

DOCUMENT RESUME

ED 064 660

24

CG 007 509

AUTHOR Baker, Frank B.
TITLE The Development of a Computer Model of the Concept Attainment Process: A Final Report.
INSTITUTION Wisconsin Univ., Madison. Research and Development Center for Cognitive Learning.
SPONS AGENCY Office of Education (DHEW), Washington, D.C.
REPORT NO TP-16
BUREAU NC BR-5-0216
PUB DATE Nov 68
CONTRACT OEC-5-10-154
NOTE 120p.

EDRS PRICE MF-\$0.65 HC-\$6.58
DESCRIPTORS *Cognitive Processes; *Computer Oriented Programs; *Concept Formation; *Information Processing; Learning; *Memory; Models

ABSTRACT

The model as currently developed consists of three major aspects: contexting, operation, and memory. The contexting aspects of the model are concerned with the higher level cognitive behavior associated with selection of appropriate behavior, maintenance of goals-directedness, and evaluation of completed behaviors. The operational aspects of the model are those behaviors which are performed during the execution of a concept-attainment strategy. Such behavior as creating a search criterion, comparing objects, and presenting concepts were considered operational. The memory component of the model was designed to facilitate the other aspects of the model as well as form the basis for a model of human memory. The memory was divided into three types of storage, each used for a particular purpose. The working memory was a buffer-type memory which received information from the external world and acted as a communication device for the transfer of internally created information. The short-term memory contained all of the information relevant to the attainment of a particular concept. The long-term memory will retain learning strategies and descriptive information necessary to implementation of these strategies and descriptive information necessary to implementation of these strategies.

(Author)

BR 5 0216
PA 2F

CG

THE DEVELOPMENT OF A
COMPUTER MODEL OF THE
CONCEPT ATTAINMENT PROCESS:
A FINAL REPORT



U.S. DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
OFFICE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.



WISCONSIN RESEARCH AND DEVELOPMENT

CENTER FOR
COGNITIVE LEARNING

CG 07 539

ED 054560

Theoretical Paper No. 16

THE DEVELOPMENT OF A COMPUTER MODEL
OF THE CONCEPT ATTAINMENT PROCESS:
A FINAL REPORT

By Frank B. Baker

Report from the Computer Simulation Project
Frank B. Baker, Principal Investigator

Wisconsin Research and Development
Center for Cognitive Learning
The University of Wisconsin
Madison, Wisconsin

November 1968

The research reported herein was performed pursuant to a contract with the United States Office of Education, Department of Health, Education, and Welfare, under the provisions of the Cooperative Research Program.

Center No. C-03 / Contract OE 5-10-154

FOREWORD

Contributing to an understanding of children's cognitive learning and improving related educational practices is the goal of the Wisconsin R & D Center. One of the Center's three major research and development programs—Conditions and Processes of Learning—consists of laboratory-type research projects, each concentrating on certain basic organismic or situational determinants of cognitive learning, but all united in the task of providing knowledge which can be utilized in the construction of instructional systems.

One reason that computer modeling has proved to be a valuable approach to gaining knowledge of cognitive processes is that explicit questions are raised—and must be answered—in programming. During the four years he spent modeling the concept attainment process, Professor Baker encountered increasingly complex questions whose answers required, finally, further basic research on learning before he could proceed further in computer technology. His project generated many ideas for gaining knowledge about the psychological processes in concept learning, research ideas to challenge the most inventive psychological experimenter. Although many questions remain to be clarified, the sophisticated model described in this Theoretical Paper represents a major forward thrust in computer technology.

Herbert J. Klausmeier
Director

PREFACE

The pioneering work of the group at the Mellon-Carnegie Institute of Technology, lead by A. Newell and H. A. Simon, had aroused considerable interest in non-numeric computing; however, the lack of readily available list processing languages limited the number of persons able to engage in this activity. In early 1962, Dr. R. K. Lindsay and J. H. Dauwalder, the University of Texas, programmed IPL-V for the Control Data 1604 Computer, thus making it available to the University of Wisconsin. With this new capability at hand, the present author decided to develop computer programs which simulated some aspect of cognitive behavior. Concept attainment was chosen for a number of reasons, paramount of which was that Dr. H. J. Klausmeier and his students at the University of Wisconsin had been working in this area for several years and would provide knowledgeable resource persons; Hunt's book [1962] provided an entry into an unfamiliar literature; and, finally, the concept attainment process appeared to be easy to simulate by means of a computer program.

The first program, which I wrote myself in the fall of 1963, served primarily as a device through which IPL-V was learned. The experience gained from this program convinced me that much could be accomplished and a computer simulation project was written into the original R & D Center proposal. A long-term project in this area was planned during the summer of 1964 with support from a graduate school research grant, and many of the fundamental ideas were developed that summer. Two years were spent in what seemed to be an endless loop of running subjects, writing programs, and redesigning the model. Since the initial program was written, considerable progress has been made; however, we are far from our goal of modeling the processes of human concept attainment.

The purpose of the present final report is threefold. It is first to describe where we currently stand in our research efforts and perhaps provoke some research in the areas we feel are important. The second purpose is to provide others with some insight into the nature and magnitude of the problems a neophyte encounters when developing computer models of cognitive behavior. The third is to illustrate that the "state of the art" is very primitive and much remains to be done.

I would like to emphasize the crucial role played by Mr. Tom Martin, who has programmed all but the first in the long series of programs. He has consistently worked to prevent the programs from becoming what programmers refer to as a "kludge" and has forced me to vastly sharpen my rather fuzzily conceived ideas. Many of my pet schemes have fallen apart and others have been coalesced into vastly improved schemes by his penetrating inquiries. He has also independently developed programs such as MIMIC which are significant contributions to the programming art themselves.

Mr. Alan Pratt collected the first two sets of protocol data and Miss Carin Cooper has collected the remaining five sets. Miss Cooper has also thoroughly reviewed literature in simulation and memory, thus relieving me of a tedious task. Mrs. Eva Bradford conducted the planning experiment and prepared the several levels of program narratives.

Although much remains to be done, this project has been terminated. The two reasons underlying this decision are: First, the "state of the art" limitations

in psychological research have been reached, making it impossible to collect the type of data needed for this computer model. Second, my background is in psychometrics, statistics, and computers; but progress in the computer model requires a learning theorist and I did not wish to change my professional orientation to fit the needs of the project. I hope to pursue the linguistic model on an informal basis as the approach is too interesting to drop completely.

F. B. B.

Madison, Wisconsin
1 September 1968

CONTENTS

	page
List of Tables and Figures	ix
Abstract	xi
I. Introduction	1
Goals of the Project	1
The Experimental Situation to be Modeled	3
Development of the Computer Model	3
"Think-Aloud" Protocols	4
Utilizing the Computer Program	4
Summary	5
II. A Computer Model of the Concept-Attainment Process: CASE Mark IV, Mod 2	6
Introduction	6
Assumptions	6
Representing Cognitive Process	7
Memory Structure	8
The Contexting Hierarchy	8
The Current Program—Mark IV, Mod 2	9
The Structural Details of the Computer Program	12
Symbolic Representation of Behavior	12
Memory Structure Mechanics	14
The Memory Entry Point	17
The Details of the Phase Lists to the P Level	18
Summary of the Chapter	22
III. Post Mark IV, Mod 2	23
Planning Experiment	23
Methodology	23
Results	24
Discussion	27
A Linguistic Approach to a Computer Model of Concept Attainment	27
IV. Summary and Conclusions	33
A Summary of the characteristics of the Various Versions of the Model	33
Modeling Considerations	36
Internal vs. External Information	36
The Memory Model	36
Attribute Structure	36

CONTENTS (continued)	page
Use of Protocols	37
Contexters	37
Programming Techniques	38
Research Ideas Generated by the Computer Model	39
The State of Our Art	40
References	42
Appendix A: Think-Aloud Protocol	43
Appendix B: Listing of CASE Mark IV, Mod 2	49
Appendix C: Narrative Description of Mark IV, Mod 2	107

LIST OF TABLES AND FIGURES

Table		page
1	Symbolic Representation of the Conservative-Focusing Strategy List as Used in Mark IV, Mod 2	7
2	Phase Lists to the P-Q Level as Created by the C61 Context Routine	13
3	Symbolic Representation of Routine P61	14
4	Definitions of Concepts Used in Experiment	25
5	Rating Scale Used to Measure Extent of Planning	25
6	Average Extent of Planning Score on Problem I Using 5-Point Scale with 3 Being "Moderate Planning"	25
7	Rating Scale to Assess Development of Planning Over Problems	26
8	Comparison of the Average Summed Planning Scores by Treatment for Each Problem Separately	26
9	Comparison of the Average Extent and Development of Planning for Each Scale Separately	26
Figure		
1	One of the 72 Possible Animal Configurations	4
2	Flow Chart of the High Level Contexter List S3	9
3	Partial Representation of the Circuit Memory Structure Created by the Computer Program Mark IV, Mod 2	15
4	Typical Modular Memory Module Used in Short-Term Memory	16

ABSTRACT

Development of the model described in this final report was begun to obtain a better understanding of the psychological processes underlying human concept attainment. The model has been based upon theoretical grounds, "think-aloud" protocols, and speculations as to the nature of concept attainment. The model developed is embodied in a computer program written in the IPL-V language. The program exists primarily as a device for expressing complex ideas and relationships in a convenient form. The current version of the program, called Mark IV, Mod 2, exhibits a wide range of the behavior observed in the "think aloud" protocols obtained from human subjects.

The model as currently developed consists of three major aspects: contexting, operations, and memory. The contexting aspects of the model are concerned with the higher level cognitive behavior associated with selection of appropriate behavior, maintenance of goal-directedness, and evaluation of completed behaviors. Such functions were labeled *contexting* as the associated computer programs essentially analyze the current situation and define the context within which the operational routines are executed. The operational aspects of the model are those behaviors which are performed during the execution of a concept-attainment strategy. Such behaviors as creating a search criterion, comparing objects, and presenting concepts were considered operational. The memory component of the model was designed to facilitate the other aspects of the model as well as form the basis for a model of human memory. The memory was divided into three types of storage, each used for a particular purpose. The working memory was a buffer-type memory which received information from the external world and acted as a communication device for the transfer of internally created information. The short-term memory contained all of the information relevant to the attainment of a particular concept. The short-term memory was constructed as a "circular memory structure" with a modular format; such a structure enables memory to grow as information is created. The long-term memory will retain learning strategies and descriptive information necessary to implementation of these strategies.

On the basis of the present model it appears that the most fruitful area of future inquiry would be development of a computer model in which the higher level information processing is handled at a verbal level. Such an approach has been labeled a "linguistic model of concept attainment." The salient features of the linguistic approach have been discussed but the corresponding computer programs not written.

Because existing psychological theories and published research do not provide the types of information necessary to further development of the model, a number of areas of fruitful research have been described. For example, the model suggests that the majority of the information processed by human subjects is internally created; hence new techniques are needed to elicit this information.

When compared to earlier versions, the current model has considerable power and seems sophisticated; however, when compared to human concept attainment it is very rudimentary. Hopefully the current model can serve as the basis for further interesting research.

INTRODUCTION

GOALS OF THE PROJECT

The fundamental purpose of the present project has been to obtain a better understanding of the psychological processes involved in the attainment of concepts by humans. The particular vehicle through which these understandings have been acquired is that of computer programs which serve as models of the concept attainment process. The use of computer programs as models of cognitive behavior has its origins in the early work of Newell, Shaw, and Simon [1958] who proposed that the "Logic Theorist" program was a model of human problem-solving behavior. Later work by this group and others has resulted in a well established field which is generally called simulation. But because the word *simulation* is widely used in fields other than psychology, most authors currently prefer to use the term *computer models*. The use of computer programs to represent a psychological process involves a number of factors which make the technique extremely valuable. First, because of the small steps by which computers proceed, it is difficult to write programs for something which is ill-defined. Hence, the computer forces one to probe very deeply into a psychological process in order that it be understood well enough to be programmed. Second, the computer program can be manipulated in a number of ways such that it can assist one in understanding the ramifications of the available knowledge about the processes involved. Third, the computer program serves as a repository of the understandings one has acquired up to any given point in time. The ideas are preserved in a language form which is unambiguous and open to study by others. Fourth, in any modeling process one is forced to make assumptions and in a computer program it becomes quite apparent what role these assumptions play in the model. Briefly, the computer serves as an extremely strict task master who forces one to commit to paper what one understands and, more importantly, what one does not understand.

Because one can approach a problem from many points of view the emphasis in a project of this type is a function of the interests of the investigator. In that the present author is firmly committed to "process" psychology rather than S-R psychology, no attempt has been made to study the relation of the model to stimulus materials. The interest here is in *how* a human subject performs the concept attainment task, not in what variables the experimenter can use to manipulate the subject's responses. Using S-R terms, the attempt is to model the intervening variables not the gross S-R connections. In the long term an understanding of the internal processes of a human subject holds considerably more promise for yielding new teaching techniques, classroom materials, etc. than does the traditional S-R approach.

At the present time there appear to be two general approaches to the design of a computer model of cognitive behavior, and these are referred to below as the basic premise approach and the surface approach (see Baker 1967). Under the basic premise approach one postulates a minimum set of operational rules or procedures and then designs a computer program around the successive application of the basic premises or their derivatives to the data presented in the program. The underlying idea is to ascertain how much interesting behavior can be generated by a set of basic premises devised by the investigator. Computer programs which perform pattern recognition [Uhr and Vossler, 1961], the sequence learner of Simon and Kotovsky [1963], and the concept learning system [Hunt, Marin and Stone, 1966] are clearly of this type. Such programs assume that a human has the basic premises and the ability to apply them built in or acquired from past experience, and that the investigator has made a reasonable assumption as to what basic premises are involved. An additional somewhat contradictory assumption usually involved in such programs is that the program, i.e. subject, begins a given computer run with no past experience relative to the data it will process by means of the basic premises, the "clean-

slow" assumption. A considerable amount of intelligence, to use the term loosely, is built into the program in regard to how to process the data, but none is built into the program in regard to previous presentations of the data. The resulting behavior of such a program is typically the construction and modification of decision-trees which are completely dependant upon the sequence of data fed into the program.

The surface approach tends to be associated with computer programs based upon human "think-aloud" protocols [Laughery and Gregg, 1962; Johnson, 1964]. Under this approach one attempts to use the protocols to ascertain the gross behavior patterns of humans in a particular problem-solving or learning situation, and then to write computer programs which reproduce these gross behaviors. Such programs can usually reproduce the overt behaviors observed in humans, and the computer-generated protocols can be reasonable facsimiles of corresponding human protocols under the same conditions. In contrast to the basic premise approach, the surface approach does not postulate any specific underlying mechanisms; rather it follows some well defined overall plan, such as the concept-attainment strategies of Bruner, Goodnow and Austin [1956]. In addition, it does not make the clean-slate assumption of the former in that knowledge about the data known to be relevant to a particular phenomenon is built into the program.

In that so little is known about how humans solve concept-attainment problems, it was not felt appropriate to make the assumptions necessary for the basic premise approach. In addition, if one follows the basic premise approach, one has very little likelihood of discovering new understandings or obtaining new insights as the total system is based upon a preconceived set of basic premises. However, starting from a surface-type approach one can change the system to match the new understandings acquired as one digs further into the problem rather than being constrained by an artificial set of initial basic premises. Throughout the current project a surface-type approach has been followed, but this is not to say that basic assumptions concerning such things as memory and other facets were not made. However, these assumptions have normally grown out of difficulties encountered within a computer program rather than being preconceived assumptions about the process itself.

In view of the investigator's commitment to the surface approach, a method of attack has been developed which allows one to elicit as much information as possible from the con-

struction of the computer program and at the same time "keep control" of the computer model. The procedure followed is given in the paragraphs below.

One begins with a computer program which corresponds to the behavior of an intelligent, experienced subject performing a particular type of concept-attainment problem after having had considerable practice. One then slowly builds into the model various types of behavior which are not as efficient as those used by the experienced subject and thus degrades the performance of the computer program. What one attempts to do is work backwards toward a computer program which will be as inefficient and stumbling as a person attempting the problem for the first time. By deliberately introducing a particular change into the computer program and then observing how the subject's performance of the task has been degraded by that change, one gets an understanding of the ramifications of each change made to the computer model. Such an approach is somewhat at variance with a large number of other simulation projects which have attempted to write a computer program for a subject who is initially very inefficient and inept at solving a problem and then improves to the performance level of an experienced subject. The latter approach appeared to the present author to be a more difficult task as at the current state of knowledge one does not have a good grasp of what causes the subject's inefficient, inept performance. It seemed much more appropriate to start from the experienced subject and slowly work backwards with a good understanding of each backward step and its effects upon performance. Thus, the amount of variability in the behavior built into the computer program is a function of our understanding of the concept attainment process. Eventually the computer program would become as inefficient as a human attempting a problem for the first time, but at that point one would understand the reasons for this level of performance and the processes by which a subject improves his performance over a sequence of problems. Then, one would expect to have an extremely good model of the concept-attainment process which could serve as a guideline for further educational-psychological investigations in the classroom.

The so-called backward approach has proved to be very feasible and quite rewarding in terms of the understandings of the concept-attainment process we have been able to obtain. The backward approach allows one to continually tie the computer model back to actual subject behavior and to insure that

what has been built into the computer program does in some manner represent actual subject behavior. It does not imply that the mechanisms are true representations of the subject's internal processes; however, the external behavior of the program can be observed in subject behavior.

THE EXPERIMENTAL SITUATION TO BE MODELED

The type of experimental situation for which a computer model has been developed is that described by Bruner, Goodnow and Austin [1956] and used extensively by psychologists. In such experiments, a subject is seated before a board containing a number of objects. Each object contains m dimensions and each dimension has n values; thus a complete board has n^m different objects. The experimenter explains to the subject that the objects can be divided into two mutually exclusive groups, members and nonmembers of the set defined by a classification rule (concept) consisting of a particular combination of dimension values. The experimenter designates an object (the focus object) as a member of the set. The subject's task is to discover the classification rule by choosing objects and having the experimenter designate their set membership. When the subject feels he knows the underlying classification rule, he tells it to the experimenter. If the rule is correct, it is assumed that the concept has been attained; if not, the subject continues until he can present the correct classification rule.

Bruner, Goodnow and Austin [1956] identified and labeled a number of strategies which subjects employed in this experimental situation and two, the "conservative focusing" and "wholist," are of interest to the present paper. Using the conservative focusing strategy, a subject chooses an object from the board identical to the focus object except for one dimension whose value has been varied. When such an object is designated as a member of the set, a "yes" object, he knows that the dimension is not included in the classification rule, hence is irrelevant. If the object so chosen is designated as not being a member of the set, a "no" object, he knows that the dimension is relevant, the dimension value of the focus object is included in the classification rule. In the conservative focusing strategy, the subject varies one dimension at a time and systematically checks each of the m possible dimensions. Thus, the minimum number of object choices to attain the concept is m , the number of dimensions.

Using the wholist strategy, the subject determines the classification rule through the intersection of all objects designated as members of the set by the experimenter. If an object chosen by the subject is designated as a member of the set, it will have certain dimension values in common with the focus object. Thus, a "yes" object is of value to the subject and a "no" object is of no value under this strategy. The subject continues developing the intersection of a series of "yes" objects and the focus object until he feels he knows the concept. Typically, under this strategy, subject will present a concept for designation by the experimenter after each "yes" object.

Although both strategies had a concept attainment, they differ in two major aspects. First, the method for choosing objects under the conservative focusing strategy is well defined and quite obvious to the observer, whereas under the wholist strategy the object choice mechanism is not so clearly observable. Second, the meaning of a "yes" and "no" designation of an object choice is reversed in the two strategies. In the conservative focusing strategy a "no" is the desired designation, and in the "wholist" a "yes" is the desired designation. Programs for both of these strategies were developed in the present project, but the primary emphasis has been upon the conservative focusing strategy.

DEVELOPMENT OF THE COMPUTER MODEL

The books by Bruner, Goodnow, and Austin [1956], Miller, Galanter and Pribram [1960], and Hunt [1962] and journal articles on the concept-attainment process were read to develop some understanding of what others had done in the concept-attainment area. On the basis of this initial investigation and the author's own intuitive understanding of how he would solve a concept-attainment problem, a computer program was written which would "simulate" concept attainment. The initial computer program called Mark I, Mod O was published in early 1964 [Baker, 1964]. On the basis of this program protocols were collected to ascertain how sophomore subjects from the University of Wisconsin, who had not previously seen this kind of problem, would solve it. The "think-aloud" procedure was used to collect data which was then analyzed by the project staff. On the basis of the analysis of the protocols, it was determined that the majority of the subjects very rapidly developed a conservative focusing strategy. Therefore, the computer program was redesigned to

incorporate a conservative focusing strategy. A computer program to model a specific subject has not been developed; rather many subjects, both male and females, have been used and the "normative" behavior of the subjects was modeled.

THINK-ALoud PROTOCOLS

The data gathering device used throughout the project has been the think-aloud protocol. As the experiment is being run, the subjects verbalize what they are doing and why they are doing it. Such a procedure has been a standard practice among those developing simulation programs even though it is not held in high esteem in many psychological circles. It was found quite early that the raw protocols were not very rich in information and a modified system was adopted in which the experimenter asked preplanned questions at certain points within the problem. The questions arose from the computer program and were designed to help fill the gaps in the program. For example, at one point the interest was in whether subjects remembered specific object choices, thus after the fifth object choice, they were asked to identify the second object chosen. Such information would not be yielded by the usual protocols, yet is easily obtainable through selective interrogation. A total of seven sets of protocol-gathering sessions, each involving five male and five female subjects were conducted, tape recorded, and reproduced in mimeographed form. In each of these seven runs a different set of questions was used to aid in development of the computer program.

Analysis of the early protocols revealed that the materials used by Bruner et al. [1956] and Klausmeier, Harris, and Wiersma [1964] involved psychologically dependent dimensions; subjects were unable to treat shape as an independent dimension. To overcome this problem new materials, consisting of animals whose dimensions were ears (long-short), neck (long-short), body (thin-fat), color (yellow-blue-brown) and tail (straight-bent-curl) were devised. Figure 1 presents one of the 72 possible animal configurations. Two of the dimensions were three-valued to overcome the artificiality of all binary valued dimensions. The new materials proved very successful and were used throughout the remainder of the project.

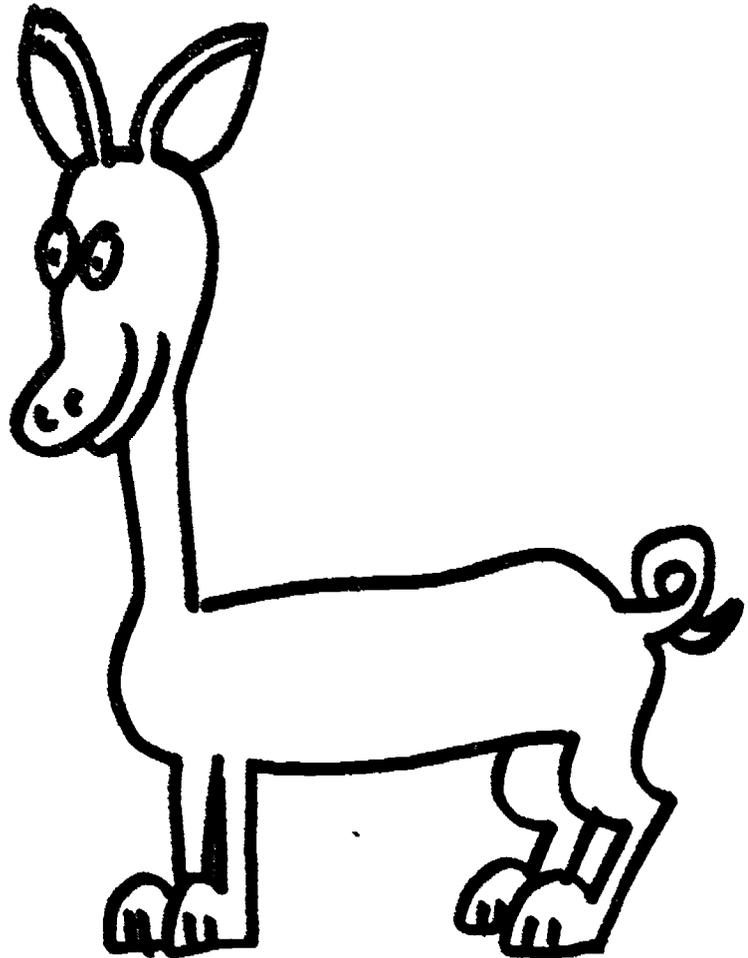


Fig. 1. One of the 72 Possible Animal Configurations

UTILIZING THE COMPUTER PROGRAMS

After writing a computer program to model the behavior of the subjects in the concept-attainment task, one spends a considerable amount of time analyzing the computer program itself in order to reduce it to a simple structure. It is very easy to become trapped with a computer program which is so complex and clumsy that it does not lend itself to the continual modification required by the so-called backward approach. Therefore, extreme care has been exercised to avoid a situation which requires periodically starting from the beginning.

A vast amount of effort has been devoted to the mechanics of the computer program itself in order to facilitate the modeling process. Data representation schemes, methods of communication within the computer program, and methods of executing the computer programs representing various types of behavior have been devised. What has been developed is essentially a small computer programming system within which a model of the concept-attainment process may be developed. Strange as it may seem, much of the understanding of the concept-attainment process has arisen out of attempts to develop a systematic computer program for use in the modeling process.

After a particular version of the computer program has been reanalyzed, rewritten, and polished to the point where it is a reasonable representation of our current understanding of the concept-attainment process, considerable effort is devoted to looking at the points where insufficient information exists. Questions are then devised to be asked during the next protocol run which will help clarify the particular point of concern. Thus, a large feedback loop exists in which attention shifts from subjects to computer program, to subjects, and then to the computer program again.

