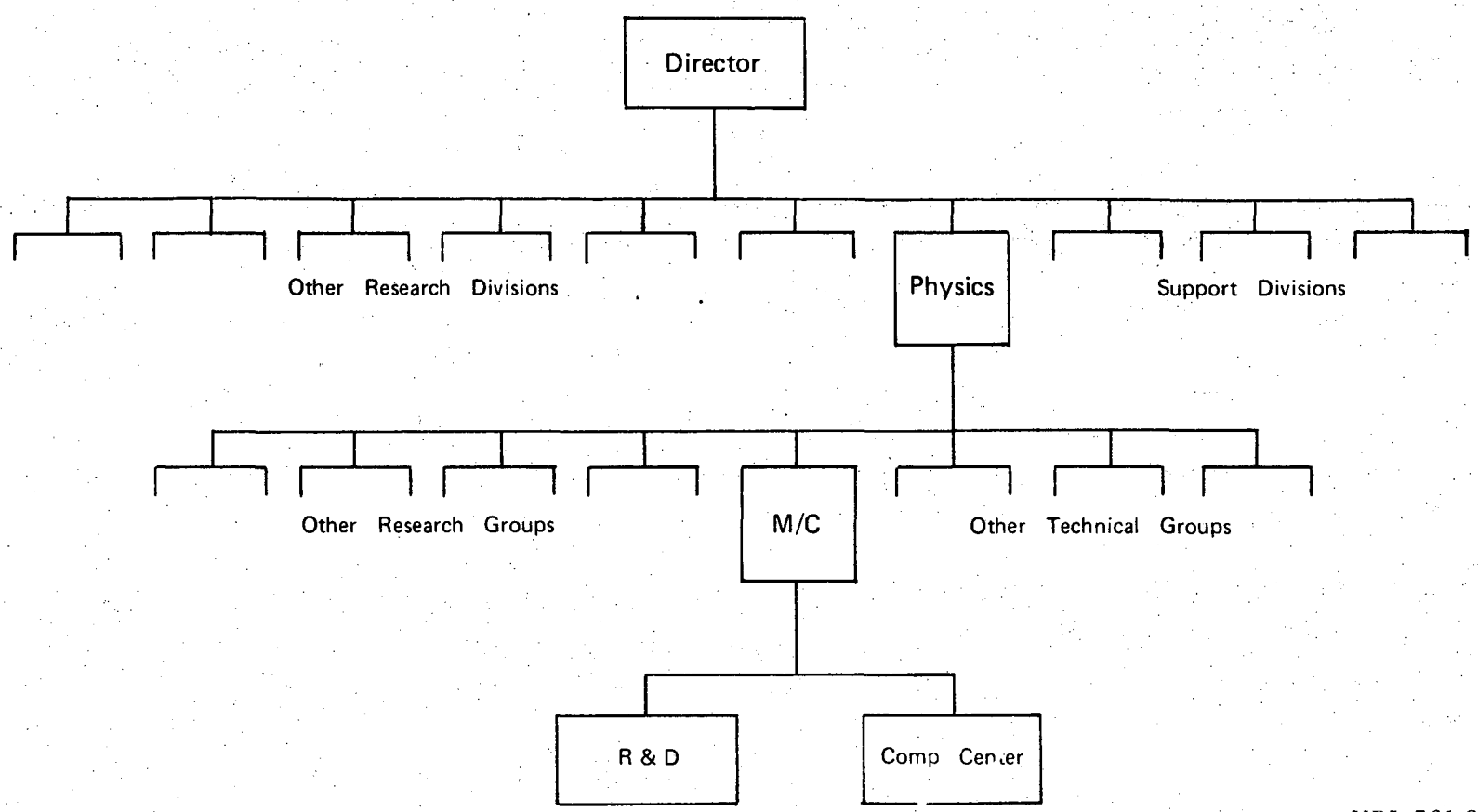
LBL – MIS COMPONENTS

Hardware	RECC Communications Network CDC - 6600, Disks, Etc. IBM - Data Cells, Tapes CDC - 7600 (Batch only)
Software	BKY Operating Systems Interactive Application Programs Procedural Controls and Data Bases Batch Application Programs
Personnel	**Management Users** **Programmers** Data Processing Staff Computer Operators

XBL 761-272

Fig. 1

LBL – ORGANIZATION



XBL 761-271

Fig. 2

TIME SEQUENCE

	On-line Facilities	Users	Other Events
	(Preliminary Data Bases)		
1972	Budgetor	Physics Div.	
1973	Report Generation	Math/Computing	Study Group on Interactive Computing
1974	Monitor	Chemistry Div. Bio-Medical Div.	Presented to Director LBL - 3089 Published
1975	Data Reduction	Engineering Div. Employee Services Div.	ADP Review Committee BMS Users Group

XBL 761-270

Fig. 3

LEGAL NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

TECHNICAL INFORMATION DIVISION
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720