SUMMARY

The goal of the present project was to develop a model of the various cognitive processes involved in human concept attainment,

and it was toward the end of understanding these processes that the above procedures were directed. A lesser interest was in a computer program which would be an interesting tool in developing further understandings and insights into the concept-attainment process. There was little concern with developing a computer program which could generate large amounts of interesting data for later analysis or reporting in "See what our computer program can do" fashion as has been so typical of past efforts. The computer program is considered to be a repository of ideas about the processes involved in concept attainment expressed in computer programs in the IPL-V language. Recording ideas in this way may seem peculiar, but in a problem as complex as concept attainment it is virtually impossible to verbally represent all of the facets involved.

**A COMPUTER MODEL OF THE CONCEPT-ATTAINMENT PROCESS:
CASE MARK IV, MOD 2**

INTRODUCTION

In this chapter several levels of description of the most recent model of the concept-attainment process produced by the project staff are provided. One level will be rather gross so that the internal structure of the program can be seen without the clutter of mechanical details. The second will be at the sub-routine level to provide the reader with some appreciation of the formidable problems faced in implementing a computer model of cognitive behavior. In order to present the latter level it is necessary to discuss various mechanical details underlying the actual computer program. A full understanding of the model can only be obtained through a detailed study of the listing of the computer program which is presented in Appendix B. Although the computer program has been written in IPL-V [Newell, et al., 1964], a serious attempt has been made to describe the program without involving more than a bare minimum of the IPL-V language.

Assumptions

In order to program the present model a certain number of assumptions were made. The foremost of those was that the subject's perceptual processes are perfect; thus, the processes of perception were ignored. Even though a large proportion of the errors made in the concept-attainment process can be attributed to perceptual errors of one type or another they were not modeled. Secondly, the assumption has been made that memory is perfect; i.e., the computer model does not contain any forgetting processes. At some later point in time, it is anticipated that both decay and interference-type forgetting can be introduced into the computer program, but at the current time such mechanisms would obscure other more crucial aspects.

A major effort in the development of this

computer model has been devoted to eliminating the necessity for large numbers of input parameters and prestored information. At the current time only three types of information are prestored for use by the computer program. One of these is the dominant dimension values. Analysis of the initial protocols indicated that subjects possess a preference for certain dimensions and certain values of these dimensions. For example, it was found that female subjects invariably will utilize the dimension of color rather early in the solution of their problem, and certain people will prefer yellow over blue or brown. Built into the computer program is a selection device based upon probability values assigned to the dimensions and to their values. However, this information is used only at one point in the computer program and is not crucially involved in many of the psychological processes. It should be noted, however, that considerable variability in behavior can be accounted for by these dominance values. Three constants have also been prestored in the program which help mechanize certain types of within-problem variability. These constants are associated with the number of dimensions that a subject will use during a particular concept-attainment problem and the number of dimensions he will add to his initial approach when he discovers that it has not worked. The third and final prestored parameter is one known as an awareness factor. The protocols have indicated that many subjects use less than the total number of dimensions in their problem solution and that some of these people are aware of the fact that they are using less, others are not. Therefore, a flag is used to indicate whether the subject is aware that he is using less than the full number of dimensions in his approach to the problem. Other than these three types of information, all data gained by the subject is stored in memory as it is either received from the external world or created by the subject himself.

Representing Cognitive Process

In order to describe the computer model of the concept-attainment process it is necessary to explain a certain amount of symbolic representation used internally by the computer program. In that this project was influenced quite heavily by the earlier work of Bruner et al. [1956] and that of Miller, Galanter and Pribram [1960], the program is built around the idea of strategies, and the mechanics of the program are designed to implement strategies or plans. Quite early in the project it was discovered that the programming system must have the capability to minimize the impact of significant changes and simultaneously to maximize the ability to make such changes. Therefore, a pseudocode system and an interpreter, both using IPL-V, were developed as a reasonable solution to this technological problem [Baker & Martin, 1965a].

The strategy consists of an IPL-V list containing symbols representing routines which are to be performed as well as local symbols which indicate branches in the program. These lists, however, do not contain any IPL-V primitives and are not executable IPL-V programs. Table 1 contains a list of symbols representing a typical concept-learning strategy.

Each symbol on the list can be the name of a list of symbols; this representational form can be carried to any depth desired. These symbols are referred to as pseudocodes as they are merely abstract representations of psychological processes. In the current program there are three levels in the list structure which constitute a learning strategy. The highest level, the S level, is essentially an executive level description of the overall learning strategy. The second level consists of major procedures, the Z or D routines, which perform salient tasks such as hypothesis generation. The third and lowest level are the P's and Q's which are executed to perform the information-processing tasks necessary for concept attainment. The P's and Q's are contained within the Z's and D's and the Z's and D's are contained within S. Throughout the list structure a distinction is maintained between programs which do things, the Z's and P's, and those which provide decision-making information, the D's and the Q's. The former are analogous to the O routines and the latter to the T routines in TOTE units [Miller et al., 1960]. Only the lowest level routines can result in the direct execution of subroutine coded in IPL-V, and the higher levels serve only to hold together various combinations of executable routines. The underlying principle

Table 1. Symbolic Representation of the Conservative Focusing Strategy List as Used in Mark IV, Mod 2

S2	9-0	
	Z0	Process focus information
	C21	Create procedure Z7
	Z7	Establish search criterion
	D4	Determine whether subject should proceed
	9-2	No, error exit
9-1	Z1	Construct search criterion
	Z2	Select object from external environment
	C37	Create decision procedure D1
	D0	Determine whether object selected meets subject's needs
	9-1	No
	Z3	Experimenter designates set membership of the object choice
	C38	Create routine Z4
	Z4	Process information gained through object designation
	D1	Determine whether a concept can be presented
	9-1	No
	Z5	Form a concept
	Z8	Experimenter designation of correctness of concept
	C22	Construct procedure Z6
	Z6	Subject's reaction to designation of concept
	D3	Determine correctness of concept
	9-1	No
	0 0	Yes
9-2	X21	Error exit
	0 0	

is that the P's and Q's are the basic information processing capabilities possessed by a subject, and various tasks are performed by assembling the proper sequence of P's and Q's into the Z's or D's. The Z's and D's are then assembled into the strategy list (S). Such a strategy list is then executed by a special purpose interpreter [Baker & Martin, 1965a] which works its way through this list structure until it finds a routine which is executable, namely at the lowest P or Q level. It executes the routine and then returns up to the next higher level to ascertain the next executable routine. Fundamentally the interpreter is an ordinary IPL-V recursive program which calls upon itself to work its way up and down the branches of the list structure representing the learning processes.

Memory Structure

Quite early in the development of the concept attainment program, it was determined that memory plays a crucial role in the concept-attainment process, and it was necessary to design a rudimentary model of memory. Under this model, memory was divided into three major aspects: working memory (WM) which is a temporary, buffer-type memory; short-term memory (STM) in which all information relative to a given problem solution is stored; and long-term memory (LTM) where the subject stores information which is to be retained over a longer period of time. Thus, the partitioning of memory is a function of the duration of time over which the information is to be retained. Such a three-part memory does not correspond directly to the memory model ordinarily used by psychologists, which involves only a short-term memory and a long-term memory. Most of the functions of what was called short-term memory are embodied in their long-term memory. However, investigation of the protocols seems to indicate that subjects retain information about a problem only long enough to solve that particular problem and then do some re-coding to save the salient features over longer periods of time. Therefore, it was suggested that there is a distinction between short-term and long-term memory which psychologists do not normally recognize.

The short-term and long-term memories have a highly interconnected net structure which is developed by the program as information is acquired. The dynamic nature of the memory structure is an important feature of the three-level model, but discussion of the actual mechanics of this system will be deferred to a later section in the present paper.

The Contexting Hierarchy

The internal organization of computer models constructed under either the basic premise or the surface approach is focused upon implementing a rather specific psychological phenomena and does not take into direct account a higher level of cognitive behavior, namely that which in some sense directs, maintains, and evaluates the overall problem-solving or learning behavior of a human subject.

In order to clarify this issue, let us briefly examine a problem-solving or learning experiment as it is usually conducted. In such an experiment there is a fairly typical sequence of events which transpire in roughly the following order:

- (a) The experimenter explains the nature of the task, the characteristics of the experimental materials, and the types of products the subject is to produce.
- (b) The subject relates the given information to what he already knows.
- (c) Once the subject has assimilated the information to his own satisfaction, he embarks upon an approach to the task which is resplendent with errors and inappropriate decisions; false strategies, and unproductive acts; nonetheless, his behavior is goal directed.
- (d) The subject is able to evaluate, in some sense, how well he is doing by means of both internal and external clues.
- (e) With sufficient experience on the same task, the subject is usually able to modify his own behavior to the point where he becomes proficient at the task and his once clumsy performance becomes smooth and effortless.

In that such a pattern of behavior is essentially independent of the particular task, it is very difficult, for the present author at least, to conceive of a realistic model of human behavior whose internal organization does not provide for some form of a central executive to account for this communality. The relevant issue is the form of this central executive and the internal organization of a computer program necessary to represent it in a computer model of cognitive behavior. Unfortunately, it is extremely difficult to obtain direct evidence from either protocols or psychological experiments from which to develop a model of such a central executive. In addition, how to create one is not obvious; as Newell [1962] said, "In attempting to create such a central organization we found—as we had in the problem of communicating strategies—that we had no concepts and no formal language to discuss the variety of results and their uses [p. 410]."

In earlier editions of the program such a central executive was confounded with the strategy list; however in the current version, the supervisory or executive aspects of the program have been separated from those of the operational aspects. The C routines represent this executive function and constitute a hierarchy of control whose role changes as a function of the stage of the task performance. Because the function of the supervisory program changes often, the term *contexter* is probably more appropriate for these routines than *central executive*, which carries an unwarranted connotation of a single supervisory program. Although the role of a contexter is

a function of the situation in which it operates, there is nonetheless an underlying communality throughout all levels of contexters which can be described by a series of questions which a context routine attempts to answer; i.e., (a) What is the current situation? (b) What does it mean? (c) What could be done? and (d) What will be done? Thus, whether the contexter is dealing with a gross overall plan of approach to a task, or to some small operation within a subtask, the fundamental framework of a contexter routine is invariant; what varies is the situation in which the contexter functions and the procedures by which it attempts to answer these questions. It is worth noting that the definition of the current situation includes not only the available data but also the sequence of behaviors leading up to the present point in time. The result of calling a contexter routine is to execute some behavior for which an appropriate context has been established.

A contexter may be said to create a plan or a strategy for behavior. At high levels in the model, it creates a plan for overall behavior such as the S list, and at low levels it creates plans for very specific actions such as P lists. Such a planning hierarchy was first envisioned by Miller, Galanter, and Pribram [1960] when they suggested the existence of plans which create plans. Because of the rather complex interrelationship between the contexting programs and the strategy lists, a detailed discussion is deferred to a description of the actual computer program itself.

THE CURRENT PROGRAM—MARK IV. MOD 2

The preceding paragraphs have acquainted the reader with some of the major considerations in the design of the simulation program; the overall picture of the operation in the current version of the computer model of concept-attainment is as follows. First, the experimenter verbally describes the experimental situation to the subject—what the experiment is about, what the board looks like, the dimen-

sions on the board, and their values. The experimenter also indicates to the subject that he is to select objects which the experimenter will designate as to their set membership. When the subject feels he understands the concept, he is to present it to the experimenter for designation. Upon receipt of the instructions, the subject proceeds to try to attain the concept. The computer program is set up with an initialization phase which utilizes the subject's past experience and his particular characteristics, namely some of the constants mentioned earlier and the dominance values, to establish an initial set of conditions within the subject. After completing this initialization phase, the computer program creates a search criterion and locates an object in the external environment which it also feels is a member of the set. If the object found meets the requirements of the subject's search criterion, it is presented to the experimenter for designation. After receiving the designation, the subject processes the meaning of *yes* or *no* in light of his own understanding of the problem. If the subject feels he can present a concept he proceeds. However, in most cases the subject takes several object choices before he has enough information available to decide whether or not he understands the concept. Therefore, at this stage, the computer program returns to creating a new criterion and locates other objects from the board visible to the subject. The final phase of the program occurs when the subject feels he has enough information to present the experimenter with a concept for designation. If it is incorrect, the subject then has to construct a reaction to this incorrect designation and return to the first phase in which he searches for additional objects that will enable him to ascertain the correct concept. If the concept is correct, the problem is terminated and the subject then evaluates what he has accomplished during the course of this particular problem. So much for an overview of the concept-attainment process. Let us now turn our attention to a discussion of the flow chart which is given in Figure 2.

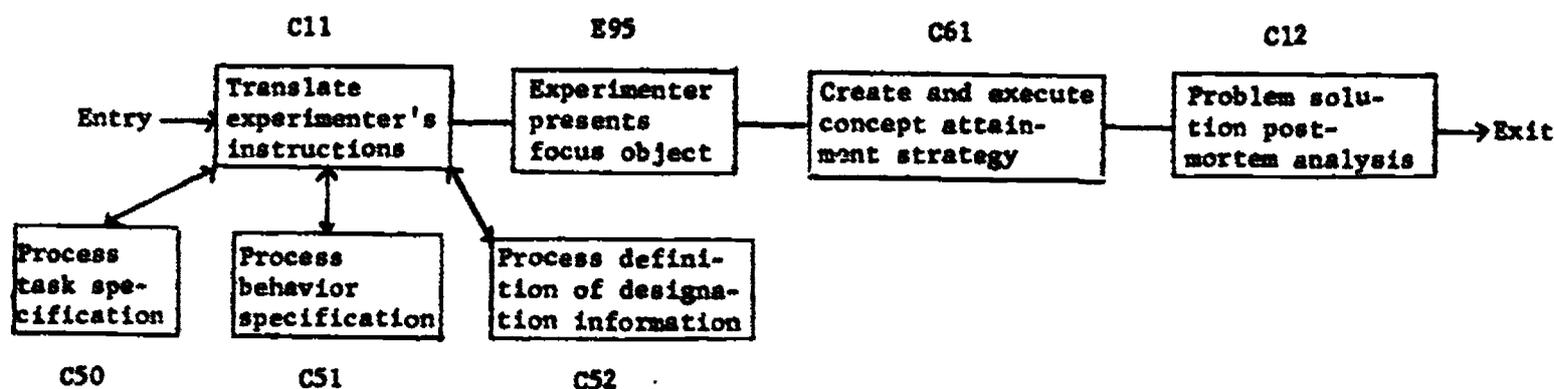


Fig. 2. Flow Chart of the High Level Contexter List S3

This flow chart will be discussed in terms of the particular routines which are in the flow chart. No attempt to go into all the programming or mechanical details is made, but a verbal description of what occurs within the program is given and any relevant assumptions which have been made by the particular program are indicated. The highest level program list in the computer model is called S3 which is the high level contexting list. This program essentially contains a gross description of what is to occur in the attainment of a concept and consists of four computer programs: C11, which creates a tentative strategy from the experimenter's instructions; E95 in which the experimenter presents the focus object to the subject; C61 which creates and executes a strategy phase-by-phase until the concept is finally attained; and C12, a problem solution postmortem analysis in which the subject ties together what he has done into a workable learning strategy for future use. It should be noted that C11, C61, and C12 are contexting-type routines.

The messages from the experimenter explaining the problem to be solved have been coded in terms of attributes and particular attribute values which essentially describe major behaviors; such coding gets around the syntactical-semantic analysis problems normally associated with translating English language into a computer program. This particular problem was completely by-passed because it is a major research project in itself. Rather, experimenter messages have been created describing behavior in terms of attributes and values so that the long-term memory can be searched to see whether other behaviors carrying this description are available for assembling into a strategy. Such an approach is rather crude, but it enables a form of translation of instructions to behavior to be introduced into the model.

The C11 routine accomplishes the translation of experimenter messages to a description of a rough skeleton learning strategy. In the first problem attempted by a subject, the C11 routine creates a skeleton strategy list which describes the gross behaviors necessary to attack this concept-attainment problem. In subsequent problems, C11 searches the long-term memory to obtain a strategy list from a previous problem which can be utilized as the approach to the problem. C11 consists of three major subroutines, each one associated with a different type of message which can come from the experimenter. The first subroutine, called C50, creates a problem list, indicates what the problem is, and stores descriptive information such as which problem

it is. The second subroutine, called C51, creates a description of a skeleton strategy for solving the problem. The skeleton strategy does not include all of the behaviors necessary to attain the concept, but stores the major framework of the experimental situation contained in the experimenter's messages. The third major subroutine, C52, is designed to store specific types of information which the experimenter presents, such as that he will designate set membership of an object by the words *yes* and *no*. Thus, C11 takes in a particular message from the experimenter's verbal instructions and translates it into descriptions of particular behaviors which the subject must perform in order to attain a concept. Which of the three C50 routines used is a function of the message that has been received from the experimenter, and there are decision processes within C11 enabling the program to call the proper subroutine for a given message. There are two major outputs of the C11 routine, one of which is the beginning of the short-term memory structure which the program will grow during its solution of the concept-attainment problem. The second is a skeleton strategy list containing symbols representing the major procedures within the concept-attainment task as indicated by the experimenter. The initial point of the short-term memory, the symbol L100, represents the problem and carries the description of the problem obtained from the experimenter messages. The skeleton strategy list contains symbols which are not executable routines at this point, but are merely symbols which hold descriptions of the kinds of behaviors necessary to accomplish the task. If the subject has previously attained a concept, rather than constructing a skeleton strategy C11 locates the recently used strategy in long-term memory and places its name in memory.

When C11 has been completed the subject knows in a general way how he is to perform the concept-attainment task. To specify a particular concept-attainment task, the experimenter must identify the focus object, which is an exemplar of an object belonging to the set defined by the unknown classification rule. Routine E95 performs this function by placing the name of the focus object in the subject's working memory. The name of the focus object is accompanied by descriptive information indicating the set membership of the focus object. Upon completion of this action, control of the program returns to the subject.

The major contexting routine in the current program follows E95 and is called C61. At the time this routine is executed, the short-term memory contains the symbol L100 which

represents the problem to be solved, and the contents of the working memory are either the name of the skeleton strategy or a previously developed strategy. The routine checks a flag to determine whether this is the first problem it has solved or not. If it is not the first problem, C61 assumes that the contents of working memory are a fully developed strategy which is given to the interpreter for execution. If it is the first problem, the program must translate the description of the skeleton strategy created by C11 into executable behavior. The C61 routine creates a symbol for the strategy list and then creates a symbol for the first phase of the strategy. Having created a phase symbol, it then searches long-term memory to find a routine whose behavioral description matches that of the first routine on the skeleton strategy list. In this situation it will be routine Z0 which receives the focus object and its designation from the experimenter and stores them on the problem list. Because Z0 receives information from the external environment, it is followed by a contexting routine. It should be noted that one of the rules of the computer program is that a contexter routine follows whenever information is received from the external world. In this case contexter C21 is inserted on the phase list after routine Z0. The function of the contexter C21 is to create routine Z7 which uses the focus object and characteristics of the subject to establish the initial working hypothesis. Because C21 will create routine Z7 at execution time, the C61 contexter will place a symbol on the phase list to hold a position for this future routine. The symbol for Z7 is followed by routine D4 whose function is to determine whether the information yielded by Z7 is adequate. If it is adequate, the next phase can be constructed; if inadequate, C61 will have to attempt to create a new Phase 1. The basic process embodied in C61 is one which creates a phase list containing routines matching the behaviors described by the experimenter. Routines processing information from the external world are followed by decision routines which ask "Can the strategy proceed?" These decision routines also terminate the phase list. Thus C61 creates a phase, executes it, and, if it receives the go-ahead, creates the next segment of behavior.

Phase 2 created by C61 is the object-choice phase, consisting of routine Z1 which creates a search criterion by varying one or more dimension values of the working hypothesis, routine Z2 which chooses an object from the board, C37 which establishes the test conditions for the last routine, D0, which ascertains whether or not the object choice

meets the subject's needs. If it does, C61 continues to Phase 3; if not it returns to Phase 2 and re-executes it.

Phase 3 is the experimenter designation of the object phase, containing routine Z3 which presents the object to the experimenter for designation and routine C38, a contexter that establishes the routine Z4. The procedure Z4 processes the information provided by the experimenter's designation of the set membership of the object chosen. Following the logic of the conservative-focusing strategy, Z4 flags the dimension or the dimension value as relevant or irrelevant, depending upon whether or not the object was designated *yes* or *no*. The last routine in Phase 3 is routine D1 which ascertains whether or not a concept can be presented at the current point. D1 checks each of the dimensions of the working hypothesis and determines whether the subject considers them relevant, irrelevant, or untested. If all dimensions have been flagged by Z4 as either relevant or irrelevant, sufficient information is available for the subject to present a concept to the experimenter. But D1 is also conditioned if the subject is using less than the total number of dimensions and all those he is working with have been flagged. If a concept can be presented, Phase 4 is entered. If not, the program returns to Phase 2 and executes Phases 2 and 3 over again.

Phase 4, which is the final phase, consists of procedures Z5, Z8, C22, Z6, and D3. In routine Z5 the subject searches the dimension values of the working hypothesis for those values which are relevant and from the relevant dimension values creates a concept, i.e., a list of dimension values which it believes defines the classification rule. The next routine is Z8 which presents this concept to the experimenter for his designation. Upon designation, a contexting routine, C22, is executed because Z8 brought in information from the external world. C22 is very similar to C38 in that it will create a situationally dependent routine Z6 for utilizing the information provided by the designation of a concept; Z6 is only created if the concept is incorrect as a subject then has to ascertain what is wrong with his prior behavior. Z6 primarily looks for dimensions which have not been involved in the concept itself. In other words, it looks through the dimension and dimension values of the focus object searching for untested dimensions. If it finds untested dimensions, it then adds them to the working hypothesis list. For example, if a subject initially only used three out of the five possible dimensions, Z6 will add one or more dimensions as a function of the number of untested dimensions

available and the value of the parameter K97 which specifies how many dimensions are to be added. If Z6 discovers that all the dimensions have been flagged and the subject still has not attained the concept, it then assumes that he has misflagged a dimension. Instead of adding untested dimensions to the working hypothesis, it will unmark dimensions on the working hypothesis list so that a new search criterion will include those which have been used in the past. The last routine in Phase 4 is routine D3 which ascertains whether or not the subject should continue to attempt the problem. The phase lists created by C61 are presented in Table 2.

Upon the completion of Phase 4, C61 realizes it has a list of executable routines for all the behaviors from creating a search criterion to testing the concept; therefore, it treats this list as a substrategy, i.e., the strategy is complete except for the initialization phase, but because nothing has to be reinitialized, this subprogram can be executed as if it were a total strategy. Hence, if the concept were incorrect, the substrategy would be executed until the concept is attained.

All four phases created and executed by C61 from the skeleton strategy list have the same general structure. There are one or more major procedures at the Z level. When information is received from the external world, a contexter routine creates a situationally dependent routine to determine the meaning of the external information. The final routine in each phase is a D routine which essentially asks whether the subject can proceed or must return to the object selection phase to get further information.

In review, C61 proceeds step-by-step and performs the behaviors indicated by the skeleton strategy as being involved in the concept-attainment process. It picks its routines from long-term memory by comparing the description of what needs to be done with the description of the capabilities of routines stored away in long-term memory. Each phase is created to handle logical units of behavior within the concept-attainment process, and the phase is given to the interpreter for execution. If progress can be made, C61 will move on to create and execute the next phase within the problem, and repeat this process until it can present a concept. If, upon completion of Phase 4, the concept is correct, the program is terminated; if incorrect, C61 treats what it has already created as a strategy and executes the strategy until a concept has been attained.

The routine following the successful completion of a strategy is a postmortem analysis

routine called C12. Because the model has not progressed beyond the within-problem analysis stage, this aspect of the process has not received more than cursory attention. At the current time the task of C12 is to tie together the total strategy which has been created rather piecemeal by C61. It places symbols representing Phases 1, 2, 3, and 4 all in a common strategy list and puts links from each of the phases to Phase 2. It was intended to have C12 do an analysis of the execution of the program, ascertaining whether there were unnecessary behaviors and smoothing out a successful strategy. C12 also stores the successful strategy on the long-term memory so that it can be used by C11 when a subsequent problem is attempted.

The computer program described above attempts to model the salient features of a subject performing a concept-attainment task. The initial stages are quite slow because experimenter instructions must be understood, a rough idea of how to proceed must be constructed, and the subject proceeds step-by-step. Once the full process has been gone through the pace quickens since the subject repeats behaviors established during the early phases, thus eliminating most of the C61 level contexting previously required. Hopefully what has been developed is a reasonable framework within which investigations of the concept-attainment process can continue.

THE STRUCTURAL DETAILS OF THE COMPUTER PROGRAM

Symbolic Representation of Behavior

In order to present a detailed discussion of how the computer program attains concepts, it is necessary to elaborate further upon the internal structure of the computer program. Attention will be given to the representational scheme for subroutines, the attribute system, and the memory structure.

Let us examine one process within the strategy list, routine P61 which appears in procedure Z0. The list of symbols representing the routine P61 is given in Table 3. The symbol P61 represents a nonexecutable routine whose function is to hold the description of the executable routine P60. Thus, the P61 symbol is a pseudocode whose description defines the context within which the executable routine will function. A given routine may appear as the executable routine of several different pseudocodes. Such a feature permits the development of powerful generalized routines which are independent of a particular

Table 3. Symbolic Representation of Routine P61

P61	9-0		Pseudocode
	P60	0	Executable Routine
9-0	0		
	A1		Input Attribute
	V1	0	
		M1	Working Memory
		F3	0 M1,N Flag
	A2		Output Attribute
	V2	0	
		M10	Memory Entry Point
		A20	0 Focus Object Attribute
	A3		Process Description Attribute
	V3	0	
		A305	0 Remembering

context. The description list 9-0 of the pseudocode P61 contains attribute A1 whose value V1 is a list of the inputs to P60. The attribute A2 has on its value list V2, the names of the locations at which the outputs will be placed. The attribute A3 has the symbol A305 on its value list which describes P60 as a routine involving remembering. The descriptions held by the pseudocode can be used by higher level context routines to ascertain the characteristics of the routine. Such a system provides a rudimentary description of behavior. Whether they are contexters, strategies, procedures, or processes, all routines are represented in the computer program by means of this scheme.

The special interpreter [Baker & Martin, 1965a] extracts the inputs from the description list of the pseudocode and places them in the IPL-V Communication Cell H0, it then executes routine P60 in IPL-V. The outputs created by P60 are left in the location named on the output list of P61. Except for the memory processes, all P and Q level routines leave their outputs in working memory.

The pseudocode and interpreter system permit the development of general purpose programs which can be used in a number of different situations. For example: Processes P61, P62, P63, P64, etc. contain the same executable routine P60 and differ only in the information contained on their respective input-output lists. Thus, P61 may store the focus object in short-term memory, whereas P62 may remember the experimenter designation of an object choice. Such a scheme was designed to permit eventual development of contexter routines which will place information on the input-output lists of a pseudocode rather than having the human programmer code in the information, a step toward programs which can create programs.

Memory Structure Mechanics

Much of the design of the computer model is dependent upon the mechanics of the three-level model of memory employed. In the paragraphs below the working memory and short-term memory are examined in detail. Long-term memory was never constructed.

The working memory consists of only two cells—M1,N and M1,D—which are on a list called M1. M1,N contains the name of a particular piece of information, for example, the name of an object chosen from the external world or the name of a search criterion by which the subject is scanning for objects having certain characteristics. M1,D contains what we have called an unattached or dummy description list (DDL). The dummy description list contains a description of the symbol within the M1,N portion of the short-term memory, the idea being that subject has not attached the description to the item itself, but rather he has created a description which later routines will process and attach either to the element named in M1,N or to some other item of information. The rationale is that the dummy description list corresponds roughly to a chunk as discussed by Miller et al. [1960]. However, this chunk will not necessarily be attached to the item named in M1,N.

The working memory plays two roles within the simulation program. In the first role, it acts as an input buffer from the external world. All information from the experimenter such as the focus object or the designation of an object choice or a concept comes to the subject through the working memory. For example, in the case of the focus object, M1,N would contain a symbol representing the focus object and M1,D would contain a dummy description list which designates this object as being a member of the set. The information left in working memory is then acquired by a subsequent processing routine and can be stored or processed further within the concept-attainment processes themselves. The second role of the working memory is that of an internal communication device between various subroutines within the computer program. In the early days of the present computer program, it was felt that most of the information processed was obtained from the external world. However, protocol analysis very quickly showed that a major portion of the information processed by the subject was created internally, therefore a requirement existed for some means of temporarily storing a piece of information so that a series of processes could work upon it. Most of the low level routines within the computer program receive their information

```

L100 9-0
X1 0
9-0 0
A300 External environment
V300 0
E9 9-1 Subjects internal M13 9-2 Structure of the external environment
01 representations Color E10 9-3
Q2 of the external E11 E11 9-4 0
Q3 environment E12 E12 9-4 0
. . E13 E13 0 Brown A77
. . 9-2 0 9-3 0 V77
. . E11 3624 0
Q72 0 N10 + 50 N11 + 50
9-1 0 E20 E12 Dominance
A101 N20 + 60 Dominance N12 + 50
V101 0 E30 E13 Values
M13 0 N30 + 10 Values N13 + 25
E40 N40 + 35
A302 Strategy S2 9-1 Strategy list
V302 0 Z0
S2 0 . .
. . D3
. . 0 0
. . 9-2 0
A18 Search criterion
V18 0 3671 9-2 Search criterion
3671 3671 E11
2033 2033 E32
0112 0 0112 0 E41
9-2 0 9-2 0 A16
A16 V16
V16
Objects found
0 C10 9-3 Object 10
C10 E11
C27 E22
C11 0 E32
E42 0 E42 0
9-3 0 How object found
A26
V26 0
2163 2163 9-4 0 "From-to list"
3011 0 9-4 0 A9
A9 V9 0 Dimension varied
E10 Color E10 0 Color
A10 from
V10 0 E12 Blue E12 0 Blue
A11 to
V11 0 E11 Yellow E11 0 Yellow

```

Fig. 3. Partial Representation of the Circular Memory Structure Created by the Computer Program Mark IV, Mod 2.

from the working memory and, having processed it, leave their outputs in working memory. In many cases a subsequent routine remembers the information in short-term memory. From a programming point of view, the working memory, acting as an internal information buffer, solves many mechanical programming problems which otherwise would become enmeshed in the idiosyncrasies of IPL-V itself. Working memory is very similar to the HO Communication Cell of the IPL-V except that it is in the program rather than the programming language.

The second major portion of the memory structure is short-term memory which contains all of the information relevant to solving a particular problem. The current structure of this memory is one that can best be described as a highly interconnected net. The short-term memory structure shown in Figure 3 is designed to grow as the information in the problem is acquired. However, the growth is constrained by a modular memory structure as shown in Figure 4. For example, in Figure 3 the symbol L100 represents the problem which is currently being solved. On the description of L100 is an attribute A302 whose value describes the

problem by means of the current strategy. Describing the current strategy is an attribute A18 whose value list contains the names of the search criteria used. Each search criteria has an attribute A16 whose value list contains the objects which have been found. Describing each object which has been found is a "from-to lists" containing the dimension that had been varied in order to find this particular object. The memory structure is, in reality, a tree; however, the information on one branch of the tree is not unique to that branch. For example, the from-to list on one branch describes what dimension was varied to find a particular object is also contained on another branch of the tree which describes how the search criterion was created from the focus object. The highly connected net of memory was originally conceived of as what one might call a circular memory, in that once it has grown over a period of time there is no real beginning and no real end to it because information is crosslinked and interlinked so heavily that the structure of the tree has become obscured. At the present time, because of the single problem solution involved, the tree does not get overly complex.

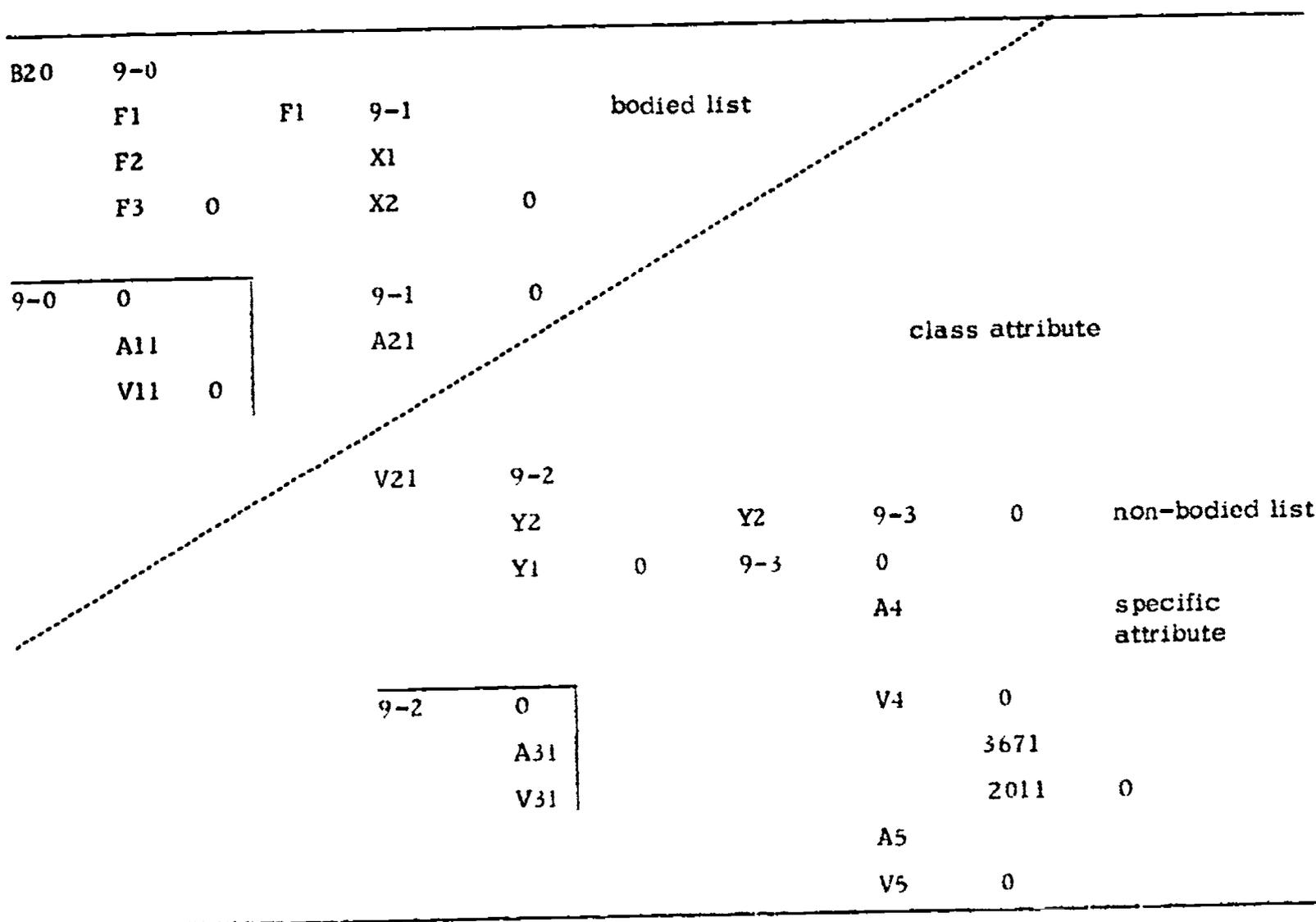


Fig. 4. Typical Modular Memory Module Used in Short-Term Memory

For clarity, the symbols in Figures 3 and 4 are IPL-V regional symbols, but in the actual program these symbols are created by the program as the information is acquired. The memory net given in Figure 3 does not exist in short-term memory prior to execution of the program. The computer program only possesses the capability to create memory as it needs to store information. Such a memory capability differs considerably from that usually employed in computer programming where the programmer accounts for every memory location used. The dynamic memory structure originating here represents a first step toward a computer program which can store and recall information without outside intervention.

Various attributes under which information is stored constitute some of the basic assumptions of the current computer program. These attributes are felt to be an intermediary step between current status and desired status, in that the investigators understand neither how to describe behavior nor how people store information in memory. Therefore, this is an approach to these particularly difficult problems. In order for the memory recall processes to work, a memory structure was developed which enabled the program to tell when it reached a point at which information was available, hence class and specific attributes were devised. In Figure 4 the attribute A21 is a class attribute as its function is to hold a series of specific descriptions on its value list V21. The symbols on the body of list V21 are dummies whose sole function is to hold a description list containing specific information. Thus, the description of list Y2 contains specific attributes A4 and A5 whose values are lists V4 and V5. Such a memory structure is symmetrical above and below the dashed line, thus permitting a single set of remember and recall routines to function at all levels. It should be noted that usable information can only be obtained from the specific attribute level, all higher levels are merely symbols representing larger units of information. The class attribute value list V21 is also time ordered with the most recent data at the top of the list. The description lists contained to the left of the brackets describe a particular list and are themselves of the same modular structure.

The long-term memory has not been designed because the between-problem variability stage of the project has not been reached. From the initial protocols and from the current computer program, it appears that what is stored in long-term memory are strategies and key pieces of information necessary to

execute a particular strategy. It does not appear that a great wealth of detailed information is ever stored in long-term memory. Later development of the computer program will be devoted to studying the program of long-term memory and trying to realize an adequate model for this aspect of the memory structure.

The Memory Entry Point

One of the major problems in developing the circular memory structure was some means for entering the memory or at least keeping track of a present position in the memory having entered it. The device invented was called the "memory entry point." If one looks at the concept-attainment process, it becomes evident that as the subject goes through the various phases the information created is normally about a particular point within the process. For example, if an object is chosen from the external environment, the subject spends a fair amount of time processing various types of information about this object—what dimension was varied, what the experimenter's designation of the object was, etc. Much of the information to be stored or recalled is related to the particular item. Therefore, the object chosen serves as the memory entry point. As the process moves on to another piece of information, for example, the creation of a concept, the memory entry point changes. However, this change is normally either upward or downward on the branch of the short-term memory so that the memory entry point is really a push down, pop up list in which the subject keeps track of where he has been in memory. The problem of how to initially enter the memory structure has not been resolved, but once in the memory structure the computer program can keep a record of where it has been. The problem of initial entry was also encountered by Quillian in his memory net project; however, he chose to avoid it and entered memory at arbitrary points by manual means. Because several of the context routines have to revert to previous levels, there are two small routines, C30 and C40, which add names or take them away from the memory entry point list. The memory entry point technique is not a very satisfactory solution; however, at the present time it is a feasible one to program until a better understanding of memory processes is obtained.

Remembering of information in short-term memory and recalling of information from it are accomplished by generalized executable processes P60 and P500, respectively. The

pseudocode containing P60, say P61 in Table 3, has on its input list the symbol representing working memory (M1) and a flag indicating whether M1,N or M1,D is to be remembered. The output list of the pseudocode contains the symbol representing the memory entry point (M10) and the descriptive attribute (A20) under which the information is to be stored. All storage operations are assumed to describe the symbol named in the memory entry point, and the basic process is unaffected by the type of information stored. The distinction between storage under a class attribute and storage under a specific attribute is handled within the basic P60 routine, thus the program does not need to concern itself with this distinction. The basic recall routine is P500 which is the converse of P60 and shares much of its internal programming.

The communication of the subject to the experimenter is one of the points in the concept-attainment process of little concern from a psychological point of view. Therefore, all communications to the experimenter take place through a special output register called E1 into which the subject puts information and from which the experimenter removes information in order to designate objects or concepts. It is mechanically simple, but not necessarily psychologically sophisticated.

THE DETAILS OF THE PHASE LISTS TO THE P LEVEL

Having described some of the underlying mechanics, let us turn our attention to the details of the phases created by routine C61. Some of these phases will be skipped over rather lightly, others will be described in some detail so that the reader may get the "flavor" of the program without excessive tedious detail. Table 2 above presented the lists representing behaviors to the P level which are constructed by C61. Reference to this table will aid the reader in following the discussions below.

In the initialization phase, three routines are involved, Z0, C21, and D4. Z0 remembers the focus object and its designation as a member of the set of objects defined by the concept. C21 creates routine Z7 which establishes the initial conditions within the "subject." D4 determines whether the subject is to continue onto Phase 2.

The processing of the focus of object information by Z0 is accomplished by four P level routines, P21, P61, C31, and P62. Because the subject and the experimenter both

manipulate objects, it was necessary to design the program so that information received from the external environment could be separated into its external representation and the subject's internal representation of the same information. Thus, P21 creates a copy of the focus object with its dimension values in dominant dimension order and also creates a dummy description list indicating that the focus object is a member of the set defined by the unknown classification rule. If this separation is not made, descriptive information created by the subject becomes attached to the object in the external world, an undesirable situation. P21 leaves the name of the subject's representation of the focus object in M1,N and its set membership on a DDL in M1,D. The memory process, P61, remembers the focus object under an attribute of the problem list and C31 places the name of the focus object at the top of the memory entry point list. P62 then remembers the set membership of the focus object under an attribute of the focus object. At the present time routine D4 is a dummy routine as the decisions subjects make at this point have not been ascertained, but it has been inserted to keep the phase list structure consistent.

The procedure Z7 is created by the contexting routine C21; a detailed discussion of contexters is given later in the present section. In Z7 the first routine is P191, and the inputs of this routine are the name of the memory entry point and K99 which is an input constant specifying how many of the possible dimensions are to be worked with throughout this attempt at attaining the concept. The function of P191 is to create a working hypothesis from the focus object remembered by Z0. The working hypothesis is created by selecting the first K99 of the m dimensions on the focus object and placing them on a separate list, the rationale being that some subjects deliberately work with less than the total number of dimensions and other subjects do so inadvertently. The working hypothesis is a list in its own right; a description of how it was created from the focus object, namely the dimensions which have been removed in order to obtain the working hypothesis, is made into a dummy description list. P191 leaves the name of the working hypothesis and the name of the description in working memory M1. Following P191 is a memory process routine, P63. P63 remembers the name of the working hypothesis under an attribute of the problem list devoted to the working hypothesis. At this point, the program needs to remember the description of the working hypothesis rather than something about the problem, hence P63 is followed by a memory

process, C31, which will put the name of the working hypothesis into the memory entry point list, pushing down the list and saving the name of the problem. C31 is then followed by a memory process, P64, which remembers how the working hypothesis was formed from the focus object. Thus, the problem was described by the working hypothesis and the working hypothesis was described by how it was created. The rationale underlying this type of description is that other routines and contexting operations can utilize the information to determine what has happened and then modify or create routines to change the behavior, if necessary. It should be noted that the contents of working memory remain unchanged during P63, C31, and P64 from the time P191 creates the working hypothesis and the dummy description list for describing it; however, the memory entry point changes from the focus object to the working hypothesis to make it available to the next routine or the next phase.

The second phase is the object-selection phase which consists of three routines, Z1, Z2, and C32. Routine Z1 creates a search criterion from the working hypotheses; Z2 locates an object matching the search criterion; and C32 creates the routine D0 which determines whether the subject can proceed to the next phase. The procedure Z1 consists of four routines: P131, P141, P151, and P64. The inputs to P131 consist of M10, the memory entry point, and a constant, K98, which specifies the number of dimensions to vary. In the normal conservative-focusing strategy K98 would be one; however, it can be set to any number up to the number of dimensions on the working hypotheses. Note that $K98 \leq K99$. M10 contains the name of the working hypothesis at the top of its list and it is from this hypothesis that P131 will select the dimensions to vary. Because P21 had arranged the working hypothesis in dominant dimension value order, P131 merely needs to select the first K98 of these dimensions, thus implementing the dominance feature in the program; i.e., the first dimension value on the list is the most dominant and the last is the least dominant. P131 creates a copy of the working hypothesis and puts the name of this copy into working memory on M1, N. It also creates a DDL on which it lists the names of the K98 dimensions which are to be varied at this point. The DDL is only a partially completed list which eventually will become a from-to list; i.e., it will name the dimensions, tell what their values were originally, and tell to what value they were changed. However, P131 only places the names of the dimensions

to be worked with on the DDL. The output from P131 is left in M1, N and M1, D. Routine P141 takes the information from M1 and extracts the dummy description list containing the names of the dimensions to be varied. It then enters the dimension list, M13, which has been stored on the problem list L100 under the name of the external environment, ascertains a given dimension, determines what values are available other than that of the focus object, and, if there are more than two values, selects a dimension value on the basis of its dominance value. The value found is then added on the DDL under the "changed to" attribute and the value on the focus object is stored under the "changed from" value. Upon the completion of P141, the name of a copy of the working hypothesis is in M1, N, and the name of the dummy description list is in M1, D. P151 is the routine which varies the dimension values to create a search criterion from the working hypothesis. It receives the working hypothesis and the DDL through M1 and will use the from-to list to change the dimension values on the copy of the working hypothesis to their new values, thus accomplishing the dimension variation. If an initial input flag dealing with awareness indicates that the subject is aware, the DDL is placed in M1, D with the name of the search criterion which has been created in M1, N. It should be noted that the working hypothesis had not been disturbed because the changing of dimension values has occurred on a copy of a working hypothesis known as the search criterion which is used to locate objects on the board. If the awareness flag indicates the subject is not aware, it is then assumed that he has inadvertently varied more than one dimension, even though he believes he is only varying a single dimension. It was very common in the protocols for subjects to choose objects which varied in more than one dimension even though they believed they were searching for an object which varied in only one dimension. Possibly this is a perception problem; however, because perception has been eliminated, awareness is handled in this somewhat mechanical fashion. If the subject is unaware, routine P151 will then delete from the dummy description list all of the dimensions and their values other than the first one. From this point on the subject's description of what he has done indicates that only one dimension has been manipulated even though multiple dimensions were actually varied. P151 is followed by a memory routine, C32, which will place the name of the search criterion on the memory entry point list with a push down of previous information. C32 is followed by a

memory process routine, P64, which describes the search criterion with the DDL from the working memory.

The remainder of the simulation program operates in a fashion quite similar to what has been described above; as information is created or received from the external world, it is initially left in the working memory, and the routines which process this information create a description and leave it in the working memory. Depending upon the information and how it is used, it is either left in working memory for subsequent routines to pick up and use as information or, usually at the end of a series of routines, attached to a previous unit of information through a memory process and the memory entry point list.

As Phase 3 involves the use of an interesting contexter, the operation of this particular routine within its situation will be described in detail. At the time Phase 3 is entered, routine Z2 has located an object meeting the search criterion and has stored it in short-term memory under an "object found" attribute of the search criterion. The name of the object found has been stored in the memory entry point list, and the object has been described by the dimensions of the working hypothesis that were varied in order to find it.¹ The first routine in Phase 3 is procedure Z3 which consists of processes P71, E93, and P62. Routine P71 is a memory output process which transfers the subject's name for an object to an output buffer called E1 from which the experimenter will receive the information. Routine E93 is an experimenter routine which acquires from the E1 buffer the subject's name for the object chosen and compares the dimension values of the object with those of the concept to ascertain whether or not the object contains the dimension values of the underlying classification rule. E93 creates a dummy description list similar to those previously used which will contain a designation attribute and a value of *yes* or *no* for the set membership of the object choice. E93 also returns the subject's name for the object to M1,N so that the subject may associate the designation with the object he has presented to the experimenter. Because the memory entry point contains the name of the object found, a memory process, namely P62, can be used to attach the experimenter's designation in working memory to the object choice in short-term memory.

¹The reader should refer to Figure 3 to trace the levels of short-term memory involved.

The experimenter's designation of the object is information from the external world, hence it is mandatory that C61 insert a contexting program at this point. Again some corners have been cut in that the appropriate contexting routine for this situation, C38, has been preprogrammed whereas in a more sophisticated program the C61 contexter would analyze the total situation and create the contexting routine C38. However, this level of sophistication in the program development has not been reached. The contexting routine C38 will create the routine Z4 which is the reaction of the subject to the experimenter's designation of the object. Initially it creates a description of the characteristics of the required Z4 routines. C38 then receives from the input list its own location in the phase list that the interpreter is currently executing. Using this information, C38 ascertains whether or not the next symbol on the phase list has a description matching that of the routine Z4 which it wishes to execute. If the routine following the C38 contexter is a Z4 routine, it will be removed from the strategy and its symbol replaced by the symbol representing the new Z4 which will be created. If no Z4 symbol follows C38, as in the first time through the phase, a symbol to hold a position for the Z4 routine is inserted on the phase list. Notice that at this point the phase list merely contains a symbol whose description indicates what the behavior should try to accomplish; however there is no executable subroutine associated with the particular symbol. The phase list is also described through the use of the DDL technique to indicate that a routine has been either inserted or replaced on the phase list. The long-term goal is for contexters to utilize this change description to ascertain what has occurred during the execution of the program. The contexter C38 uses the memory entry point to obtain the name of the object and through a descriptive attribute ascertain whether it is a *yes* object or a *no* object. If the object was designated a *no*, C38 determines whether or not the subject was aware; if the subject was aware, C38 checks to see whether the number of dimensions varied was equal to one or not. If it was greater than one, no information has been gained; the subject consciously varied more than one dimension and received a *no* so he does not know which of the two dimensions is the relevant one. In this situation, C38 will pop the memory entry point back to the working hypothesis so that Phase 2 can be executed again. If the subject was aware and only one dimension was varied, or if the subject was unaware, the program returns to the creation of Z4. Again, difficulties have been circumvented

by merely inserting routines that we know are necessary to accomplish the reaction to object designation. A routine called P502 is inserted which will recall the object designation. It is followed by C41 which pops up the memory entry point from the object found to the working hypothesis. Popping the memory entry point is necessary because Z4 must have both the object designation and the working hypothesis to react to the object designation. The next routine inserted is P96 which uses the information about the working hypothesis and object designation to flag the dimension values involved as relevant or irrelevant, depending upon the designation. Following P96 is P91 which looks at all possible values of a dimension and checks whether they are marked relevant or irrelevant. If all values are marked, the dimension itself is then marked as relevant or irrelevant. However, if any dimension value is still untested, P91 will not attempt to mark the total dimension. We have found that many subjects will not consider a dimension to be relevant or irrelevant until they have checked all *n* dimension values. If the subject is unaware of the number of dimensions actually varied, no further processes are required in Z4. However, if the subject is aware, he then also normally realizes that any dimension flagged irrelevant is no longer of concern in selecting objects and a routine called P100, which removes an irrelevant dimension from the working hypothesis, is inserted after P91. The net effect of P100 is to enable the subject to choose objects in Phase 2 which vary in two, three, or four dimensions from the focus object even though he is varying only one relevant dimension; the remainder no longer enter into any of his decisions. One can obtain what looks like rather peculiar object choice sequences; however, the subject is truly varying only one dimension. If routine P100 has been inserted, it will be followed by a memory process, P64, which remembers the description of dimensions removed from the working hypothesis so that at some later point a routine can put these dimensions back in again if necessary. The final operation performed by C38 is to put a terminal symbol on routine Z4 so that it can be properly terminated by the interpreter at execution time. At this point a rather tricky operation takes place. C38 creates the next routine to be executed and places its symbol on the strategy list; when C38 terminates, the interpreter executes this routine from the strategy list.

Phase 3 is terminated by procedure D1 which determines whether enough information is available to present a concept. D1 consists

of a single decision process routine, Q101, which uses the memory entry point to obtain the name of the working hypothesis. Each dimension of the working hypothesis is checked to determine whether it has been flagged relevant or irrelevant. If all dimensions have been flagged, sufficient information is available to present a concept. Such a test is rather stringent as it requires the subject to vary all dimensions of the working hypothesis prior to forming a concept. Experienced human subjects using the conservative focusing strategy do vary all dimensions, as they know the concept must consist of the relevant dimensions. The result of D1 is an indication to the interpreter to either continue to Phase 4 or to return to Phase 2 and vary additional dimensions. Note again that a phase terminates in a decision routine.

There is actually little variation in the routine Z4 created by the contexter C38 during the first pass through the program. However, once Phase 4 has placed previously unused dimensions on the working hypothesis, the Z4 routine can vary slightly depending upon the decision net through which it passes. C38 could be made much more extensive in the future; however, the rather rudimentary contexting operation used reflects current lack of understanding of mechanisms in reaction to an object designation. The only other contexter of any consequence in the program is C22 which creates process Z6, the reaction to a concept designation. C22 operates in much the same fashion as does C38, using factors such as the number of dimensions varied, the number of untested dimensions, and whether or not the subject is aware to create Z6.

Procedure Z6 is the subject's reaction to a concept which has been designated by the experimenter as incorrect. The procedure consists of P level routines Q41, P181, and C31. The contexter routine C22 has returned the memory entry point to the focus object so that Q41 may inspect it. Q41 uses the dimension values of the focus object to ascertain which dimensions have not been varied and creates a dummy description list containing their names. The number of untested dimensions to be used is controlled by the parameter K97, which specifies how many of the available dimensions to use. P181 uses the dummy description list created by Q41 to restore K97 dimension values to the working hypothesis. It partially undoes the work done by P191 in procedure Z7. If all dimensions have been varied and the concept was incorrect, Q41 assumes a dimension has been misflagged and P181 makes a copy of the focus object for the next working hypothesis. C31 places the name of the new working

hypothesis in the memory entry point. P64 remembers how the new working hypothesis was created. The strategy now returns to the beginning of Phase 2 and is reexecuted. It should be noted that the high level contexting performed by C61 is no longer needed as a complete strategy list is available for execution. C61 merely presents this strategy list to the interpreter for execution.

The paragraphs above presented the major features of the computer model. To describe the lower level programs which the computer program actually executes would require a sophisticated knowledge of IPL-V and is beyond the scope of the present report.

SUMMARY OF THE CHAPTER

The computer model of concept attainment embodied in Mark IV, Mod 2 consists of two parallel fractionizations of the behaviors involved in a learning strategy and a memory mechanism which facilitates those two aspects of the model. The parallel breakdown consists of the contexter routines and operational routines. The former represent the higher level cognitive processes associated with developing strategies, maintaining goal-directedness, and improvement of learning. The latter represent those aspects of the model which actually perform the tasks involved in attaining the concepts. Routines which perform tasks required by the information-processing language rather than behavior required to attain a concept have been hidden at a lower level within the program. One of the difficult tasks in the present project was to recognize that operational information processing had to be separated from those higher level cognitive behaviors associated with attaining the goal. The latter have been embodied in the computer model as a hierarchy of contexting routines. The high level contexters have been designed to translate the experimenter's verbal instructions into a skeleton strategy for behavior. The second level contexters create routines associated with initializing a problem and analyzing the behaviors in a completed problem, and the third level contexters create situationally dependent procedures in an attempt to adapt behavior to the situation. Although the hierarchy of contexting routines is rather rudimentary at the current time, the distinction between operational aspects of learning and the contexting aspects of learning is a crucial one not previously made.

Because information, both acquired and internally created, plays such a crucial role in the concept-attainment process, it was necessary to create a model of memory which would enable the computer program to both remember and recall this information. The memory model created consisted of three levels: a working memory, which is an internal buffer-type memory; a short-term memory in which all of the information relative to attaining a given concept has been stored; and a long-term memory in which learning strategy and certain crucial pieces of information relating to them are stored for use in solving subsequent problems of the same or similar types. In the present model, the working memory serves primarily as a holding or communication device for information which is to be passed from one behavior to another within a section of the computer model. The short-term memory has a circular structure so that any given point in the memory structure looks as if it were the beginning of an information storage tree. Several generalized memory processes have also been programmed to permit the model to remember and recall information within this circular memory structure. Although the model has not solved the problem of how a human enters memory under a given set of circumstances, it does include a memory entry point scheme for keeping track of the subject's location within memory, once memory has been entered. The majority of work on the model has been devoted to the working memory and short-term memory; the long-term memory has not been modeled. The memory model developed, although somewhat rudimentary at the current time, has been designed with considerable expansion capabilities which provide internal flexibility without sacrificing much of the capability already acquired.

The Mark IV, Mod 2 version of the concept-attainment model can reproduce a wide range of the behavior observed in the think-aloud protocols collected from human subjects. The range of behavior is accomplished with a relatively small number of computer routines, some of which, such as the memory processes, are completely general; others, such as the contexters, are very specific. Unfortunately much of the variability is controlled by the three input constants and the awareness flag; however, even this is encouraging in that so much variability can be controlled by so few parameters. The long-term goal is to eliminate such parameters and utilize only generalized routines to accomplish what Newell [1962] has called a "solution by understanding."

III POST MARK IV, MOD 2

The completion of the Mark IV, Mod 2 version of the computer model was accompanied by a sense of frustration. The structure of the contexters was not consistent with the basic format followed in the rest of the program, hence the computer program needed a considerable amount of work. In addition, numerous ways of cleaning up the operational programs were apparent. A period of time was devoted to studying the program and sketching how adjustments could be made, but the changes were never programmed as they were not deemed worth the effort. It was felt that any major programming effort should be made only to implement a better conceptualization, not to improve the mechanics of an old one. Prior to Mark IV, Mod 2 the importance of the higher level cognitive processes within the computer model was not understood, but the development of the contexters emphasized how crucial these processes are to a computer model. The importance of these processes and their implementation in the contexters meant that further exploration was needed in areas such as how persons select behaviors from their repertoire of behaviors, how they maintain goal-directedness during a problem, and how they structure their attempts at problem solving. Classical psychology does not appear to be interested in these problems at a level which would contribute to a computer model of concept attainment. Thus, we had reached "state of the art" limitations with regard to psychology. The final frustration was that the computer programmer who had written all of the IPL-V code and made major contributions to the total project completed his degree program and departed. It would take at least 18 months to bring another programmer up to his level of sophistication with regard to the problem and its programming techniques. The net result of these frustrations was a considerable let-down in enthusiasm for the project. Our data were about exhausted; the programming capability was diminished; and the "state of the art" in psychology seemed to have been

reached. After some thought it was decided that the last area seemed more worthwhile to attack. Perhaps some psychological experiments could be devised that would yield data upon which a better conceptualization of the contexters could be based. Therefore, attention was directed toward preparing some experiments involving human subjects. The initial experiment was called the planning experiment.

PLANNING EXPERIMENT

One issue raised by the development of a computer model of the concept-attainment task was that of the processes used by a subject to create plans or strategies, and the role of instructions in forming such plans. What was needed was some way of externalizing the development of a plan, and the processes involved therein, as the subject proceeded to solve the task presented him. One way of accomplishing this was thought to be to vary the instructions given to the subject, specifically with regard to the formation of plans, and to note the resulting effects on the subject's performance. It was hoped that by providing subjects with differential amounts of clues as to the formation of a plan, some light might be shed on the processes involved. It was hypothesized that the more complete the instructions concerning the formation of a plan, the better the performance of the subject.

Methodology

Subjects. The subjects were 20 female University students whose names were obtained from the Student Employment Service.

Materials. The subjects were presented with a board on which were mounted 36 photographs of cartoon animals which were basically similar but varied along four dimensions. These dimensions and their values were body color (yellow, blue, or brown), neck length

(short or long), ear length (short or long), and type of tail (straight, bent, or curly). The 36 animals were randomly arranged on the board in rows of six.

Design. A completely randomized design employing four treatment levels was employed. The treatments consisted of four levels of instruction ranging from the introductory overall instructions all subjects were given to a brief outline of the conservative focusing strategy. The treatments were defined as follows:

INTRODUCTORY INSTRUCTIONS, ALL SUBJECTS

What you will be doing in this experiment is trying to find out what animals go together to form a category that I consider correct, in other words, to find out what the rule is that describes the category of objects that I have in mind. The rule is based on some *systematic characteristics* of the *animal itself*.

To start off, I will point out one object that *does* belong to the category I have in mind. From then on you will attempt to discover the rule by choosing one animal at a time, and I will tell you whether it belongs to the category or not. Whenever you think you know what the correct rule is, tell me and I will tell you whether you are correct or not.

It is important that you think out loud during this experiment so I have a way of knowing what you are trying to do.

TREATMENTS ADMINISTERED AFTER FIRST PROBLEM

Treatment I: No further instruction (control group).

Treatment II: "Before you begin the next problem, it might be of help to you if you take a minute to think about how you will go about solving the next problem."

Treatment III: The instructions of Treatment II plus: "... so that you will get the most information from each choice that you make."

Treatment IV: The instructions of Treatment III plus: "One efficient way of doing this is to vary only one characteristic of the animal at a time from the one that I first give you."

Treatments II-IV: "Let me know when you are ready to begin again."

Procedure. All subjects were run individually. The subject was seated before the board and given the introductory instructions, followed by the first problem. In brief, the experimenter designated one card that did belong to the concept and from then on the subject indicated which of the cards he wanted membership information about. Although only the data of those subjects who did not appear to be planning in the first problem (according to criteria that will be detailed below) was to be analyzed, all subjects completed the experiment. After finishing the first problem, subjects were given additional instructions depending on the group they had been assigned to (or in the case of the control group—no further instructions), and continued to solve five more problems. All subjects received the problems in the same order. See Table 4 for the list of concepts and the focus card given in each problem. A tape recording was made of all the experimental sessions, and the experimenter kept a written record of the number of choices the subject made as well as the objects chosen.

A four-item rating scale was devised to assess the extent of planning displayed in each problem solution. Each item was rated on a five-point scale with a rating of "1" reflecting the least planning and a score of "5," the most. All subjects who received an average score of moderate planning (3.0 or above) on the four items were considered as having planned on the first problem and were dropped from further analysis. See Table 5 for the rating scale for extent of planning and Table 6 for the scores obtained on the first problem. Of the 20 subjects tested, five were considered to have planned, four from the Treatment I (control) and one from Treatment II.

Those 15 subjects who received an average planning score below "moderate" for the first problem had their remaining five problems rated on the same planning scale by the experimenter. In addition, each of the 15 subjects was rated on the development of planning across all six problems by way of a four-part scale (Table 7). The ratings were made by the experimenter from typed scripts of the tape recordings of the experimental sessions.

Results

The dependent variable was the subjects' scores on the instruments dealing with extent of planning. To test for differences between experimental groups, each individual's planning score was found for each problem separately by summing his score on each of the four scales,

Table 4. Definitions of Concepts Used in Experiment

Problem	Concept	Focus Card
I	Short neck, bent tail	Blue, short neck, short ears, bent tail
II	Yellow, long neck, curly tail	Yellow, long neck, short ears, curly tail
III	Brown, short ears	Brown, long neck, short ears, bent tail
IV	Brown, short neck, straight tail	Brown, short neck, short ears, straight tail
V	Short ears	Blue, short neck, short ears, curly tail
VI	Curly tail	Brown, short neck, short ears, curly tail

Table 5. Rating Scale Used to Measure Extent of Planning

<i>1. Extent and consistency of planning</i>				
1	2	3	4	5
Little planning, low consistency		Moderate planning, moderate consistency		Much planning, high consistency
<i>2. Extent and consistency of "Z's"</i>				
1	2	3	4	5
Few Z's used, inconsistent and/or incorrect		Moderate use of Z's, moderately consistent		Many Z's correctly used, consistent
<i>3. Sufficiency-loops, repetitions, and attention to irrelevant characteristics</i>				
1	2	3	4	5
Highly inefficient		Moderately efficient		Highly efficient
<i>4. Number of object choices</i>				
1	2	3	4	5
Many more than necessary (over 18)		About average (10-13)		About as few as possible (1-6)

Table 6. Average Extent of Planning Scores on Problem I Using 5-Point Scale with 3 Being "Moderate Planning"

	Number of Ss	Average Planning Score
Excluded from analysis	1	5
	2	4
	2	3
	6	2
	3	1.5
	6	1
N = 20		

Table 7. Rating Scale to Assess Development of Planning Over Problems

1. Development of planning over time				
1	2	3	4	5
Little improvement, slow to improve		Moderate improvement, moderate speed		Great improvement or planned from the beginning
2. Development of use of Z's over time				
1	2	3	4	5
Little improvement, slow to improve		Moderate improvement, moderate speed		Great improvement or used Z's from the beginning
3. Elimination of inefficiency over time				
1	2	3	4	5
Little improvement, slow to improve		Moderate improvement, moderate speed		Great improvement or efficient from the beginning
4. Reduction in the number of object choices over time				
1	2	3	4	5
Little reduction in number, or slow		Some reduction, moderate speed		Much reduction or as few as possible from the beginning

Table 8. Comparison of the Average Summed Planning Scores by Treatment for Each Problem Separately

Treatment	1	2	3	4	5	6
II	6.5	9.0	14.5	15.0	18.5	16.5
III	6.4	11.6	13.8	15.6	14.8	17.2
IV	6	9	13.2	14.4	15.6	16.8

Table 9. Comparison of the Average Extent and Development of Planning for Each Scale Separately

Treatment	Extent				Development			
	1 Planning	2 No. of Z's Used	3 Efficiency	4 No. of Choices ^a	1 Planning	2 No. of Z's Used	3 Efficiency	4 No. of Choices
II	4	3.25	3	3.25	3.25	3.25	3.5	4
III	3.8	3.4	3.2	3.4	3.2	3.4	3.6	3.8
IV	3.2	3.2	3	3.6	3	2.6	3	4

^aReduction in the number of choices.

and the group mean computed from summed individual scores for each problem. Table 8 compares the average summed extent of planning score for each problem for the three groups. No data are presented for the control group since only one subject did not demonstrate planning according to our criterion. As can be seen, the major trend is one of solving the problems with progressively more planning, but there are no systematic differences between experimental treatments. Upon inspection of the data, a statistical test did not appear warranted.

It was also possible to compare the ratings of extent and development of planning across the six problems. In Table 9 the average group rating on each of the scales separately is presented for the three treatment groups (Treatment I is again omitted). Once again no consistent differences between groups in planning over the six problems were found and no statistical test conducted.

Discussion

From inspection of the data, it seemed apparent that the major effect obtained was one of learning to solve the problem more effectively with time, and that treatment differences did not play any systematic part. While a more methodologically tight study would require a larger sample size, and some check on the reliability of the ratings made, the results of this pilot study did not indicate that further effort in this direction would be profitable.

The lack of results from this experiment which was highly structured toward obtaining treatment differences caused some concern about the earlier decision to pursue this route. It appeared that commitment to a long-term series of such experiments would result in a very low ratio of information yield to man-hours invested. Some additional minor inquiries were conducted informally with results similar to those just observed. It was also realized that a long-term commitment to such experiments would require a complete professional reorientation of the principal investigator from computers to learning theory. Thus, it was concluded that the focus of the project should return to the computer model.

A LINGUISTIC APPROACH TO A COMPUTER MODEL OF CONCEPT ATTAINMENT

In one of the earlier progress reports written about this project [Baker, 1965] it was

stated "It is interesting to note that the names of the subroutines almost form a verbal description of the concept attainment strategy, a possibility which offers some interesting possibilities for a string language notation." The idea was premature in our thinking and was not developed at that time, but the flow chart books contained numerous marginal notes relating to a verbal representation of the concept attainment process. The viability of this idea was further enhanced by the development of the C11 contexter routine within Mark IV, Mod 2. The function of this routine was to receive coded representations of the experimenter's verbal instructions and to translate them into a skeleton strategy or plan for attaining the concept. The nature of this subroutine and the realization that the contexter routines represented higher level cognitive processes led to the notion of taking a linguistic approach to the computer model.

Not having a concise idea of what such an approach would entail, we decided to reanalyze the protocols and the computer program from a linguistic frame of reference. The study of the protocols revealed that the subjects used a very limited vocabulary to describe their own information processing. The flow charts of Mark IV, Mod 2 also indicated a similar limited vocabulary. The presence of a limited vocabulary and an intuitive feeling that humans represent information internally as verbal symbols suggested that the scope of the problem was within reason.

The Mark IV, Mod 2 version of the computer model was coded in IPL-V and nearly each line of the program was annotated to explain what the instructions or series of instructions was trying to accomplish. Thus, a verbal description of the total program, line by line, existed. The annotations made by a programmer tend to be more concerned with the mechanics than with the conceptual basis of the program. Therefore, a narrative which would verbally describe the existing computer model was written from the program and the flow charts. The narrative is included as Appendix C. Inspection of the narrative revealed that the descriptions of the processes are mainly in first person and the sentences are imperative in form. A close look at sentences—"Remember the focus object," "Find an object on the board with a long neck"—showed that they involve a single verb accompanied by its object and various modifiers. Basically, the verb represents an operation to be executed and the rest of the sentence represents the context within which the operation is to be performed. Such a division of functions already existed in our computer model and it also seemed to be the

natural one for a linguistic approach; i.e., the operational level represents a verb and the contexter program provides the objects and modifiers of the verb. Such a linguistic approach would be extremely powerful in that one need only implement a set of verbs to perform certain kinds of operations associated with the learning task. The remainder of the model would perform a syntactic analysis of verbal descriptions of the behavior desired.

An examination of this approach at the lowest level within the computer program will show how feasible it seems. For example, P131 can be described by the sentence "Select dimension of the working hypothesis." Here the verb is select, the object is a dimension which is a property of the working hypothesis. Thus, one would write a subroutine for the verb select and the contexter would provide it with the inputs of what is to be selected and where it can be found. The mechanics of Mark IV, Mod 2 would be of help here in that the memory entry print (MEP) would contain the name of the working hypothesis and the contexter could specify that it was a dimension of the working hypothesis which is to be selected. In this example the sentence would be received from a higher level routine by the low level contexter. This contexter then would find the verb, acquire the corresponding subroutine, and use the rest of the sentence to establish the entering arguments for the verb as symbols on the input list of the subroutine. Once the process had been completed, the verb routine would be executed and the output placed in working memory.

The key elements of such a scheme already exist in Mk IV Mod 2. The interpretive scheme described by Baker and Martin [1965] enables one to implement the verbs as subroutines with their inputs and outputs being situationally dependent. In addition, due to the design of the memory structure, the internal communication within the system is by and large automatic regardless of the sequencing of the subroutines. The general form of a verb under this scheme could be as follows:

R211	9-1		Pseudocode
	R210	0	verb
9-0	0		
	A1		verb modifiers
	V1		
	A2		object
	V2		
	A3		object modifiers
	V3	0	

From the above it would appear that the linguistic approach is feasible at the exe-

cutable program level at least. For such a scheme to be practical from an implementation point of view, the vocabulary would have to be quite limited and the meanings of the verbs, objects, modifiers be very specific; hence, the next step was to attempt to write the narrative descriptions of the program in a restricted vocabulary, yet retain the essentials of the system. Only the operational portion of the program was rewritten in this fashion due to our greater familiarity with this aspect of the program. The condensed form of narrative is presented below.

P190 Create working hypothesis from copy of focus object

C190 { Enter memory [(MEP) → M1, N = CFO]
Use CFO as WH

P190 Retain K99 dimension value of WH

P131 Select dimension to vary

C131 { Enter memory [(MEP) → M1, N = WH]
Create description
Hold description

P131 { Collect DV of WH not having "relevancy" attribute
Retain K98 DV of collection
Make DV list the value of "from" attribute of description
Collect value of "dimension" attribute of elements of DV list
Make dimension list the value of "dimension" attribute of description

P141 Find new dimension value

C141 { Enter memory [(MEP) → M1, N = WH]
Obtain value of "dimension" attribute of WH

P141 { Save value
Choose DV
Is DV an element of WH?
Yes—choose again
No—add "from" value to "from" list; add "to" value to "to" list)
Repeat ()
Make "from" list the value of the "from" attribute of the description
Make "to" list the value of the "to" attribute of the description

P151 Replace "from" DV on WH by "to" DV

C151 { Obtain value list of "from" attribute of description
Obtain value list of "to" attribute of description

P151 (cont.)

P151 { (Save "from" DV and save "to"
DV
Replace "from" DV on WH by "to"
DV)
Repeat ()

P50 Search board

C50 Enter memory [(MEP) → M1, N = WH]
Search board for object matching
WH
P50 { Hold name of object
Obtain value of "how varied"
attribute of WH
Hold value as description

Q50 Verify object chosen

C51 { Enter memory [(MEP) → M1, N =
object]
Obtain value of "to" attribute of
object
Enter memory [(MEP) → M1, N =
Focus Object]
Collect not common element of
FO and object
Hold name of collection
Collect not common elements of
previous collection and "to" list
Hold collection
Q50 { (Is relevancy an attribute of DV,
alt: Is DV marked?
Yes—O.K.
No—Set H5 negative)
Repeat ()

P70 Present object to experimenter

C70 { Enter memory [(MEP) → M1, N =
object]
P70 { Transmit name of object to ex-
perimenter

P96 Mark DV

C96 { Obtain value of "designation"
attribute of object
Obtain value of "from" attribute
of object
Retain K96 DV of "from" list
P96 { (Save value of "from" list
Use experimenter's designation
of object to determine the "rele-
vancy" attribute of DV)
Repeat ()

P91 Make conclusions on dimensions

C91 { Enter memory [(MEP) → M1, N =
object]
Obtain value of "designation"
attribute of object
Obtain value of "dimension"
attribute of object
P91 { (Save dimension
Are all but one DV marked?
Yes—Set value of the dimension's
relevancy attribute and concept
name
No—Exit)
Repeat ()

P101 Remove irrelevant DV from WH

C101 { Enter memory [(MEP) → M1, N = WH]
Obtain value of "how varied" at-
tribute of WH
Obtain value of "dimension" at-
tribute of description
Obtain value of "from" attribute
of description
Create description
Hold description
P101 { (Save DV and save dimension
Is irrelevant the value of the rele-
vancy attribute of the dimension?
Alt: Is dimension marked irrele-
vant?
Yes—remove DV from WH
No—go to repeat
Mark DV the value of the "deleted"
attribute of the description)
Repeat ()

P171 Replace "to" value of WH by "from" value

C171 { Enter memory [(MEP) → M1, N = WH]
Create description of hold ?
Obtain value of "to" attribute of
WH
Obtain value of "from" attribute
of WH

P171 { (Replace "to" value of WH by
"from" value
Make "to" the value of the "from"
attribute of description
Make "from" the value of the "to"
attribute of description)
Repeat ()

Q101 Can concept be presented?

C101 Enter memory [(MEP) → M1, N = WH]

Q101 (cont.)

Q101 { (Obtain value of "relevancy" attribute of DV
Was value obtained?)
No—Exit, set H5 negative
Yes—Repeat ()

P121 Form concept

C121 { Enter memory [(MEP) → M1,N = WH]
Create description and concept name
Hold description
Make WH the value of the "how formed" attribute of the description
P121 { Collect elements of WH having relevancy attribute to form concept

Q41 Find untested dimensions

C41 { Entry memory [(MEP) → M1,N = Copy of Focus Object]
Q41 { Collect dimension values of CFO not having relevancy attribute
Retain K97 elements of collection
Create description
Make collection the value of the "to" attribute of the description
Collect the values of the next upper attribute of elements of collection
Make collection the value of the "dimension" attribute of description

Q43 Find dimensions marked irrelevant

Same as Z41 above except that it collects elements having irrelevant as the value of relevancy attribute

P181 Add dimension to WH

C181 { Enter memory [(MEP) → M1,N = CFO]
Obtain value of hypothesis attribute of CFO
Obtain value of "to" attribute of description
P181 { (Add element of "to" list to WH)
Repeat ()

The initial impression of this condensed narrative is that it is primarily concerned with internal data processing. A large proportion of the statements deal with the mechanics of organizing information from memory, holding the information for subsequent use, and making

decisions based upon characteristics of this information. Again, this is in keeping with the earlier observation that most of the information processed is created internally during the problem solutions. Part of the problem rests with the design of the P and Q level routines which, in general, encompass too large a segment of behavior. The result is that the execution of behavior requires too many interrelated steps. The condensed narrative does provide a reasonable fractionation of these behaviors into smaller units. In many cases these smaller units are actually subroutines in the present program, although they were not structured with a linguistic approach in mind. From this condensed version, it was possible to compile a list of the verbs employed and their objects, modifiers, etc. As initially compiled, the list of verbs contained considerable redundancy and overlap of function, but after some effort the following verbs were defined:

Verb	Definition	Equivalent IPL-V Primitive
Remember	M1,N or M1,D becomes value of attribute__ of the contents of the Memory Entry Point (MEP)	J11
Recall	The value of attribute__ of the contents of the MEP is placed in working memory	J10
Enter	The contents of the memory entry point is placed in working memory	
Use	An undescribed copy of a list is created and given a name which is left in working memory	J73
Collect	(a) Elements of list X having (not having) Y as value of attribute Z are placed on a list whose name is left in working memory (b) Elements not common to lists A and B are placed on a list whose name is left in working memory (c) The values of attribute__ of the symbols on list__ are placed on a list whose name is left in working memory	
Remove	Delete symbol __ from list __	J69
Create	Make a symbol for a name or a dummy description list	J90

Verb	Definition	Equivalent IPL-V Primitive
Describe	Make Y the value of attribute Z of X	J12
Retain	Keep only the first K elements of list X	J75
Obtain	Get the value of attribute Z of list X	
Select	Randomly pick a dimension or dimension value	J16
Add	Insert symbol ___ on list ___	J65
Repeat	A section of the program is reexecuted	
Replace	Element ___ is removed and element ___ is put in its place	J67
Search	The external environment is inspected for an object matching the search criterion. Name of the object found is left in M1,N	
Transmit	Move symbol A to location B	

Property	Decision Routine	Equivalent IPL-V
Belong	Is ___ an element of list ___?	J62
Equal	Is symbol ___ the same symbol as ___?	J2
Found	Was ___ obtained	
Characteristic	Is value of attribute ___ of list ___ equal to ___?	

The utter dependence of the computer model upon the underlying structure of the language in which it has been programmed is quite clear in the list of verbs. It should be noted that the working memory is our analogue to the HO communication cell of IPL-V, but IPL-V has no analogue to the memory entry point. Nearly all verbs have corresponding IPL-V primitives which perform nearly the same function. This illustrates the fact that when programming in IPL-V one has difficulty developing higher level programs which do not reflect its characteristics. The natural question is why not write directly in IPL-V and forget Z's, P's, Q's, and the contexters. There are three major reasons why one should not do so. First, there is considerable hope of establishing a circular

memory structure in which information can be stored and retrieved by the computer model rather than by the computer programmer. Second, the interpretive scheme, in conjunction with the contexters, allows the computer model to create program sequences and modify them. Third, most if not all of the messy housekeeping details of the IPL-V language are buried deep within the subroutines and are never a consideration within the computer model. If such details are not hidden, the major portion of the model becomes enmeshed with mechanics of housekeeping. Thus, what has been developed in Mark IV, Mod 2 is essentially a higher level set of IPL-V primitives which permits symbol manipulation without concern for the mechanics of memory management or housekeeping details inherent in the language. If one does not get the model above these details, it becomes impossible to develop contexters and other routines which can create program sequences which are arbitrarily ordered. Without such a higher level symbol manipulating capability, the linguistic approach would be exceedingly difficult to implement. In addition, it would seem, to this author at least, that the current approach could lead to a computer modeling language in which a verbal description is given of what is to be done and the underlying IPL-V is compiled. The existence of many verbs in subroutine form in Mark IV, Mod 2 and the Baker-Martin interpretative scheme suggest that this aspect of a linguistic approach is possible.

Although the operational verbs can be implemented, they are only a small portion of the linguistic approach. The major portion of such a model must deal not with the translation of the experimenter's verbal instructions, but with their elaboration into numerous subbehaviors. This elaboration procedure is performed covertly by the subject and is exceedingly difficult to study. To illustrate the nature of those elaborations, let us examine the following sentence in the experimenter's instructions: "You are to choose an object from the board." The sentence tells the subject what to do but does not specify how. The subject must elaborate this sentence into a complex sequence of behaviors. The sequence is roughly as follows: A basis for choosing an object must be established; a procedure for comparing this basis with the object must be developed; and certain of the resulting information must be remembered. These sentences must be elaborated even further. For example, the basis of the object choice involves combinations of dimensions and their dimension values, the number of dimensions to vary at once, and rules for deciding whether the object agrees with the criterion. All of

these internally created considerations must be organized into purposeful behavior and retained for execution. The nature of this elaboration process is not entirely clear and how one would develop the mechanics for its implementation is obscure.

The General Problem Solver [Newell, Shaw, & Simon, 1958] has solved the elaboration problem within a special framework, and perhaps the approach could be adapted to the present problem. The GPS program determines the discrepancy between the present state and the desired state. It then attempts to reduce this discrepancy into smaller units, each of which are handled in the same way. Eventually resolution of a small discrepancy permits higher level discrepancies to be handled. In the present situation the experimenter's instructions could be the present state and the verb, with its requirements, be the desired state. The elaborations could consist of trying to meet the verb's requirements. Let us use the verb *choose* to illustrate how this might be done. A prototype of the verb *choose* could be stored in long-term memory and its description would contain a specification of the kinds of information necessary to execute the verb—the object of *choose*, the basis for the choice, the environment from which the choice is to be made, and that the chosen object is to be remembered. The contexter would then use the experimenter's message to meet as many of the verb's requirements as possible. It could specify what is to be chosen and the environment it is chosen from. The rules of the programming system dictate that the object chosen be given an internal name which is left in M1, N of the working memory. Thus, the only discrepancy at this point is the basis for choosing the object. Several possibilities exist. First, one could look for the word *basis* in long-term memory and attempt to fulfill its requirements in a similar fashion. Second, one could invoke the conservative focusing strategy which formulates that object to be chosen by varying one dimension per object choice. The latter is easier to implement, but the former is probably the proper approach although *basis* would need to be properly defined as it could have

several different meanings depending upon the context. If such an elaboration process were successful, the end product would be the symbol for the verb *choose* with its description list containing the symbols necessary to execute the verb in its present context. Thus, at the IPL-V code level *choose* would mean to compare symbols on one list with symbols on another list. If they match, the object is chosen; if not, a new list is obtained and the process repeated.

Another problem intrinsic to a linguistic approach is that of automating when information is to be remembered and when it is to be recalled. At the present time simulation programs remember everything or the computer programmer has built in his intelligence to define when memory processes are to be performed. The heuristics underlying automatic memory processes are completely beyond the author of the present report and perhaps beyond the state of knowledge at the present time.

Because of the difficulties associated with elaboration and self-initiating memory processes, it does not appear feasible at the present time to attempt to develop a full-blown linguistic model. A much more feasible approach would seem to be one in which the condensed narrative presented above is structured in linguistic form so that every statement involves a verb. The complete concept-attainment task could then be written out as in the condensed narrative. Low level contexters could be written which analyze these short sentences and establish the requirements of each verb. Then each verb would be executed. The higher level contexters would be used to effect the elaboration from the experimenter's instruction to the *known* lower level verbal statements contained in the condensed narrative. Such an approach could enable one to develop low level contexters to handle the verbs and high level contexters which would provide some insight into the elaboration process. Attacking the linguistic model from this angle would seem to offer the greatest possibility for a better understanding of the concept-attainment process. But because this is a final report for this project, others will have to implement these ideas.

IV SUMMARY AND CONCLUSIONS

The model has been developed to its current state through a combination of protocol analysis, computer program analysis, and hours of spirited debate. A comparison of the first concept-attainment program with the current version reveals many differences—some obvious, some subtle, but, hopefully, all in the direction of increased understanding of the concept-attainment process. As was indicated in the introduction, the concept-attainment task was chosen because it appeared to be a simple task and easy to program. There was little realization that it would lead to a hierarchy of contexting routines, a model of memory, pseudocode schemes, and many other facets of the present model. Each problem encountered and the solution devised for it merely served to expose previously hidden considerations which were more difficult and more important than the problems previously encountered. Thus, the deeper the project has delved into concept attainment, the more complex the psychological processes have become. The original estimation of the simplicity of the task has changed to respectful awe at the potential complexity of even the most rudimentary cognitive behavior. Such a new frame of reference has strongly reinforced the author's conviction that computer modeling provides a powerful tool for investigating cognitive behavior.

In the preliminary report of this project [Baker 1965b] each of the various versions of the program was explained in some detail. In this chapter the important features of the several programs are summarized. The rest of the chapter has been devoted to discussing the salient aspects of what the principal investigator feels was learned from his experiences in computer modeling of the concept-attainment process.

A SUMMARY OF THE CHARACTERISTICS OF THE VARIOUS VERSIONS OF THE MODEL

During the course of development of the various concept-attainment programs, a num-

ber of major themes developed, some of which occurred rather early in the project, others only after the investigator had considerable experience in attempting to model the concept-attainment process. The original computer program, Mark I, was based upon a rather intuitive idea as to how the author would solve the concept-attainment problem. In attempting to write an IPL-V program for the concept-attainment task, it was necessary to introduce things such as random number generators to create hypotheses and record keeping systems for determining which possible combinations of dimension values had been used. The program reflected neither a clear-cut underlying strategy nor any clear-cut understanding of the underlying mechanisms. Mark I was just an attempt to see whether a program could be written to attain a concept. In addition, an attempt was made to provide the program with a certain amount of variability in its object choice behavior through the use of various constants, length of lists, and mechanisms of this general type. At the time the first program was written, such was the basic approach underlying many of the published programs for various cognitive behaviors.

A number of lessons were learned from programming Mark I version of the program and most of these were associated with programming in the IPL-V language. Although extensive subroutining is standard practice in scientific programming, it is somewhat easier to accomplish in that setting because programmers have experience with fractionating problems and recognizing reasonable subroutines. Such was not the case for the author in IPL-V as the procedures and processes involved were relatively new and how to fractionate the problem was not readily apparent. The original program tended to be one straight-line program with little subroutining. However, from the program it was obvious that greater care is needed in subroutining in simulation programs than in scientific programs and much of the later effort of the project was devoted to a continual fractionation process in order to break down

the cognitive behavior into smaller modules. Although Mark I was not very sophisticated, it clearly demonstrated the feasibility of this type of programming to the present author and suggested that a longer term project would be feasible.

The learning strategies suggested by Bruner et al. [1956] have served as a focal idea within the concept-attainment project, and the concept of a strategy list appeared very early in the development of the system. Although such a strategy list was not used in the original version, the strategy list and a symbolic representation of procedures, processes, and information processing modules were developed in the flow charting books worked out during the summer of 1964. The only features of the strategy list idea that have changed very much over a period of time are some of the mechanical aspects, such as how many links follow a decision point. A number of different schemes were proposed to implement the actual execution of the strategy lists, and eventually an interpreter program was developed by Mr. Martin. The interpreter is an extremely sophisticated IPL-V program. The interpreter developed in late 1964 to execute the strategy list remained unchanged through Mark IV, Mod 0. When the high level contexting operations were introduced in Mark IV, Mod 1, it became necessary to make minor modifications in the interpreter to identify when a contexting routine had been entered.

One can observe in the descriptions of the various Marks and Mods of the program a rather subtle change in the nature of the routines at the Z and P level. In the early days the Z's and P's corresponded to rather large segments of the concept-attainment process, and it was necessary to continually redefine each of these symbols. Although the symbols Z1, Z2, Z3, etc. have been used since the earliest days, the routines these symbols represent have changed very radically. There have been essentially three major restructurings of the strategy lists and hence of the program itself. The first of these occurred at Mark III, Mod 1 [Baker, 1965c] after it was discovered that the several memory process routines were nearly identical except for the inputs. A major effort was then made to find communalities throughout the program and utilize the same processes in several different situations. The second major restructuring of the program occurred with Mark IV, Mod 0 when the circular memory structure was introduced. All of the memory processing routines, and a number of other routines, were redesigned to take into account the incorpora-

tion of the circular memory structure and the memory entry point in the program. The third major restructuring of the program occurred in Mark IV, Mod 2 where the contexting routines were introduced at three levels. The first level contexting routines, C10 and C61, created the strategy. Both the second level contexter, C22, and the third level contexters, C37 and C38, created Z level routines which were situationally dependent.

The fractionization process is by no means complete. It can be seen quite readily in Mark IV, Mod 2 that the working-memory processes need to be restructured again and some sub-routines developed which will handle the transfer of information to and from working memory. Such routines have been designed but have not been programmed. The P's and Q's in the current version are still too large, and the amount of information processing they do is too extensive. A further fractionization of these routines depends upon more information about human cognitive behavior than is currently either available in the psychological literature or observable in the protocols.

One of the basic tenets of the program development was that of the "backwards" approach, starting from a program for a very experienced subject and working backwards to a subject who is less experienced in solving concept-attainment problems. Through Mark III, Mod 1 the computer model was strictly that of an experienced subject. In Mark III, Mod 2 it was discovered that with relatively little effort it was possible to create nearly all the basic types of variability required by the several types of Bruner strategies and observed within the protocols by assembling the various P's and Q's into new types of Z's. Variations within the conservative-focusing strategy have been introduced principally through the means of the constants K96, K97, and K98, although in the Mark III, Mod 2 version they were handled somewhat clumsily. In the Mark IV, Mod 2 version the three constants will elicit all of the variability, other than the wholist strategy, previously observed in Mark III, Mod 2. In Mark IV, Mod 1, the awareness factor was also introduced, which was related to the psychological dependence of the dimensions in Bruner-type materials. A considerable amount of variability can be constructed through the use of the awareness factor. Its psychological origins are considerably deeper, but the parameter is a reminder to look at this type of behavior. At the current time, the within-problem variability exhibited by the computer program is quite satisfactory, but it is unfortunate that such variability results from

the "screwdriver" parameters—K96, K97, K98, and the awareness factor. The ultimate goal is to have the within-problem variability result from the "subject's" own mechanisms. Eventually, the within-problem variability will occur at the contexting level where, through misanalysis or other mechanisms, the program will create its own variability. Such a capability is presently provided by having those of us on the outside of the program code it in through the screwdriver parameters. Internal creation of variability in behavior is not a trivial problem and has been investigated by many other people. To solve it would mean accomplishing Newell's [1962] "solution by understanding"; this does not appear to be on the immediate horizon.

In retrospect, it appears that the major portion of our programming effort was devoted to memory structure. It was realized in the summer of 1964 that much of the success of the concept-attainment model would depend upon how adequately memory structures were modeled. In the original version of the program no attempt was made to model memory. Information was merely stored in IPL-V lists and data terms, and the computer programs were written to extract information from storage when necessary. The first version of the program in which any serious attempt was made at building a memory model was Mark III, Mod 0 in which the three-level breakdown of working memory, short-term memory and long-term memory was utilized. The two cell idea of the name and description within working memory was also invented. The mechanics were quite rudimentary and the idea of a dummy description list, although mentioned, was not fully developed. The Mark III version of the program also introduced the modular memory structure. The problem arose of determining when the program had reached a level at which information was available, and the nonbodied lists containing specific attributes were invoked in order to terminate the searching procedures. The Mark III version had a confused scheme for extracting information; some P's would use a memory process to acquire information whereas other P's would directly use the name of a list and obtain the information. The confusion reflects our uncertainty about the structure of memory.

In the Mark III version of the program, it became quite clear that most of the information dealt with by the program was descriptions of other information. The modular memory structure was designed to implement storage of descriptions rather than storage of specific items on lists. Although a rather highly inter-

linked memory net was inadvertently developed, it was not until a series of discussions were held with Dr. Ross Quillian at Systems Development Corporation that the possibility of completely interlinking the memory net was realized. With this concept in mind, the memory structure of the program was completely redesigned in the Mark IV, Mod 0 version where the circular memory structure and the memory entry point were introduced. Although the circular memory structure was new, the modular structure utilized in Mark III was retained as the mechanisms were well understood and seemed to be functioning fairly well. The problem of how the computer program could store and recall information under its own control is still unsolved and is one of the major unsolved problems of modeling cognitive behavior.

The subject's control of his own behavior, i.e., contexters, had its origins in the very early days of the project, all of the flow charting books carry marginal notes which record various ideas about contexting. The original contexters were conceived of as low-level programs which would establish the input list under A1 and output list under A2 for each of the P routines, but the low-level contexters were never programmed due to structural difficulties in Mark III. After development of the circular memory structure in Mark IV, Mod 0, it became obvious that representation of the total concept-attainment process was necessary, and a rudimentary model of the total contexting process from the experimenter's instructions to the actual execution of the program was made. Again the low-level contexters escaped our attention, and computer programs to set up the inputs to the various P's have not been written in that an adequate description of behavior is not available. The attribute system used in the experimenter messages to describe gross behavior and also to describe the procedures on the strategy list is a temporary device to be used until a better insight is gained. Newell's article [1962] on the internal organization of computer programs provides several examples of his attempts to resolve this problem within the General Problem Solving program. A system for describing behavior which a computer model can handle alone is an extremely difficult task and so far has eluded investigators involved in computer modeling of cognitive behavior. The final outcome from the study of the contexter was the idea for a linguistic model of the concept-attainment process. It appears to be reasonable to employ the mechanics of Mark IV, Mod 2 in conjunction with a rudimentary syntactical analysis capability to construct a

contexter hierarchy which deals solely with an English language representation of behavior.

MODELING CONSIDERATIONS

Internal vs. External Information

In the early days of the project, the concept-attainment process was thought to be primarily one of processing information received from the external world. However, after the computer program had been developed to the current point, it became apparent that the majority of the information processed does not come from the external world but is created internally by the subject. Thus, although concept attainment is an information-processing problem, the amount of external information processed minimal and consists only of the objects, the experimenter's instructions, and his designation of object choices or of concepts. It should be noted that perception problems associated with observing dimensions and their values were intentionally omitted, but this is typical of most existing computer models. As the majority of information is created internally, it is the task of psychologists to determine what internal information is created and how it is processed. For example, from a protocol it is quite easy to determine that when an object is designated as a *yes* or a *no* the subject creates information about the relevancy or irrelevancy of a particular dimension or dimension value, but on what basis he does is not clear. If one is to develop an adequate computer model, one must know what information is created, on what basis a subject created the information, what he did with it, and how much of it was retained for longer term use. Without substantial knowledge of this type it becomes difficult to develop sophisticated computer models. Unfortunately the current techniques of psychological experimentation do not seem capable of providing the requisite insight.

The Memory Model

Analysis of the concept-attainment task indicated that any significant modeling of the concept-attainment process was impossible without some model of the structure of memory and of the cognitive processes associated with remembering and recalling. The three-level structural model of memory developed for the present simulation program appears to

be a reasonable model. The idea of the working memory functioning as a temporary holding-type memory has proved to be an exceptionally useful concept as it enables information to be communicated from routine to routine without going through the rather complex mechanisms associated with short-term memory.

Conversations with Dr. Ross Quillian elicited the observation that the memory structures in the earlier editions of the concept-attainment program were very nearly memory nets. Later, the memory structure was redesigned to the present circular memory structure. The use of a list structure format for memory has seemed excessively artificial to the present author and the circular memory structure appears to provide a reasonable alternative. The significant feature of the circular memory structure is that, although the memory processes in the model can store and recall information, the memory does *not* consist of a series of predefined bins into which information is automatically placed. The memory structure is dynamic in that storage is created in the proper structure as the information is created, rather than knowing ahead of time that certain pieces of information are to be stored in given registers. The dynamic nature of the circular memory structure also gave rise to the problems of entering the memory structure and keeping track of where one is in memory. Because the order in which memory is created is situationally dependent, the memory entry point (MEP) has proven to be quite successful in performing the bookkeeping associated with the circular memory structure. The problem which is as yet unresolved is a mechanism for entering an existing memory structure, such as would be required when a second or subsequent concept-attainment problem was begun.

The memory model employed is somewhat clumsy mechanically; however, its structure does provide for the eventual inclusion of both interference and decay-type forgetting. The inclusion of forgetting in the computer model would again raise many more problems than it would solve but should prove to be of interest.

Attribute Structure

The Mark IV, Mod 2 computer model involves approximately 25 attributes under which various types of information can be stored. These attributes were divided into class attributes and specific attributes, and certain mechanics were invented in order for the computer program to ascertain what information

was available under these attributes. For example, under a class attribute chunks of information are available; under a specific attribute, unique items of information exist which can be extracted. The attributes employed were a function of the particular experimental situation modeled and represent an initial approach to the exceedingly difficult task of describing behavior. The next logical step appears to involve creating both class and specific attributes from a minimal set of basic descriptive attributes, but the logical basis for defining such a basic set of attributes is not presently obvious to the author. Given such a basic situation, it does appear to be quite possible for the computer program to create both class and specific attributes when required by the situation. Thus, the computer model could handle the descriptive processes using its own capabilities. The attributes currently used were devised by the computer programmer and as such merely identify or label different units of information which he believed necessary. However, to shift this responsibility from the programmer to the computer program is a major step which clearly needs to be taken.

Use of Protocols

The think-aloud protocols, especially those with experimenter interrogation of the subject, have been an excellent device for eliciting the grosser behaviors exhibited by subjects within a concept-attainment task. The think-aloud protocols have been extremely disappointing in providing answers to the more fundamental questions which now need to be answered. It seems that the "state of the art" limitation in protocol analysis was reached, and it is difficult to elicit much more information from the protocols than has been extracted. The failure of the protocols to provide answers to questions about the internal mechanisms of human subjects, such as contexting and memory, suggests that new techniques of psychological inquiry are desperately needed in order to study covert behavior.

Contexters

During the early phases of the present modeling project, the computer model consisted essentially of the strategy list with its Z routines, P routines, and Q routines. Upon more detailed fractionization of the

computer program itself, it became increasingly apparent that the central executive function had to be separated from the operational function. There are actually two parallel processes which occur as a human being solves a problem. One was designated the contexting process which is the monitoring, supervising, goal-directing aspects of human behavior, i.e., the higher level, cognitive processes. The second is the operational aspects involving what one might call the subject's abilities, habits, or mechanisms. Once the difference between the contexting program and the operational program was conceptualized, a major restructuring of the computer program was possible and made for significant differences in the model of cognitive behavior.

A contexter can be viewed as creating a plan or strategy for behavior. At high levels in the model it creates plans for overall behavior and at low levels it creates plans for very specific actions. Such planning hierarchy was first envisioned by Miller, Galanter, and Pribram [1960] when they suggested the existence of plans which create plans. Their scheme and the present hierarchy of contexters have two implications for the internal organization of a computer model. First, the program must be organized so that it can treat itself as data; second, a contexter must be able to create programs from the "abilities," i.e. subroutines, possessed by the "subject." In the first case, the contexter routines must be able to analyze, modify, and otherwise manipulate the computer program itself. Without such abilities, the contexters cannot improve the subject's performance as a function of experience. The mechanics of treating the total program, including the contexters, as data can be accomplished through interpreter schemes such as that programmed by Baker and Martin [1965] in which the strategy or plan is a list of symbols representing behaviors. However, the symbols are executed by means of an interpreter rather than directly in the underlying language. Because these symbols are placed on lists, they can be treated, through the list processing language, as if they were data and can be manipulated by the contexter routines. Such a requirement rules out compiler-type list processing languages which are not capable of modifying the source language at execution time of the object program. Thus, it would appear that more sophisticated interpreter-type languages such as IPL-V will need to be developed. It should be pointed out that the Baker-Martin scheme divorces the contexting operations from the interpreter as the contexters are also executed by the interpreter.

The lack of differentiation between data and program means that both must share a common internal representation and that the internal organization of the computer program must facilitate both the storage and retrieval of information in some uniform fashion. In most existing computer models, the memory processes have been avoided by having the computer programmer remember where he stored the information and recall it for the program via the code he writes. Under an adequate computer model, the program should decide what should be stored and store it so that the program can retrieve it through its own recall mechanisms when the information is required. Uniformity of storage and retrieval in the present model has been implemented through the use of a modular memory structure accompanied by basic remembering and recalling routines which are a function of the structure of the memory rather than the list processing language employed. However, the programmer still decides what to remember and when to recall the information.

In addition to devising a system through which the program can be manipulated, it is necessary to provide contexters with the capability of creating new programs based upon new generalizations inductively acquired; i.e., the contexters should be programs which can create programs. Because the lowest level of detail in a computer model consists of basic processes which can be executed, i.e., the "abilities" possessed by the subject, all other levels of a computer model can be composed of the symbols which represent these basic processes. Hence, the procedure for creating new processes, plans, and contexters consists of restructuring these basic processes in an appropriate order. However, if the context routines are to have the capability of creating plans, they must "know" or be able to ascertain the capabilities of the basic processes and of the higher level routines which derive from them. There is a crucial and as yet unresolved requirement for being able to describe the characteristics and capabilities of a behavior regardless of the level at which it appears in the computer model. One rudimentary way is to consider a process as a transformation and use its inputs and outputs to describe the nature of the transformation. However, Newell [1962] indicates that this is not an adequate description. It would appear that the most feasible method would be to design computer models which manipulate verbal material as in the linguistic approach described in Chapter III. Regardless

of how the description problem is solved, it is quite clear that unless it is solved, progress in computer models will be very slow. It would appear that Newell's "solution by understanding" requires a prior "solution by description."

Programming Techniques

A number of computer programming techniques were developed by the project staff. The foremost of these techniques was the pseudocode interpreter system which enables representation and execution of the model as a list of symbols. The pseudocode scheme also provides the mechanical basis for the capability of the contexters to create programs from existing programs; a major unsolved task is the conceptual basis for such a capability.

The circular memory structure and its generalized remembering and recalling routines hopefully provide the basis for future computer programs which can perform these processes without human supervision. Again the mechanics have been provided but the requisite knowledge upon which to base the processes is not available.

The development of computer programs in which the program can be treated as data and new behavior sequences can be created requires that the computer model be independent of the mechanics of the language in which it is coded. In any programming language there are a large number of necessary housekeeping tasks which are unrelated to a computer model of cognitive behavior. For example, in IPL-V one must erase unneeded lists, push and pop the HO communication cell, and make copies of lists. If the computer program is to truly be a model, it should not be cluttered by additional features which take account of the housekeeping details associated with the underlying programming language. Freedom from such mechanical details can be accomplished through the use of an interpretive system, such as the pseudocode in the Baker-Martin [1965a] scheme. Alternately, if a "solution by description" were achieved, it could serve as the basis for the development of a compiler-level modeling language. One could then model the cognitive behavior in this language and be freed from the underlying list processing or other such language. Regardless of the method, the computer model needs to be freed from the housekeeping mechanics of the underlying programming language.

RESEARCH IDEAS GENERATED BY THE COMPUTER MODEL

1. In that the total computer model was developed around the idea of a strategy or plan, there exists a need for more information on what processes a subject uses to create plans and on the role of instruction in forming such plans. As was observed above, the present computer program assumes that the experimenter's instructions have a crucial role in the establishment of at least a gross plan of behavior. Unfortunately the planning experiment did not indicate that various levels of completeness of experimenter's instructions had any effect. Nonetheless the basic validity, or lack of validity, of the Miller, Galanter, and Pribram [1960] plans has not been demonstrated.

2. When one considers the vast realm of behavior which human beings are capable of exhibiting, it is remarkable that in a given situation they normally produce behavior relevant to the problem at hand. It may not be effective in a given situation, but usually it has some possibility of being useful. One of the outstanding features of the protocols was that almost all of the subjects very quickly produced a plan for solving concept-attainment problems. If subjects had not been able to select behaviors rapidly and appropriately, it would have taken a much greater period of time for them to solve these types of problems. Therefore, an important area of research is how humans select a specific behavior from their repertoire of possible behaviors.

3. In that the communication between the experimenter and the subject is minimal in the concept-attainment experiments, it is unusual that subjects can maintain a sense of goal-directedness during the entire experiment in the absence of many external clues. The relationship between what the subject sees as the task to be accomplished and the kinds of information he utilizes to ascertain whether he is making progress toward that goal needs to be investigated quite carefully. Analysis of the protocols showed that most subjects had some understanding of whether or not they were going in the correct direction despite the lack of external clues. It would be very interesting to ascertain what types of internal information they were utilizing to maintain this goal-directedness.

4. The memory entry point which was created to maintain some sense of order in the circular memory structure raises many questions about how people store information and, more importantly, how they get it back once it has been stored. The nature or struc-

ture of information stored in the human brain is not intuitively obvious. Subjects are adept at getting the information at the proper time without any great amount of visible effort. Logical analysis of the concept-attainment problem suggested that people followed some type of memory entry point sequence in that they tend to remember information about what they are currently working with without much concern for the details of the previous operation. However, many alternative models are equally reasonable.

5. Much effort was devoted to trying to introduce within-problem variability into the computer model. It was severely hampered by a lack of understanding as to how people make errors. Stimulus-response psychology has traditionally blamed errors upon the stimulus materials; however, our model tends to indicate that these errors are more likely due to errors in the contexting operations and errors in internal description rather than in the stimulus materials themselves. It would be most helpful for someone to conduct experiments which try to obtain some understanding of how humans make errors in the internal processing of data.

6. One of the large what one might call fudge factors in the current program was the awareness flag developed after the protocols showed that many subjects inadvertently worked with less than the full set of dimensions. In some cases it was clearly a perceptual problem; in other cases, it was possibly an oversight. If a subject was asked to name the dimensions, he would mention all five, yet in working on a given problem, he might deal with fewer. The behavior raises a question of how people decide upon using less than the full information. There are two sides to this coin, one of which is when the subjects know they are using less than the full amount of information and the other is when they do not. The interesting facet in the latter case is why don't they know?

7. Analysis of the protocols indicated very clearly that people do not remember the sequence of object choices; rather they remember strategies and reconstruct rather than recall the sequences. Such an observation raises many questions about the concepts currently in vogue about memory and what is stored. The protocols led to the distinct impression that people remember extremely little detailed information but do remember with great fidelity the strategies, procedures, and processes necessary to reconstruct the sequence of events. It appears that people keep detail around just long enough for it to be of

some use. However, any information stored for a longer period of time is usually stored in the form of a procedure, i.e., a strategy accompanied by enough basic information to repeat the process itself. Such a conceptualization of memory enhances the idea of the working memory and short-term memory, where working memory keeps the details just long enough for them to be used and short-term memory keeps enough of the salient information so that the process can be repeated. It would appear that the long-term memory is devoid of a great amount of detail but contains strategy lists and the necessary and sufficient amounts of crucial information to execute the strategy. However, the mechanisms by which people reconstruct rather than recall are not obvious and seem to be a good topic for future research.

8. During the development of the short-term memory, it was observed that the information was stored in a highly interlinked fashion, no matter what structure of memory was used. The existence of such extensive interlinkage seemed to suggest that interference in memory could be caused by access to inappropriate information resulting from the excessive linkages of the stored data. It would be very interesting to perform some experiments in which one deliberately caused subjects to remember certain types of linkages and then try to observe the amount and nature of interference that occurs due to the preconstructed linkages.

The types of information that are required for further development of the present computer model along the lines indicated suggest a rather different realm of psychological research than is usually reported in the literature. The concern is with what the subject does rather than what the experimenter does. In most current psychological literature, the experimenter is actually varying the material, etc., and little other than relatively gross outputs is ever attributed to the subject. The protocol analyses have shown that these gross outputs are not really informative about the processes, procedures, and so on utilized by the subject. In essence what is needed is some research in depth as to what subjects *do* in experimental situations rather than what they *produce*.

THE STATE OF OUR ART

The listing of Mark IV, Mod 2 is provided as Appendix C and includes a symbol cross reference. The program was included to enable those familiar with IPL-V to match what

was said about the program with what the program actually does. Adequate documentations of large, complex computer models is a difficult task and many discrepancies will probably be noted.

One of the unfortunate realities of computer modeling research is that it is a very lonely endeavor. The total number of such researchers is very small, their distribution sparse, and their interests very specialized. It was difficult to conduct a meaningful dialogue even with others interested in simulation of concept attainment, as each researcher has his own frame of reference and, once beyond generalities, such a dialogue requires intimate knowledge of the details of the models discussed. Inadequate documentation of published computer models is partially at fault here, but in many cases adequately documented programs are complex and deviously interdependent so that what is said and what happens is discrepant. During the course of the present project Professors L. Uhr, L. W. Gregg, R. C. Calfee, R. L. Venezky, and the author held a series of informal faculty seminars in which it was possible to reach the level of detail necessary for meaningful discussion of various computer models. These seminars were extremely fruitful and are needed on a much broader scale if the field is to progress.

Looking back to the beginning of the project, it can be seen that considerable progress has been made. With each new version of the computer program, the problems attacked were more sophisticated and the unsolved problems exposed more irretractable. What was once conceived of as a simple problem in computer application has become an extremely complex problem requiring answers to questions which are far beyond existing knowledge. At present a complete restructuring of the conceptual basis of the program from plans and strategies to language processing appears to be required. The linguistic approach coupled with many of the techniques derived within the present project would initiate a new approach to the modeling of human behavior. Such an approach might prove to be the stepping stone to the desperately needed new techniques for studying covert human behavior.

The concept-attainment program currently available, namely Mark IV, Mod 2, is a very rudimentary model of the concept-attainment process and of itself does not exhibit a great deal of what a specialist in simulation would call "interesting behavior." However, the author has not been overly concerned about this aspect as the computer program essentially represents a repository for ideas about

the concept-attainment process acquired to date. From this point of view, the program can be considered quite successful in that a reasonable understanding of at least the grosser mechanics of the concept-attainment process was obtained. In the modeling of the concept-attainment process, many problems have not been solved, but the modeling process has provided a good idea of what problems need to be solved in order for further progress to be made.

The sequence of events occurring within the present research project has followed a pattern typical of most computer modeling research, namely rapid early progress which suddenly reaches an asymptote of no progress. That this pattern exists was forcefully presented by H. L. Dreyfus [1965] in the RAND report entitled "Alchemy and Artificial Intelligence" where he drew the analogy between the alchemist's early success in extracting quicksilver from what appeared to be dirt, which resulted in a fruitless attempt to turn lead into gold, and the early success in the simulation field. An early success-long term failure pattern has been observed in nearly all aspects of computer modeling—problem solving, learning, chess playing, theorem proving and so forth. Dreyfus felt that simulation of cognitive processes reached their developmental asymptote very quickly and that future progress was limited by the inability of present computers and computer

models to handle what he called "fringe consciousness." By this term he means the wide array of subtle information which a human unconsciously draws upon when performing any cognitive task. Fringe consciousness would include information such as problems done in the past, psychological qualities of the experimenter's voice, remote prior experience in other areas which can be transferred to the present problem, and so forth. Dreyfus states that the range of such information is so great that, even if it could be enumerated, digital computers could not search it in reasonable time. The present computer model encountered the fringe consciousness problem and many of the "fudge" factors were simply attempts to avoid the problem. It appears that the fringe consciousness problem is unsurmountable if attacked head on. The only reasonable solution would seem to be to develop some creative high level theoretical abstractions that enable one to circumvent the necessity of including a human's complete range of experience within a computer model.

On the basis of the author's experience with the present research and what Dreyfus has described as the typical pattern in most modeling of cognitive behavior, it is evident that future progress is going to be slow and require the expenditure of considerable intellectual effort.

REFERENCES

- Baker, F. B. An IPL-V program for concept attainment. *Educational and Psychological Measurement*, 1964, 24, 119-127.
- Baker, F. B. The development of a computer model of the concept attainment process: A preliminary report. *Occasional Paper No. 5*. Research and Development Center for Cognitive Learning, the University of Wisconsin, 1965(b).
- Baker, F. B. CASE: A program for simulation of concept learning. *American Federation of Information Processing Societies Conference Proceedings*, Fall Joint Computer Conference, 1965(c), 27, 979-984.
- Baker, F. B. The internal organization of computer models of cognitive behavior. *Behavioral Science*, 1967, 12, 156-161.
- Baker, F. B., & Martin, T. An IPL-V technique for simulation programs. *Educational and Psychological Measurement* 1965(a), 25, 859-865.
- Brian, C. R., & Goodenough, F. L. Relative potency of color and form perception at various ages. *Journal of Experimental Psychology*, 1929, 12, 197-213.
- Bruner, J. S., Goodnow, J. J., & Austin, G. A. *A study of thinking*. New York: Wiley, 1956.
- Dreyfus, H. L. *Alchemy and artificial intelligence*. Report P-3244. The RAND Corporation, Santa Monica, California, 1965, p. 90.
- Hunt, E. B. *Concept learning: An information processing problem*. New York: Wiley, 1962.
- Hunt, E. B., Marin, J., & Stone, P. J. *Experiments in induction*. New York: Wiley, 1966.
- Johnson, E. S. An information processing model of one kind of problem solving. *Psychological Monographs*, 1964, 78 (4, Whole No. 581).
- Klausmeier, H. J., Harris, C. W., & Wiersma, W. Strategies of learning and efficiency of concept attainment by individuals and groups. U.S. Office of Education Cooperative Research Project No. 1442. Madison: University of Wisconsin, 1964.
- Laughery, K. R., & Gregg, L. W. Simulation of human problem-solving behavior. *Psychometrika*, 1962, 27, 265-282.
- Miller, G. A., Galanter, E., & Pribram, K. *Plans and structure of behavior*. New York: Holt, 1960.
- Newell, A. Some problems of basic organization in problem-solving programs. In M. Youitts, G. T. Jacobi, & A. D. Goldstein (Eds.), *Self-organizing systems*. New York: Sparten, 1962.
- Newell, A. *A guide to the general problem-solver program GPS-2-2*. Memorandum RM-3337-PR RAND Corporation, Santa Monica, California, 1963.
- Newell, A. (Ed.) *Information processing language-V manual*. (2nd ed.) Englewood Cliffs, New Jersey: Prentice-Hall, 1964.
- Newell, A., Shaw, J. C. & Simon, H. A. Elements of a theory of human problem Solving. *Psychological Review*, 1958, 65, 151-166.
- Simon, H. A., & Kotovsky, K. Human acquisition of concept for sequential patterns. *Psychological Review*, 1963, 70, 534-546.
- Uhr, L., & Vossler, C. A pattern recognition program that generates, evaluates, and adjusts its own operations. *Proceedings of the Western Joint Computer Conference*, 1961, pp. 555-569.

APPENDIX A
THINK-ALoud PROTOCOL

Problem 1 (Concept is short neck, bent tail)

- E: This card belongs to the concept. (Focus card is blue, s. neck, s. ears, b. tail)
- S: Ah, lets see it is blue and has a short neck, and has a straight tail. Ah
- E: This card.
- S: This?
- E: Yea
- S: Short neck, straight tail
- E: No that tail is bent.
- S: Oh wait!
- E: See this has a straight tail. You can compare them.
- S: Oh, that's bent and that's straight. Alright, I was looking at this, so I thought that this was the real bent one.
- E: Oh, I'm sorry.
- S: In other words these are the same categories.
- E: Well I call them curly.
- S: So there is another category.
- E: There are three kinds of tails yea.
- S: Let me see.
- E: Straight, bent, and curly.
- S: O.K., Ah
- E: Or you can call them what ever you want to.
- S: It has a bent tail, it's blue, short neck, short ears. Um. Let me see, uh
- E: What are you looking for?
- S: I'm looking for the same thing in another color to see if color is one of the categories, one of the characteristics. Is this the one? It's brown, has short neck, and bent tail. (brown, s. neck, s. ears, b. tail, varying only color)
- E: Yes, that does belong.
- S: So then it doesn't matter what color it is. Um, I'll find one in yellows to see if... I can see this one in yellow. Does that belong (yellow, s. neck, s. ears, b. tail, varying only color again)
- E: Yes that belongs.
- S: Well, three of them are the same exactly except they're in different colors, there-
- fore, the one I'm looking for... it doesn't matter what color the one I'm looking for is. So the characteristics are, let me see, um... I'm going to find out if the tail has to be bent or not, so I'll take one that has a straight tail and no ears.
- E: What are you looking for?
- S: I'm looking for, oh, here's one with a straight tail, and big ears. Oh wait, a straight tail and small ears. Does that fit in? (brown, s. neck, s. ears, str. tail, varying color and tail)
- E: No, that does not belong.
- S: So obviously the tail is the one, the characteristic that uh, rules that one out. Does this belong? It has a long neck and a curly tail. (brown, l. neck, s. ears, curly tail, varying color, neck, and tail. But S should have learned that tail is relevant)
- E: No that does not belong. So now what are you thinking?
- S: I'm thinking that, well, something with a curly tail does not belong in the category. Is it possible that the card I'm looking for must have any color, must have a bent tail, and no neck, and short ears, or short neck and short ears.
- E: Are you guessing at it now?
- S: Well Uh-hum actually I am.
- E: Would you mind repeating it.
- S: Bent tail, short neck, no ears, uh, short ears, and it doesn't matter what color it is.
- E: No it is not correct.
- S: Well I will try and rule out some other category. Does uh, Does this belong? Has all the characteristics that the first part had, but it has big ears. Does this belong? (blue, s. neck, l. ears, b. tail, varying just ears)
- E: Yes it does belong.
- S: Oh, so big ears are part of the category. Uh. Does this belong? (blue, s. neck, l. ears, curly tail, varying ears and tail. S doesn't seem too sure of tail)

E: Why are you asking?
 S: Because it has big ears, and a curly tail.
 E: No that does not belong.
 S: Then the curly tail must be a characteristic that doesn't belong. Does this belong? Has a straight tail and big ears. (blue, l. neck, l. ears, str. tail, varying neck, ears, and tail)
 E: No that does not belong.
 S: Then a straight tail does not belong. Well, then after searching out all the characteristics, I feel that the card I'm looking for must have any color, but it must have a bent tail, and a short neck, but it can have any kind of ears.
 E: That is correct. Uh-hum, O.K.

Problem 2 (Concept is yellow, long neck, curly tail)

E: This card belongs to the concept. (Focus card is yellow, l. neck, s. ears, c. tail)
 S: Yes, it has a long neck, and it's yellow, and it has a curled tail, but has no ears. Now I'm going to test for, color I think first. Uh. Here is the same thing in blue, long neck and curled tail, and small ears. Does that fit? (blue, l. neck, s. ears, c. tail, varying just color)
 E: No that does not belong.
 S: So obviously it has the same characteristics, but a different color, then the color rules it out. Does this belong? Oh, wait now I'm sorry.
 E: What are you looking for?
 S: A brown animal with long neck, small ears, and a curled tail. But I don't seem to find one. Here's one. Does that fit into the category? (brown, l. neck, s. ears, c. tail, varying color again. In all these problems, S always varied color first and checked it twice.)
 E: No this does not.
 S: Well I tested for the two other colors shown on the chart therefore, I feel that color is a characteristic that rules an animal out, so the color of an animal must be yellow. Now I will check for the size of the neck. Uh. Here is the same card, but only that it has a small neck, same animal. Does that fit in? (yellow, s. neck, s. ears, c. tail, varying only neck)
 E: No that does not belong.
 S: Well then the size of the neck is another characteristic that I'm looking for. It must have a long neck. Uh. The same card with a long neck, and big ears. Wait

now I'm sorry. Here, does that card belong? (yellow, l. neck, l. ears, c. tail, varying only ears)

E: Yes, it does belong.
 S: So it doesn't matter what size the ears are? Here is an animal. I'm looking for an animal with a short neck to see if it is the size of the neck. (But S just checked the neck) Oh here, well it has a curled tail. Oh does this animal fit in it has a bent tail? (yellow, s. neck, s. ears, bent tail, varying neck and tail. S is either not paying attention or he has a very short memory.)
 E: No this does not belong.
 S: Does the
 E: What did that tell you?
 S: Well it told me that, oh wait, I haven't tested really for a bent neck. Does this card with the short neck and curled tail fit in? Yes it does. (yellow, s. neck, s. ears, c. tail, varying neck. Same as 3rd choice. It seems that S forgot the designation rather than forgot he chose it.)
 E: No it doesn't.
 S: Oh, it doesn't!
 E: Did you forget this?
 S: Yes I forgot that. So the size of the neck does matter. Uh. I'm looking for, oh here's one. An animal with no ears, a long neck, and a bent tail. Does that belong? (yellow, l. neck, s. ears, b. tail, varying just tail)
 E: No it doesn't belong.
 S: Well that tells me that the bent tail is out. Now I'm looking for an animal who has a long neck, and a curled tail, and no ears. It there one? Oh, it's the only one there. I think that I have found it. The animal must be yellow, must have the curled tail, must have a long neck, and must have no ears, or small ears.
 E: Short ears?
 S: Short ears.
 E: No that's not correct.
 S: Oh wait, I think that I tested for the ears. Then it must have all the characteristics, but it doesn't matter what ears.
 E: O.K. (Laughed) That's better.

Problem 3 (Concept is brown, short ears)

E: This card belongs to the concept. (Focus card is brown, l. neck, s. ears, b. tail)
 S: It is a brown animal, it has a long neck, it has short ears, and a bent tail. Uh, I would like to test for color first, so I will find the same animal in a different color, and, see, long neck, bent tail, and

no ears, um. Does this animal fit in? (blue, l. neck, s. ears, b. tail, varying color)

E: Uh. No this does not belong.

S: Well obviously the blue animal with the same characteristics doesn't fit in, so I will look for a brown animal, and see if that fits in. The same characteristics. Oh I mean a yellow animal. Does this animal fit in? (yellow, l. neck, s. ears, b. tail, varying color again)

E: No it does not belong.

S: Well that tells me that color is a characteristic that rules an animal out. Now I will look for the same animal with a straight tail. Does this animal fit in, the same color, but a straight tail? (brown, l. neck, s. ears, str. tail, varying only tail)

E: Which one? Yes that does belong.

S: So it does not matter if the tail is bent or straight. I will find one with a short neck. Does this animal fit in? (brown, s. neck, s. ears, str. tail, varying neck and tail)

E: Yes that does belong.

S: It doesn't matter if the neck is short or tall, but this animal does have short ears. I will find one with big ears. Does this animal fit in? (brown, l. neck, l. ears, str. tail, varying ears and tail)

E: No it does not.

S: Well I feel that the animal must be brown, and must have long or short neck, and must... Oh, I haven't tested for a curled tail yet. Does this animal fit in? (brown, s. neck, s. ears, curly tail, varying neck and tail again)

E: Yes it does belong.

S: Well then the animal must be brown, must have a long or short neck. Must have, well it doesn't matter what size neck, or what kind of tail it has, but it must have small ears.

E: O.K., that is correct. Um-hum. (S follows a conservative strategy and varies all the values of a 3-valued dimension)

Problem 4 (Concept is brown, short neck, straight tail)

E: This card belongs to the concept. What is the first thing you think of when I point a card out to you? (Focus card is brown, s. neck, s. ears, str. tail)

S: What do you mean the characteristics, or just the first...

E: Well the first thing you think of.

S: I think of a dachshund. It looks like a dachshund.

E: (Laughed) NO, I mean, you know,

S: Brown is the first thing, and the fact that it has a short neck, and no ears, so I think that is pretty important, and a straight tail and no other characteristics. Uh.

E: Then what do you think? Continue

S: I'm trying to decide, I'm looking for a card that's the exact same thing but a different color, and here is one in blue. (blue, s. neck, s. ears, str. tail)

E: No this does not belong.

S: Well I will find one in yellow and see if yellow belongs.

E: You usually test color first?

S: Yes I do. I think that is a good way to start anyway. I guess it really doesn't matter, it is the easiest, I feel, if you differentiate in color because then you can look for other characteristics. Does this animal, yellow one fit in? (yellow, s. neck, s. ears, str. tail)

E: No it does not belong.

S: That tells me that the animal that I'm looking for must be brown, so it narrows down the field, it is very easy to find animals by color rather than other characteristics, which aren't as visible. Uh. I'm checking the tail. Does this animal fit in? Oh wait, I'm sorry. I'm looking for a short neck. Does this animal fit in? (brown, s. neck, s. ears, b. tail, varying just tail)

E: No it does not belong.

S: Tells me that the tail must be straight. Does this animal fit in? The straight tail and long neck. (brown, l. neck, s. ears, str. tail, varying just neck)

E: What do you want to find out?

S: The neck.

E: No that does not belong.

S: It tells me that the neck must be short. Does this animal fit in? It has big ears. (brown, s. neck, l. ears, str. tail, varying just ears)

E: Yes it does belong.

S: Well that tells me that the animal must be brown, must have a straight tail, must have a short neck, and big ears. (S has just described the last card he has chosen.)

E: And big ears?

S: No, small ears. Oh wait it doesn't matter what ears.

E: O.K. That is correct.

Problem 5 (Concept is short ears)

E: This card belongs to the concept. (Focus card is blue, s. neck, s. ears, curly tail.)

S: It is a blue animal with a curled tail, short neck, no ears. I'm going to check the

- color first so it is the same animal but a different color. (brown, s. neck, s. ears, c. tail, varying color)
- E: Uh-hum. Yes it does belong.
- S: Brown belongs, I'll see if yellow belongs too. (yellow, s. neck, s. ears, c. tail)
- E: Yes that does belong.
- S: Then that tells me that it doesn't matter what color it is. I will check for the tail first. Does this animal fit in? (yellow, s. neck, s. ears, b. tail, varying color and tail)
- E: Yes that belongs too.
- S: Well then that tells me that it doesn't matter if the tail is bent. Does this animal fit in? Let me see I'm looking for a curled tail. Oh wait that is a curled tail, I'm sorry, I'm looking for a small animal with a straight tail. Here it is. Does this animal fit in? (blue, s. neck, s. ears, str. tail, varying tail. Usually after checking both other values of color, S checks both other values of tail.)
- E: Yes it does belong.
- S: So that tells me that it doesn't matter what color it is or what kind of tail it has. Now I'm checking for a long neck. Does this animal fit in? Has a long neck. (yellow, l. neck, s. ears, c. tail)
- E: Yes it does belong.
- S: Then that tells me that it doesn't matter what neck it has. Does this animal fit in? (yellow, s. neck, l. ears, str. tail)
- E: Why are you asking that?
- S: It has the characteristics that are acceptable. But it has big ears, and I haven't checked for that yet.
- E: No that does not belong.
- S: That tells me that the animal may be any color, and that it may have any size neck, may have any kind of tail, but it must have small ears.
- E: That is correct.

Problem 6: (Concept is curly tail)

- E: That card belongs to the concept. (Focus card is brown, short neck, s. ears, c. tail)
- S: That one.
- E: Yea.
- S: Uh. It has a short neck, small ears, and a curled tail, and it's brown. I will find the same thing in a different color. Lets see. Wait. Does this fit in? (blue, s. neck, s. ears, c. tail)
- E: Yes it does belong.
- S: Does this fit? (yellow, s. neck, s. ears, c. tail)

- E: Yes it does.
- S: That tells me that it doesn't matter what color is it. I'm looking for the size of the neck now. Does this fit in? (yellow, l. neck, s. ears, c. tail, varying color and neck)
- E: Yes it does belong.
- S: That tells me that it doesn't matter what size the neck is. Does this fit in? (yellow, s. neck, l. ears, str. tail, varying color, ears, and tail)
- E: No that does not belong.
- S: Ah wait that doesn't tell me anything because there are two different characteristics, I should have pointed to something else. Does this fit in? Oh wait, yea, Does this fit in? (blue, s. neck, l. ears, c. tail, varying color and ears)
- E: What are you trying to find out?
- S: What size the ears are.
- E: Yea that belongs.
- S: That tells me that it doesn't matter what size the ears are. Does this fit in? (blue, l. neck, l. ears, c. tail, varying everything except tail)
- E: Yes that belongs.
- S: Well thats for the long neck. Now have I tested for everything else? Does this fit in? (blue, l. neck, l. ears, b. tail, varying everything. S should have known the concept 2 choices ago.)
- E: Which one, the blue one?
- S: Yea.
- E: No that does not belong.
- S: Does this fit in, with the straight tail? (blue, s. neck, s. ears, str. tail, varying color and tail)
- E: No that does not belong.
- S: That tells me that I'm looking for an animal that can be of any color, can have any size neck or any size ears, but must have a curled tail.
- E: That is correct. O.K. Tell me what your impression of this board is and the procedure and everything. Any thing you would like to comment about.
- S: Uh-hum. I think it is a good way to test, well first of all the colors are good because I think you can differentiate between them pretty easily, and so that is the immediate stimulus I guess. The immediate thing that I see is a difference in color the first thing, and then if you distinguish between the colors first then you can find the different characteristics. It can be sort of confusing, forgetting if it has a bent tail, straight. I really don't see the purpose of it all. Unless it is an I.Q. or something.

E: Well this is just to see the little detailed thought processes that is behind solving problems like this.

S: Uh-hum.

E: That is all that you have to say on the subject?

S: Yea I think so.

E: What is your general strategy of solving the problem?

S: Well first I try to solve the problem with color. I think that it is easiest to differentiate, between colors, and then I usually look for neck first, and then, because that is another easy to differentiate because

it is outstanding, and then for the tail, and the ears, it doesn't really matter which way you go about it because they are both equally as easy to see.

E: Uh-hum. O.K. That will be all for today. (S always picked color first. Then he would pick tail, not neck, in 2/3 of the problems. After that, neck, then ears. This S had been run initially on the old board of circles and triangles on colored paper. That is where he developed his conservative strategy. I ended this session early because S was getting very bored by that time.)

APPENDIX B
LISTING OF CASE MARK IV. MOD 2

IPL LIST

	9		1
	2 A	600	2
	2 C	300	3
	2 D	300	4
	2 E	300	5
	2 F	300	6
	2 G	300	7
	2 K	300	8
	2 L	300	9
	2 M	300	10
	2 N	300	11
	2 O	300	12
	2 P	600	13
	2 Q	300	14
	2 R	300	15
	2 S	300	16
	2 T	300	17
	2 U	300	18
	2 V	300	19
	2 X	300	20
	2 Z	300	21
	5		22
MARK 4 MOD 2 CONTEXTER	1		23
UI PROGRAM INTERPRETER	1		24
EXECUTE RECURSION	U1	9-1	25
UNRECURSE FIRST RECURSION		309-7	26
OUTPUT FINAL H5		0199 0	27
9-1 RECURSION ROUTINE	1		28
PUSH DOWN RECURSE LIST LIKE H1	9-1	409-7	29
SAVE ARG IN HEAD		209-7	30
BEGIN OR CONTINUE SCAN	9-5	9-3	31
POP UP A LEVEL IF END LIST		700	32
TEST FOR LOCAL SYMB	9-14	129-7	33
IF YES THEN H5 DECISION JUMP		J132	34
GOTO 9-4 IF DECISION JUMP		70 9-4	35
CHECK FOR DESCRIBED ROUTINE		1297	36
I.E. HAS A LOCAL HEAD		52H0	37
TEST		J132	38
IF NOT EXECUTE THE SYMB		709-6	39
		1297	40
		J152	41
YES GET ARGS FROM D-LIST		129-7	42
COLLECT OUTPUT ARGS		U3	43
YES GET ARGS FROM D-LIST		129-7	44
		U2	45
		1297	46
CHECK FOR CONTEXT ROUTINE		10A6	47
		J10	48

YES GO TO 915	70	915	49
ENTER PR	1297		50

IPL LIST

RECURSE		9-1		51
9-2 UNRECURSE	9-2	309-7	9-5	52
915 EXECUTE CONTEXTER VIA U4	915	30H0		53
COPY CIA LIST		1097		54
		J73		55
		U4	95	56
9-7 H1 TYPE LIST	9-7	J	0	57
SAVE H5 OVER INTERPRETER	9-9	J4	0	58
9-3 SCAN ROUTINE	1			59
SCAN LIST NAMED IN HEAD OF 9-7	9-3	119-7		60
DCWN 1		J60		61
SAVE LOC AND EXIT		2097	0	62
94 DECISION BR. CHECK PREV H5	94	J199		63
IF H5 - TAKE THIS LOCAL		709-12		64
SCAN		9-3		65
UNRECURSE IF END LIST		J70		66
IF YES RESET 9-7 HEAD	9-12	129-7		67
WITHOUT RECURSION		2097	95	68
96 REQ UNDESCRIBED PR (SET SIGN)	96	0199		69
ROUTINE , EXECUTE CURRENT ONE		0297		70
SAVE H5 OVER INTERPRETER		11H5		71
THIS ROUTINE TO BE TRACED		2099		72
		10U5		73
		1297		74
		J77		75
		7095		76
NC		10M1		77
PRINT NAME ,CONTENTS M1,M10		10J150		78
		J100		79
		10M10		80
		J152		81
		10M10		82
		J81		83
		7095		84
		J150	95	85
U5 TRACE LIST (USE NON DESCRIBED	U5	0	0	86
*****1 *****				87
U2 COLLECT ARGS FROM D-LIST	1			88
ARGUMENT 0 ATTRIBUTE	U2	10A1		89
GET V-LIST OF ATT. A1		J10		90
NCNE EXIT		700		91
COLLECT ARGS		10J0	J100	92
*****1 *****				93
U3 COLLECT OUTPUT ARGS FROM	U3	10A2		94
D-LIST		J10		95
		700		96
		10J0	J100	97
*****1 *****				98
U4 CONTEXTER EXECUTER	U4	J44		99
SAVE COPY OF CIA LIST		60W4		100

IPL LIST

GET LOC CF CONTEXTER		52H0		101
GET CONTEXTER NAME		52H0		102
SAVE IT		60W3		103
LOAD OUTPUT NAMES		U3		104
LOAD INPUT NAMES		11W3		105
		U2		106
ENTER CIA LIST NAME		11W4		107
ENTER C NAME		11W3		108
GET EXECUTABLE CORE		J81		109

EXECUTE IT		J1		110
ERASE COPY CIA LIST		11W4		111
		J71	J34	112
*****1 *****				113
S2 CONSERVATIVE FOCUS STRATEGY	S2	90		114
PRCESS FOCUS INFORMATION		Z0		115
CREATE WORKING HYP		Z7		116
DOUBLE DUMMY BR		91		117
		91	0	118
CREATE SEARCH CRIT	91	0		119
		Z1		120
LOCATE OBJ FROM EXT ENVIRCNS		Z2		121
HAVE EXPERIMENTER DESIG. OBJ.		Z3		122
PRCESS OBJ. DESIG. INFORMATION		Z4		123
DETERMINE IF CONCEPT CAN BE GIVE		D1		124
NO		91		125
YES CONTINUE		93	J	126
FORM CONCEPT	93	0		127
YES FORMA CONCEPT		Z5		128
HAVE E DESIGNATE CONCEPT		Z8		129
DETERMINE IF CONCEPT CORRECT		D2		130
NO		92	0	131
CORRECTIVE ACTION	92	0		132
		Z6		133
GIVE UP		D3		134
N3,		91	0	135
	9C	J	J	136
*****1 *****				137
S1 STRAT SET IP	S1	J0		138
		10M13		139
MAKE M13 REFLECTIVE		R39		140
MAKE A3 LIST SYMMETRIC (UPPER)		10A3		141
		R39		142
	91	J140		143
		30H0		144
		11F5		145
		100		146
		J2		147
		70	J7	148
		10M10		149
		G30		150

IPL LIST

		10M1		151
		G30		152
PUT		10M1		153
EXPERIMENTER MESSAGE		10L60		154
IN M)		J65		155
EXECUTE GENERAL STRAT		10S3		156
		U1		157
PRINT PRGB HIST		X20		158
UNPARK DIMS AND VALS		10M13		159
		10A5		160
		R38		161
		J0	91	162
*****1 *****				163
X20 PRINT PROB. HISTORY	X20	11H5		164
PRINT H5		J152		165
PRINT CONCEPT		10M1		166
		J81		167
		7091		168
		J151		169
PRINT STRAT L	91	10M3		170
		J81		171
		40H0		172
		J150		173

PRINT Z,P LEXVELS		1JJ150	U6	174
	97	0	0	175
*****1 *****				176
C12 POST MGRTEM ANALYSIS	C12	JJ		177
SET NOT FIRST PROB FLAG		10L41		178
		J81		179
		1JF2		180
		10A65		181
		J11		182
		JO	J	183
*****1 *****				184
S3 HIGH LEVEL CONTEXTING LIST	S3	90		185
TRASL EXPTR MESSAGES		C11		186
EXPTR PRESENTS FOCUS		E95		187
CREATE STRATEGY TO EXECUTE		C61		188
POST MGRTEM ANALYSIS		C12	0	189
	9C	0	0	190
*****1 *****				191
C11 TRANSLATE EXP INST INTO SSL	C11	90		192
		C10	0	193
	9C	0		194
		A1		195
		91	J	196
	91	0		197
		M1		198
		M1J	J	199
*****1 *****				200

IPL LIST

SAVE MEP	C10	30H0		201
GET NAME OF MESSAGE		J81		202
LIST FROM M1		10C13	J100	203
*****1 *****				204
C13 CCNTRCL CONTEXTER	C13	J43		205
(O) MESSAGE		60W3		206
COPY MESSAGE		J74		207
		40H0		208
DELETE A6J AND VALUE		10A60		209
FROM COPY		J14		210
SELECT PROPER CCNSTRUCTOR		1097		211
VIA A6J VAL ON MESSAGE		11W3		212
DESCRIPY		10A60		213
		G10		214
FROM LIST 97		J10		215
EXIT IF NONE		70J33		216
EXECUTE CONTEXTER		J1	J33	217
97 LIST OF CONTEXTERS UNDER VALS 0	97	98	J	218
OF A60	98	0		219
PROBLEM DETAILS		F61		220
CREATE PROB LIST ETC		C50		221
PROCESS SPECIFICATION		F62		222
CREATE SLELETAL PROCEDURE		C51		223
ATTRIBUTE SPEC		F63		224
PLACE ATT ON LIST,SAVE VALUES		C52		225
TERMINATION		F65		226
NORMAL EXIT		JO	0	227
*****1 *****				228
C50 CREATE PROB LIST AND ABSORB	C50	J47		229
PRCB DETAILS(O) MESS SYMB		60W7		230
SAVE NAME OF		J60		231
DDL WITH		52H0		232
PROB DETAILS		60W4		233
SAVE		1JA50		234
VALUE OF		G10		235
A50=M13		20W2		236
PRCB LIST (L100)		J90		237

SAVE NAME		60W6	238
EXTERNAL ENVIRONMENT		J90	239
E9		6JW5	240
PUT E9 ON L100		10A300	241
		J12	242
PUT M13 CN E9		11W5	243
		11W7	244
		10A50	245
		J12	246
PUT DDL		11W6	247
ON PROB LIST (L100)		11W4	248
		10A80	249
		J12	250

IPL LIST

GET DDL		11W4	251
IS THIS		10A65	252
PROBLEM		J10	253
NOT IN		7093	254
THE		10F1	255
THE FIRST		J2	256
PUT SKELETON	93	11W6	257
ONE ATTEMPTED		7091	258
STRAT ON PROB LIST		J90	259
(L100)	92	60W3	260
		10A302	261
		J12	262
PLACE PROBLEM		10M10	263
LIST		11W6	264
CN MEP		G2 J37	265
NO GET RSL FROM	91	10M3	266
M3 LIST		J81 92	267
*****			268
C51 CREATE OPERATION SYMB FOR SSL	C51	J41	269
(0)= LXX		J60	270
GET 9-4 OFF LXX		52H0	271
		20W0	272
GET NAME OF		10M10	273
SKEL STRAT LIST		J81	274
UNDER A302		10A302	275
		G10	276
CREATE SYMBOL FOR 2 LEVEL ROUTIN		J90	277
		60W1	278
PUT NAME OF 2 ON SSL		J65	279
		11W1	280
		11W0	281
		10A3	282
		J12 J31	283
*****			284
C52 DESIGNATION INFORMATION	C52	J41	285
GET NAME OF CELL HOLDING DDL		J81	286
GET NAME OF 95		52H0	287
GET A7		J60	288
SAVE NAME OF CELL HOLDING A7		60W0	289
GET VALUE NAME		J60	290
		52H0	291
		20W1	292
PUT		10A15	293
A7		12W0	294
CN BODY OF A15		J65	295
		12W0	296
PUT 930 CN		11W1	297
		J73	298
POSSIBLE VALUES OF		10A111	299
		J11 J31	300

IPL LIST

*****1 ***** *			301
C101 FINDROUTINE ON (1) MATCHING	C101	J50	302
DES OF (0) CLASS ATT A3		1091	303
GEN RSL CELLS (1)		06	304
INVERT SIGN AND EXIT		J5 J30	305
SAVE PROCESS LOC FROM RSL	91	6097	306
CHECK IF DESES MATCH		11W0	307
DF (0)=DES OF (0)		10A3	308
		G210	309
		1197	310
PICK UP LOC OF EQ P		J5	311
INVERT H5		700 J8	312
EXIT			313
*****1 ***** *			314
C61 CONSTRUCT NEW STRAT	C61	90	315
		660 0	316
	9C	J	317
		A1	318
		91 0	319
	91	0	320
		M3	321
		M10 0	322
C60 MAJOR CONTEXTER(0)M10(1)	C60	J46	323
(1)RSL, C(M10)=L100,SAVE INPUTS		20W0	324
		60W1	325
GET RSL		J81	326
		20W6	327
GET SSL FROM L100		11W0	328
		J81	329
		10A302	330
		G10	331
SAVE IT		20W3	332
GET FERST PROB FLAG		11W0	333
		J81	334
UNDER A80 AND A65		10A80	335
		G10	336
NOT IN		7091	337
		10A65	338
		J10	339
NOT IN		7091	340
CHECK		10F1	341
		J2	342
		70 91	343
NOT FIRST		11W1	344
XEQ RECENT (1 DOWN IN M3)		J81	345
EXECUTE RSL		U1	346
PUT NSL IN LTM (M3)	95	11W1	347
		11W6	348
		G4 J36	349
FIST PROB	91	J90	350
SAVE STRAT AS CURRENT PHASE		60W2	

IPL LIST

MAKE,SAVE NSL		20W6	351
GENERATE NO LINK		J90	352
		J136	353
SAVE NO LINK		60W4	354
SAVE AS YES LINK (1ST TIME)		20W5	355
GENSSL		11W3	356
		1092	357
		J100	358
ERASE UNUSED YES NODE		11W5	359
		J9 95	360
LCCATE SSL SYMB DES	92	11W1	361
		J81	362
GN RSL		J6	363

		C101	364	
NCNE ERRCR		7099	365	
SAVE Z NAME		60W3	366	
ENTER PHASE LIST		11W2	367	
INVERT (C),(I)		J6	368	
ADD TO NSL		J65	369	
CHECK IF EXTERN INFO		11W3	370	
CHECK FOR EXTERN INFO		10A31	371	
		G10	372	
NO		7094	373	
OBJECT CONTESTERS		5097	374	
CHECK IF OBJ DESIG		11W3	375	
		10A33	376	
		G10	377	
NO,93		7093	378	
YES POP VAL		30H0	379	
ENTER CONTEXT L		50912	380	
ENTER INFO TYPE ATT	93	11W3	381	
GET TYPE		10A32	382	
		G10	383	
		70	96	384
POP CONTEXT LIST		30H0	94	385
GET PROPER CONTEXTER	56	J10		386
NCNE ,94		7094		387
DOUBLE NAME		40H0		388
ADD A6 AND STRAT NAME		11W2		389
		10A6		390
		G11		391
ACC		11W2		392
		J6		393
		J65		394
EXECUTE PHASE	910	11W2		395
		U1		396
SAVE SIGN	916	11W5		397
		20917		398
SAVE NAME OF PHASE XEQESD		11W2		399
		20915		400

IPL LIST

GET LAST SYMB		11W2		401
		J61		402
		52H0		403
CHECK IF DECISION		10A30		404
		G10		405
NO,EXIT		70J4		406
		10F31		407
		J2		408
		70J4		409
YES, ADD NO LINK		11W2		410
		11W4		411
		J65		412
ADD YES LINK		11W2		413
		11W5		414
		J65		415
SET YES LINK AS CURRENT PHASE		11W5		416
		20W2		417
GEN NEW YES LINK		J90		418
		J136		419
		20W5		420
CHECK SIGN (U1)		01917		421
+ EXIT		70	0	422
EXECUTE NO LINK IF -		11W4		423
		U1	J4	424
CHECK IF DECISION	94	11W3		425
		10A30		426
		G10		427

NO EXIT		70J4		428
DECISION VAL		10F31		429
		J2		430
NO EXIT		70J4		431
YES CHECK IF PHASE ALREADY XEQED		11W2		432
		11915		433
		J2		434
NO.910		70910		435
YES EXECUTE TEST AS SINGLE TON		11W3		436
		J91		437
		60915		438
		U1		439
XEG TEAST		11915		440
ERASE HOLDING CELLS		J71	916	441
ERROR	95	10C6J		442
		J152	J7	443
	917	0	0	444
	915	0	0	445
TABLE FOR CONTEXTERS	57	98	0	446
PER INFO TYPE	98	0		447
FDCUS		F51		448
		C21		449
OBJECT		F53		450

IPL LIST

OBJECT CHOICE		C37	0	451
TABLE OF CONTEXTERS	912	911	0	452
	911	0		453
		F53		454
OBJECT DESIG		C38		455
CONCEPT		F54		456
		C23	0	457
*****				458
C21 Z7 CREATION	C21	90		459
CIA INPUT ADDED BY U1		C20	0	460
	9C	J		461
		A3		462
		91	0	463
	91	0		464
		F32	0	465
C20 INITIALIZE Z7	C20	J44		466
SAVE CIA		60W4		467
		52H0		468
GET NEXT Z AFTER INTERRUPT		J81		469
		7092		470
IF Z7 ON STRAT ,EXIT		10Z7		471
PROCESS DES ATT.		10A3		472
CCMPARE		G210		473
YES EXIT		70	J34	474
COPY DES Z7	92	10Z7		475
		J74		476
SAVE		60W3		477
ERASE REMAINS AFTER HEAD		G30		478
PUT NWE Z4 ON STRAT	91	12W4		479
		11W3		480
		J64		481
PHASE NAME ON SELF		10C21		482
GET NAME OF PHASE		10A6		483
LIST FROM		J10		484
VALUE LIST,OUTPUT IS PHASE SYM		20W1		485
COPY		11W3		486
P191		10P191		487
PUT IT CN =Z7=		J65		488
PUT		11W3		489
P63		10P63		490
=Z7= BOCY		J65		491

PUT	11W3	492
C31	10C31	493
=Z7= BODY	J65	494
PLT	11W3	495
P64	10P64	496
=Z7=	J65	497
ADD	12W4	498
SYMBOL	10D4	499
TO PHASE 1 LIST	J65	500

IPL LIST

DESCRIBE THAT	11W1	501
Z7	11W3	502
ADDED TO PHASE 1 LIST	C3	503
DESCRIBE THAT	11W1	504
C4	10D4	505
ADDED TO PHASE 1 LIST	C3 J34	506
*****1 *****		507
C37 VERIFY OBJECT CHOICE	C37 90	508
CIA INPUT VIA UI	C36 0	509
	9C 0 0	510
C36 CREATE DO ROUTINE	C36 J44	511
SAVE(CIA)=C37	20W4	512
FIND	12W4	513
SYMBOL AFTER C37 ON	J81	514
STRAT LIST	10D0	515
IS IT	10A3	516
A DO TYPE ROUTINE	G210	517
	7J J34	518
CREATE	10D0	519
COPY	J74	520
CF DO	6JW3	521
ERASE BODY	G30	522
ADD	11W3	523
Q51	10Q51	524
TO DO LIST	J65	525
ADD	12W4	526
DO	11W3	527
TO STRAT LIST	J64	528
GET PHASE L	10C37	529
	10A6	530
	J10	531
DESCRIBE ADDITION OF	11W3	532
PHASE LIST	C3 J34	533
*****1 *****		534
C38 CREATE ROUTINE FOR Z4	C38 90	535
CIA INPUT VIA UI	C39 0	536
	9C J	537
	A1	538
	91 0	539
	91 J	540
	M10 0	541
C39 CREATE REACTION TO OBJ DESIG	C39 J44	542
SAVE CIA	20W4	543
SAVE MEP	20W0	544
GET PHASE L	10C38	545
GET	10A6	546
NAME	J10	547
LIST AND SAVE IT	20W1	548
COPY DES OF Z4	10Z4	549
	J74	550

IPL LIST

SAVE		60W3		551
ERASE REMAINS AFTER HEAD		G30		552
CHECK IF Z4 ON STRAT		12W4		553
GET NEXT Z AFTER INTERUPT		J81		554
		7091		555
SAVE NAME		60W2		556
CCPY OF Z4		11W3		557
PRCESS CES ATT.		10A3		558
CCMPARE		G210		559
NO,CONTINUE		7091		560
YES DELETE IT		12W4		561
FRCH STRAT		11W2		562
		J69		563
ERASE Z4		11W2		564
		J72		565
POT NWE Z4 ON STRAT	91	12W4		566
		11W3		567
		J64		568
STRAT LIST		11W1		569
POT DES CF CHANGE		11W3		570
ON STRAT		C3		571
SET REVERT DV		10P171		572
		20W2		573
GET		11W0		574
DESIGNATION		J81		575
CF		10A7		576
T-E		G10		577
CBJECT		10F1		578
CHOSEN		J2		579
		7093		580
SET REMOVE DV		10P101		581
		20W2	94	582
	93	10L9		583
SUBJECT		10A91		584
AWARE		G10		585
OF		10F7		586
DIMENSIONS VARIED		J2		587
NO, TO CREATE Z4, YES=CONTINUE		7094		588
HAS		10K98		589
CNE		10R1		590
DIMS CONC VARIED		J114		591
NO, CREATE NULL Z4, YES CREATE Z		70	94	592
ADD		11W3		593
C40		10C41		594
TO NULL Z4		J65	92	595
ADD P502 TO Z4	94	11W3		596
		10C41		597
		J65		598
MARK DIM VALS		11W3		599
		10P96		600

IPL LIST

		J65		601
INSERT		11W3		602
P91(MARK DIMENSIONS)		10P91		603
LN =Z4=		J65		604
ADD C41 TO Z4		11W3		605
		10P502		606
		J65		607
ADD P101 OR P171 TO Z		11W3		608
		11W2		609
	97	J65		610
EXIT	92	J0	J34	611

*****1 *****			612
C23 SET UP REACTION TO CONCEPT DES C23	90		613
	C22	0	614
	90	0	615
C22 SET UP =Z6=, REACT TO CONCEPT TO OBJECT DESIGNATION	C22	J44	616
CHECK IF Z4 ON STRAT		20W4	617
GET NEXT Z AFTER INTERRUPT		12W4	618
GO TO 91 IF NOTHING		J81	619
SAVE NAME		7091	620
CHECK IF Z6 THERE		60W2	621
PROCESS DES ATT.		10Z6	622
COMPARE		10A3	623
NO, CONTINUE		G210	624
YES DELETE IT		7091	625
FROM STRAT		12W4	626
		11W2	627
		J69	628
ERASE Z4		11W2	629
		J72	630
GET CONCEPT CHECK	91	10M1	631
		J81	632
		10A27	633
		G10	634
IF NONE ERROR IT IS A YES		7099	635
		10F1	636
		J2	637
		7092	J34
COPY DES OF Z6	92	10Z6	638
		J74	639
SAVE		0JW3	640
ERASE REMAINS AFTER HEAD		G30	641
PUT NME Z4 ON STRAT		12W4	642
		11W3	643
		J64	644
GET PHASE L		10C23	645
		10A6	646
		J10	647
PUT DES OF CHANGE ON STRAT		11W3	648
		C3	649
			650

IPL LIST

GET FOCUS OBJECT		11F6	651
OBJECT TEST		60W1	652
		10A5	653
IF ANY		R41	654
		J78	655
SET Z6 TO FIND UNMARKED DINS		10Q41	656
		70	93
SET Z6 TO FIND IRREL DINS		50Q42	657
	93	20W0	658
PUT C41	94	11W3	659
		10C41	660
CN =Z6=		J65	661
PUT C41		11W3	662
		10C41	663
CN=Z6=		J65	664
PUT C41		11W3	665
PUT Q41 OR Q42		11W0	666
CN =Z6=		J65	667
PUT P181		11W3	668
		10P181	669
CN =Z6=		J65	670
PUT P64		11W3	671
		10P64	672
ON Z6		J65	673
			674

IS		10L9		675
SUBJECT		10A91		675
AWARE		G10		677
		10F7		678
		J2		679
		70	96	680
FOCUS OBJECT		11W1		681
COUNT DIMENSIONALITY OF FOCUS		J126	97	682
	96	10K1		683
SET VALUE OF	97	J120		684
K96 ON		11W3		685
JW41.P181		J6		686
TO VALUE OF K		C8	J34	687
ERROR	95	10C23		688
		J152	J7	689

C8 SET DATA TERMS IN INPUTS OF	C8	J50		691
ROUTINES ON (1) TO VAL (0)		1091		692
GEN P ROUTINES		U6	J30	693
ENTER S FROM GENED CALL	91	52H0		694
GET INPUT PARAMS		10A1		695
		J10		696
NCNE EXIT		70J4		697
MOVE DOWN LIST OF INPUTS	95	J60		698
SAVE LOC		2097		699
EXIT IF END;		70J4		700

IPL LIST

CHECK IF DATA TERM		1297		701
		J131		702
NO CONTINUE		7093		703
YES REPLACE BY (0)		11W0		704
		2197		705
CONTINUE	93	1197	95	706
	57	0	0	707

C3 DESCRIBE CHANGE (TO)	C3	J50		709
TO Z, (1) STRAT		40H0		710
MAKE DDL WITH S AS DIM		10A9		711
AND A9		U20		712
DOUBLE NAME OF DDL		40H0		713
DESCRIBE Z ADDED AS (TO) CHANGE		11W0		714
		10A11		715
		J12		716
PUT DDL ON S		10A12		717
UNDER CLASS ATT		J12	J30	718

Z0 PROBLEM SPECIFICATIONS	ZC	9-J		720
COPY FOCUS		P21		721
REMEMBER CFO ON PROB LIST		P61		722
PUT COPY OF FOCUS ON MEP		C31		723
REMEMBER SET MEMBERSHIP OF CFO		P62	0	724
DESCRIPTION OF PROCESS	9C	0		725
GENERAL DESCRIPTION		A3		726
		93	0	727
	93	0		728
		94	0	729
	94	95	J	730
	95	0		731
TYPE OF PROCESS		A30		732
		96		733
COMMUNICATION		A31		734
		97		735
INFORMATION CREATED		A32		736
		98		737
DESIGNATION INFO		A33		738

		99	J	739
VALUE OF A30	96	0		740
DOING		F30	0	741
VALUE OF A31	97	J		742
FROM EXTERNAL		F41	0	743
VALUE OF A32	98	0		744
FOCUS OBJECT		F51	0	745
	99	0	0	746
*****1 *****				747
Z7 CREATE SEARCH CRITERION	Z7	90		748
CREATE WORKING HYP FROM CFO		P191		749
REMEMBER WH		P63		750

IPL LIST

PUT WH ON MEP		C31		751
REMEMBER WHAT WH MADE FROM		P64	J	752
DESCRIPTION OF PROCESS	90	0		753
GENERAL DESCRIPTION		A3		754
		93	J	755
	93	0		756
		94	0	757
	94	95	J	758
	95	0		759
TYPE OF PROCESS		A30		760
		96		761
COMMUNICATION		A31		762
		97		763
INFORMATION CREATED		A32		764
		98		765
DESIGNATION INFO		A33		766
		99	J	767
VALUE OF A30	96	0		768
DOING		F30	0	769
	97	0	0	770
VALUE OF A32	98	J		771
FOCUS OBJECT		F51	0	772
	99	0	0	773
*****1 *****				774
D4 PROCEED FROM PHASE 2 FROM 3NE	D4	90		775
		D40	0	776
	90	J		777
		A3		778
		93	0	779
	93	J		780
		94	0	781
	94	95	0	782
	95	0		783
		A30		784
		96		785
		A31		786
		97		787
		A32		788
		98		789
		A33		790
		99	0	791
	96	0		792
		F31	J	793
	97	0	0	794
	98	0		795
		F53	J	796
	99	0	0	797
D40	D40	J4	0	798
*****1 *****				799
Z1 FORM OBJECT SELECTION CRIT	Z1	90		800



IPL LIST

SELECT DIMS TO VARY		P131		801
VARY DIMS SELECTED		P141		802
MODIFY MH INTO SEARCH CRITERION		P151		803
REMEMBER HOW MH MODIFIED		P64	0	804
DESCRIPTION OF PROCESS	9C	0		805
GENERAL DESCRIPTION		A3		806
		93	0	807
	93	0		808
		94	0	809
	94	95	0	810
	95	0		811
TYPE OF PROCESS		A30		812
		96		813
COMMUNICATION		A31		814
		97		815
INFORMATION CREATED		A32		816
		98		817
DESIGNATION INFO		A33		818
		99	0	819
VALUE OF A30	56	0		820
DOING		F30	0	821
	57	0	0	822
VALUE OF A32	98	0		823
SEARCH CRITERION		F52	0	824
	95	0	0	825
*****				826
22 SELECT AN OBJECT	22	9-0		827
THAT AGREES WITH FOCUS AND		P51		828
REMEMBER OBJECT FOUND		P65		829
PUT OBJ CN REP		C31		830
REMEMBER HOW OBJ FOUND		P66	0	831
DESCRIPTION OF PROCESS	9C	0		832
GENERAL DESCRIPTION		A3		833
		93	0	834
	93	0		835
		94	0	836
	94	95	0	837
	55	0		838
TYPE OF PROCESS		A30		839
		96		840
COMMUNICATION		A31		841
		97		842
INFORMATION CREATED		A32		843
		98		844
DESIGNATION INFO		A33		845
		99	0	846
VALUE OF A30	56	0		847
DOING		F30	0	848
VALUE OF A31	97	0		849
FROM EXTERNAL		F41	0	850

IPL LIST

VALUE OF A32	98	0		851
OBJECT CHOICE		F53	0	852
	99	0	0	853
*****				854
DO VERIFY OBJ CHOICE	9C	90		855
		Q51	0	856
	9C	0		857
		A3		858
		93	0	859
	93	0		860
		94	0	861

	94	95	0	862
	95	0		863
		A30		864
		96		865
		A31		866
		97		867
		A32		868
		98		869
		A33		870
		99	0	871
	96	0		872
		F31	0	873
	97	0	0	874
	98	0		875
		F51	0	876
	99	0	0	877
Q51 VERIFY OBJ CHOICE	Q51	90		878
		Q50	0	879
	9C	0		880
		A1		881
		91		882
		A2		883
		92	0	884
	91	0		885
		H10	0	886
	92	0		887
		H5	0	888
Q50 CHECK OBJECTS CHOICE	Q50	J6		889
		J51		890
GET OBJ		11M1		891
GET OBJ		J81		892
		60M1		893
GET FOCUS		11F6		894
F-0		R3		895
		20M0		896
GET LIST		11M1		897
		10A26		898
		G10		899
		70J31		900

IPL LIST

(TO) DES		10A11		901
		G10		902
		11M0		903
		7092		904
FOCUS-0)-(TO LIST)		R3		905
		11M0		906
		J71		907
CHECK IF VALS ON		40M0		908
(F-0)-(TO) ARE MARKED		1091		909
		J100		910
	92	J4		911
ERASE		J71	J31	912
CHECK IF VALS MARKED	91	10A5		913
		R81		914
		700	J8	915
*****				916
Z3 HAVE EXPERIMENTER DESIGNATE IT	Z3	9-0		917
TRANSFER OBJ NAME FROM REP TO E1		P71		918
EXPERIMENTOR DESIGNATES OBJ		E93		919
REMEMBER DESIG. (SET)		P62	0	920
DESCRIPTION OF PROCESS	9C	0		921
GENERAL DESCRIPTION		A3		922
		93	0	923
	93	0		924

		94	0	925
	94	95	0	926
	95	0		927
TYPE OF PROCESS		A30		928
		96		929
COMMUNICATION		A31		930
		97		931
INFORMATION CREATED		A32		932
		98		933
DESIGNATION INFO		A33		934
		99	0	935
VALUE OF A30	96	0		936
DOING		F30	0	937
VALUE OF A31	97	0		938
BOTH		F42	0	939
VALUE OF A32	98	0		940
OBJECT CHOICE		F53	0	941
VALUE OF A33	95	0		942
OBJECT		F71	0	943
*****I ****				944
Z4 PROCESS OBJ DESIG. INFO.	Z4	90		945
RECALL SET MEMBERSHIP OF OBJ		P501		946
POP MEP		C41		947
FLAG DIM AND VALUES		P91		948
REVERT DIM VALS ON WH		P171		949
REMEMBER CHANGE IN S.C.		P64	0	950

IPL LIST

DESCRIPTION OF PROCESS	90	0		951
GENERAL DESCRIPTION		A3		952
		93	0	953
	93	0		954
		94	0	955
	94	95	0	956
	95	0		957
TYPE OF PROCESS		A30		958
		96		959
COMMUNICATION		A31		960
		97		961
INFORMATION CREATED		A32		962
		98		963
DESIGNATION INFO		A33		964
		99	0	965
VALUE OF A30	96	0		966
DOING		F30	0	967
	97	0	0	968
	98	0	0	969
VALUE OF A33	95	0		970
OBJECT		F71	0	971
*****I ****				972
D1 CAN A CONCEPT BE PRESENTED	D1	90		973
		0101	0	974
DESCRIPTION OF PROCESS	90	0		975
GENERAL DESCRIPTION		A3		976
		93	0	977
	93	0		978
		94	0	979
	94	95	0	980
	95	0		981
TYPE OF PROCESS		A30		982
		96		983
COMMUNICATION		A31		984
		97		985
INFORMATION CREATED		A32		986
		98		987
DESIGNATION INFO		A33		988

		99	0	989
VALUE OF A30	56	0		990
DECIDING		F31	0	991
	97	J	J	992
VALUE OF A32	98	0		993
CONCEPT		F54	0	994
	95	J	J	995
*****1 *****				
Z5 FORM CONCEPT, HAVE EXPERIMENTER	Z5	9-0		996
FORM CONCEPT		P121		997
REMEMBER CONCEPT		P67		998
PUT CONCEPT ON MEP		C31		1000

IPL LIST

REMEMBER HOW CONCEPT FORMED		P68	0	1001
DESCRIPTION OF PROCESS	90	0		1002
GENERAL DESCRIPTION		A3		1003
		93	0	1004
	93	0		1005
		94	0	1006
	94	95	0	1007
	95	J		1008
TYPE OF PROCESS		A30		1009
		96		1010
COMMUNICATION		A31		1011
		97		1012
INFORMATION CREATED		A32		1013
		98		1014
DESIGNATION INFO		A33		1015
		99	0	1016
VALUE OF A30	96	0		1017
DOING		F30	0	1018
	97	0	0	1019
VALUE OF A32	98	0		1020
CONCEPT		F54	0	1021
	95	0	0	1022
*****1 *****				
Z8 HAVE E DESIGNATE CONCEPT	Z8	90		1023
NAME OF CONCEPT TO E		P72		1024
E DESIGNATE CONCEPT		E94		1025
REMEMBER DESIGNATION OF CONCEPT		P69	0	1026
DESCRIPTION OF PROCESS	90	0		1027
GENERAL DESCRIPTION		A3		1028
		93	J	1029
	93	0		1030
		94	0	1031
	94	95	J	1032
	95	0		1033
TYPE OF PROCESS		A30		1034
		96		1035
COMMUNICATION		A31		1036
		97		1037
INFORMATION CREATED		A32		1038
		98		1039
DESIGNATION INFO		A33		1040
		99	0	1041
VALUE OF A30	96	0		1042
DOING		F30	0	1043
VALUE OF A31	97	0		1044
BOTH		F42	0	1045
VALUE OF A32	98	0		1046
CONCEPT		F54	0	1047
VALUE OF A33	95	0		1048
CONCEPT		F72	0	1049

IPL LIST

*****1 *****				1051
D2 DETERMINE CONCEPT DESIGNATION	D2	90		1052
		P501		1053
DETERMINES IF CONCEPT CORRECT		Q12	0	1054
DESCRIPTION OF PROCESS	9C	0		1055
GENERAL DESCRIPTION		A3		1056
		93	0	1057
	93	0		1058
		94	0	1059
	94	95	0	1060
	95	0		1061
TYPE OF PROCESS.		A30		1062
		96		1063
COMMUNICATION		A31		1064
		97		1065
INFORMATION CREATED		A32		1066
		98		1067
DESIGNATION INFO		A33		1068
		99	0	1069
VALUE OF A30	56	0		1070
DECIDING		F31	0	1071
	97	0	0	1072
VALUE OF A32	98	0		1073
CONCEPT		F54	0	1074
VALUE OF A33	99	0		1075
CONCEPT		F72	0	1076
*****1 *****				1077
Z6 CORRECTIVE ACTION	Z6	90		1078
POP MEP TO WH		C41		1079
		C41		1080
FIND UNTESTED DIM VAL(S)		Q41		1081
		91		1082
SCME		92	0	1083
MCNE EXIT	91	R0	0	1084
BR (+)	92	0		1085
ADD DIMS TO WH		P181		1086
PUT NEW WH ON MEP		C31		1087
REMEMBER HOW NEW WH FORMED		P64	0	1088
DESCRIPTION OF PROCESS	9C	0		1089
GENERAL DESCRIPTION		A3		1090
		93	0	1091
	93	0		1092
		94	0	1093
	94	95	0	1094
	95	0		1095
TYPE OF PROCESS		A30		1096
		96		1097
COMMUNICATION		A31		1098
		97		1099
INFORMATION CREATED		A32		1100

IPL LIST

		98		1101
DESIGNATION INFO		A33		1102
		99	0	1103
VALUE OF A30	56	0		1104
COING		F30	0	1105
	57	0	0	1106
VALUE OF A32	98	0		1107
SEARCH CRITERION		F52	0	1108
	99	0	0	1109
*****1 *****				1110
D3 GIVE UP	D3	90		1111

		J0	0	1112
DESCRIPTION OF PROCESS	90	0		1113
GENERAL DESCRIPTION		A3		1114
		93	0	1115
	93	0		1116
		94	0	1117
	94	95	0	1118
	95	0		1119
TYPE OF PROCESS		A30		1120
		96		1121
COMMUNICATION		A31		1122
		97		1123
INFORMATION CREATED		A32		1124
		98		1125
DESIGNATION INFO		A33		1126
		79	0	1127
VALUE OF A30	96	0		1128
DECIDING		F31	0	1129
	97	0	0	1130
VALUE OF A32	98	0	0	1131
VALUE OF A33	95	0	0	1132
*****1 *****				1133
EXIT (DO NOTHING)	90	90		1134
		J0	0	1135
	90	0	0	1136
*****1 *****				1137
E95 PRESENT FOCUS OBJ TO SUBJECT	E95	90		1138
		E2	0	1139
	90	0		1140
		A1		1141
		91		1142
		A2		1143
		92	0	1144
	91	0		1145
CONTAINS NAME OF FOCUS OBJ		F6	0	1146
	92	0		1147
CONTAINS NAME AND DESIG OF F OBJ		M1	0	1148
E2 EXPERIMENTER PUT FOCUS	E2	J51		1149
CLEAR		11W1		1150

IPL LIST

M1		G30		1151
AND DESCRIPTION ON STM (M1)		11W1		1152
DESCRIBE FOCUS AS YES		10F1		1153
		J91		1154
		J136		1155
ATTACH DESCRIPTION		10A7		1156
MAKE DESCRIPTION		U20		1157
PUT DESCRIPTION ON M1 STM		G2		1158
ENTER STM (M1)		11W1		1159
ENTER FOCUS		12W0		1160
PUT OBJECT ON STM		G2	J31	1161
*****1 *****				1162
P21 COPY AND PUTBACK THE 1ST ITEM	P21	9-0		1163
		P20	0	1164
	9-0	0		1165
		A2		1166
		9-2		1167
		A1		1168
		9-1	0	1169
	9-1	0		1170
		M1	0	1171
	9-2	0		1172
		M1	0	1173

P20 MAKE COPY OF LIST STRUCTURE	P20	J51		1174
IN (0) AND REPLACE TOP	9-2	11W1		1175
PUT NAME OF COPY BACK IN M1		11W0		1176
GET MPP		G1		1177
SAVE NAME		40H0		1178
GET DIM VAL		G1		1179
GET DIM		10A6		1180
		G10		1181
GET M13		10A6		1182
		G10		1183
(0)FOCUS,(1)M13		J6		1184
MAKE CFO IN DAY ORDER		R190		1185
MAKE LOCAL		J136		1186
		G4	J31	1187
*****1 *****				1188
P191 CREATE INITIAL HYPOTHESIS	P191	90		1189
		P190	0	1190
	9-0	0		1191
		A2		1192
		9-2		1193
		A1		1194
		9-1	0	1195
	9-1	0		1196
		M10		1197
		K99	0	1198
	9-2	0		1199
		M1	0	1200

IPL LIST

P190 MAKE K99 DIM ED S.C.	P190	J50		1201
(0)K99,(1)M10,(2)M1		G1		1202
GET CFO COPY SAVE		R73		1203
		6097		1204
FIND K99TH ELEMENT ON CFO		11W0		1205
		J200		1206
CUT LIST		J75		1207
		J136		1208
DOUBLE NAME OF REMAINS		40H0		1209
COLLECT DIMS OF REMAINS		R6		1210
MAKE FOM TO D-LIST		10A9		1211
WITH DIMS VARIED		U20		1212
SAVE DDL		60W0		1213
ATTACH		J6		1214
DIM VALS FROM		10A10		1215
TO DDL		J11		1216
CLEAR M1		40H0		1217
		G30		1218
PUT DDL		40H0		1219
INM1		11W0		1220
		G2		1221
PUT S.C.		1197		1222
IN M1 EXIT		G2	J30	1223
	97	0	0	1224
*****1 *****				1225
P131 SELECT A DIM TO VARY	P131	9-0		1226
		P130	0	1227
	9-0	0		1228
		A2		1229
		9-2		1230
		A1		1231
		9-1	0	1232
	9-1	0		1233
		M10		1234
		K98	0	1235
	9-2	0		1236
		M1	0	1237

P130 SELECT DI TO VARY	P130	J44	1238
(0)NO. TC VARY,(1)MEP,(2)M1		20W0	1239
GET S.C.		G1	1240
COLLECT DIM VALS		60W3	1241
NOT DESIGN ED		10A5	1242
		R41	1243
SAVE LIST		60W1	1244
CUT OFF LIST		11W0	1245
AFTER IWO		J200	1246
ERASE REMAINS		G30	1247
		11W1	1248
COLLECT DIMS		R6	1249
MAKE DDL		10A9	1250

IPL LIST

OF DIMS (TO VARY)		U20	1251
		60W2	1252
ACC DIM VALS LIST		11W1	1253
TO DDL		10A10	1254
		J11	1255
CLEAR M1		40H0	1256
		G30	1257
PUT DDL IN M1		40H0	1258
		11W2	1259
		G2	1260
PUT S.C. IN M1		11W3	1261
		G2 J34	1262
*****1 *****			1263
P141 VARY SELECTED DIMENSION	P141	9-0	1264
		P140 0	1265
	9-0	0	1266
		A2	1267
		9-2	1268
		A1	1269
		9-1 J	1270
	9-1	0	1271
		M1 0	1272
	9-2	J	1273
		M1 0	1274
P140 GET (CHANGE TO DIM VALS)	P140	J45	1275
(0) M1,(1)M1		20W5	1276
GET DDL		J82	1277
SAVE NAME		60W4	1278
GET (CHANGE FROM) DIM VALUES		10A9	1279
		J10	1280
		70J45	1281
GET DIMS VARIED		11W4	1282
		10A10	1283
		J10	1284
GENERATE IN PARALLEL		1091	1285
		R100 J35	1286
SAVE DIM AND DIM VALS	91	J51	1287
DIM		11W1	1288
GET PREF ORDER LIST		10A8	1289
		G10	1290
SAVE		60W1	1291
SET PROB OF FOCUS DIM VAL -		11W0	1292
		G9	1293
CHOOSE DIM VAL	93	11W1	1294
		R52	1295
NCNE EXIT		70J31	1296
SAVE VAL		60W2	1297
CHECK IF LIST		10A5	1298
		G10	1299
NO		7092	1300

IPL LIST

YES PICK ANOTHER		30H0	93	1301
ACC (CHANGE TO) ATT	92	11W4		1302
AND VAL TO DOL		11W2		1303
		10A11		1304
		J12		1305
SET SIGN		11W1		1306
(TO) VAL +		11W2		1307
		J13		1308
		J122		1309
		30H0		1310
SET SIGN (FROM)		11W1		1311
PRUB +		11W0		1312
		J10		1313
		J122		1314
		30H0		1315
		J4	J31	1316
*****1 *****				1317
P151 VARY THE HYPOTHESIS	P151	9-0		1318
		P150	0	1319
	9-0	0		1320
		A2		1321
		9-2		1322
		A1		1323
		9-1	0	1324
	9-1	0		1325
		M1	0	1326
	9-2	0		1327
		M1	0	1328
P150 VARY SEARCH CRITERION	P150	J42		1329
		20W0		1330
FRM A10 VALUE TO A11 VALUE		G1		1331
INPUT (0) M1		20W2		1332
SAVE S.C. GET DESCRIPT		11W0		1333
		J82		1334
SAVE		60W1		1335
GET CHANGE		10A10		1336
FRM LIST		J10		1337
ERROR		70J32		1338
GET (CHANGE TO) LIST		11W1		1339
		10A11		1340
		J10		1341
GENERATE BOTH IN PARALLEL		1091		1342
		R100		1343
		J4	J32	1344
CHANGE SEARCH CRITERION	91	J51		1345
REPLACE CHANGE FROM		11W2		1346
DIM VAL BY CHANGE TO		11W1		1347
		11W0		1348
POP		J31		1349
		J67	J4	1350

IPL LIST

*****1 *****				1351
P51 LOCATE OBJ. VIA SEL. CRITERIA	P51	9-0		1352
		P50	0	1353
	9-0	0		1354
		A2		1355
		9-2		1356
		A1		1357
		9-1	0	1358
	9-1	0		1359
		M10		1360
		E0	0	1361

	9-2	0		1362
		M1	0	1363
P50 SEARCH FOR OBJ MATCHING HYP	P50	J52		1364
GET CURRENT HYP		11W1		1365
		G1		1366
SAVE NAME		20W1		1367
(0)EO,(1)HO,(2)MI	91	11W0		1368
HYP		11W1		1369
SEARCH BOARD FOR OBJECT		R2		1370
MATCHING CURRENT HYP		70J32		1371
SAVE OBJECT NAME		20W0		1372
CLEAR NZ1		11W2		1373
		G30		1374
ENTER HYP NAME		11W1		1375
PUT S.C.		10A12		1376
FRCH TO ATTS.		G10		1377
ON OBJECT CHOICE		J74		1378
		J136		1379
PLACE OBJECT		11W2		1380
DESCRIPTION		J6		1381
ON SYN (1W2)		G2		1382
PLACE OBJECT		11W2		1383
LCC		12W0		1384
COPY OBJ.		R73		1385
ON SYN.EXIT		G2	J32	1386
*****1 *****				1387
P61 ATTACH CFO TO PROB LIST	P61	90		1388
		P60	0	1389
	9-0	0		1390
		A2		1391
		9-2		1392
		A1		1393
		9-1	0	1394
	9-1	0		1395
		M1		1396
		F3	0	1397
	9-2	0		1398
		M10		1399
		A20	0	1400

IPL LIST

P62 ATTACH SET MEM-SHIP TO CFO	P62	90		1401
		P60	0	1402
	9-0	0		1403
		A2		1404
		9-2		1405
		A1		1406
		9-1	0	1407
	9-1	0		1408
		M1		1409
		F4	0	1410
	9-2	0		1411
		M10		1412
		A15	0	1413
P63 REMEMBER WORKING HYP	P63	90		1414
		P60	0	1415
	9-0	0		1416
		A2		1417
		9-2		1418
		A1		1419
		9-1	0	1420
	9-1	0		1421
		M1		1422
		F3	0	1423
	9-2	0		1424
		M10		1425

		A18	0	1426
P64 REMEMBER HOW NH FORMED	P64	90		1427
		P65	0	1428
	9-0	0		1429
		A2		1430
		9-2		1431
		A1		1432
		9-1	0	1433
	9-1	J		1434
		M1		1435
		F4	0	1436
	9-2	0		1437
		M10		1438
		A12	0	1439
P65 REMEMBER OBJECT FOUND	P65	90		1440
		P60	0	1441
	9-0	0		1442
		A2		1443
		9-2		1444
		A1		1445
		9-1	0	1446
	9-1	0		1447
		M1		1448
		F3	0	1449
	9-2	J		1450

IPL LIST

		M10		1451
P66 REMEMBER HOW OBJ FOUND	P66	A16	0	1452
		90		1453
		P60	0	1454
	9-0	0		1455
		A2		1456
		9-2		1457
		A1		1458
		9-1	0	1459
	9-1	0		1460
		M1		1461
		F4	0	1462
	9-2	0		1463
		M10		1464
		A26	0	1465
P67 REMEMBER CONCEPT	P67	90		1466
		P60	0	1467
	9-0	J		1468
		A2		1469
		9-2		1470
		A1		1471
		9-1	0	1472
	9-1	0		1473
		M1		1474
		F3	0	1475
	9-2	0		1476
		M10		1477
		A17	0	1478
P68 REMEMBER HOW CONCEPT FORMED	P68	90		1479
		P60	0	1480
	9-0	0		1481
		A2		1482
		9-2		1483
		A1		1484
		9-1	0	1485
	9-1	0		1486
		M1		1487
		F4	0	1488
	9-2	0		1489

		M10		1490
		A19	0	1491
P69 REMEMBER DESIGNATION OF CONCEP	P69	90		1492
		P50	0	1493
	9-0	0		1494
		A2		1495
		9-2		1496
		A1		1497
		9-1	0	1498
	9-1	0		1499
		M1		1500

IPL LIST

		F4	0	1501
	9-2	0		1502
		M10		1503
		A15	0	1504
P60 ATTACH (M1,X) TO V-LIST	P60	J52		1505
OF CLASS ATT (2) OF (3)		J81		1506
(0) FX, (1)M1, (2)AFT, (3)M10		11M1		1507
		11W0		1508
		G12		1509
		11W2		1510
		J12	J32	1511
*****				1512
C31 PUT COPY OF FOCUS AT TOP MEP	C31	90		1513
		P70	0	1514
	9-0	0		1515
		A2		1516
		9-2		1517
		A1		1518
		9-1	0	1519
	9-1	0		1520
		M1	0	1521
	9-2	0		1522
		M10	0	1523
*****				1524
P71 NAME OF OBJECT TO E1	P71	90		1525
		P70	0	1526
	9-0	0		1527
		A2		1528
		9-2		1529
		A1		1530
		9-1	0	1531
	9-1	0		1532
		M10	0	1533
	9-2	0		1534
		E1	0	1535
P72 NAME OF CONCEPT TO EXPERIMENTE	P72	90		1536
		P70	0	1537
	9-0	0		1538
		A2		1539
		9-2		1540
		A1		1541
		9-1	0	1542
	9-1	0		1543
		M10	0	1544
	9-2	0		1545
		E1	0	1546
P70PUT FIRST SYMB FROM MEMORY	P70	J51		1547
LIST ON A MEMORY LIST (OR E1)		11W1		1548
INPUTS., (0) MEN.LIST,(1) E1		11W0		1549
GET OBJECT		G1		1550

IPL LIST

		G2	J31	
PUT OBJECT ON MEMORY LIST				1551
*****				1552
P501 RECALL SET MEMBERSHIP	P501	90		1553
		P500	J	1554
	9-0	0		1555
		A2		1556
		9-2		1557
		A1		1558
		9-1	0	1559
	9-1	0		1560
		M10		1561
		A15	0	1562
	9-2	0		1563
		M1	0	1564
P502 RECALL CONCEPT DESIGNATION	P502	90		1565
		P500	0	1566
	9-0	0		1567
		A2		1568
		9-2		1569
		A1		1570
		9-1	0	1571
	9-1	0		1572
		M10		1573
		A12	0	1574
	9-2	0		1575
		M1	0	1576
P500 (M1,N)=MEP,(M1,D)=POT SYMB	P500	J51		1577
OF V-LIST OF ATT (0) OF MEP		40H0		1578
(0) CLASS ATT (0), (1)MEP,(2)M1		G30		1579
		40H0		1580
		11W1		1581
		J81		1582
		11W0		1583
		G10		1584
		70J31		1585
		G2		1586
		11W1		1587
		J81		1588
		G2	J31	1589
*****				1590
E93 DESIGNATE OBJECT	E93	9-0		1591
		E3	J	1592
	9-0	0		1593
		A2		1594
		9-2		1595
		A1		1596
		9-1	0	1597
	9-1	0		1598
		E1	J	1599
	9-2	0		1600

IPL LIST

		M1	0	1601
E3 DESIGNATE OBJECT	E3	J51		1602
CLEAR		11W1		1603
M1		G30		1604
(0) E1, (1) M1		11W0		1605
GET OBJECT NAME FROM E1		G1		1606
DOUBLE NAME IN H0		40H0		1607
ENTER CORRECT CONCEPT NAME		11F5		1608
TEST IF CONCEPT IN OBJECT		R8		1609
ENTER M1		11W1		1610
GET RESULT OF TEST		G8		1611



		J91		1612
		J136		1613
CONSTRUCT		10A7		1614
STM DESCRIPTION, PLACE		U20		1615
DESCRIPTION ON STM		G2		1616
PLACE OBJECT ON STM		11W1		1617
		J6		1618
		G2	J31	1619
*****				1620
C41 POP UP-MEP 1 LEVEL	C41	90		1621
		P400	0	1622
	9-0	0		1623
		A1		1624
		9-1	0	1625
	9-1	0		1626
		M10	0	1627
P400 POP LIST	P400	31H0	J8	1628
*****				1629
P91 MAKE CONCLUSIONS ON DIM VALS	P56	90		1630
		P95	0	1631
	9-0	0		1632
		A2		1633
		9-2		1634
		A1		1635
		9-1	0	1636
	9-1	0		1637
		A7		1638
		M10		1639
		A12		1640
		M1		1641
		K96	0	1642
	9-2	0		1643
		M1	0	1644
P95 MARK DIM VALS REL OR IRREL	1			1645
(0)K96, (1)M1, (2)A12, (3)MEP, (4) M1				1646
SAVE INPUTS	P55	J54		1647
MAKE 99=0		5099		1648
		J124		1649
		30H0		1650

IPL LIST

GET OBJECT D		11W1		1651
		J82		1652
GET E DESIG		11W4		1653
		G10		1654
SAVE		2097		1655
GET WH		11W3		1656
		J81		1657
SAVE		6096		1658
GET DIM VARIED D		11W2		1659
		G10		1660
SAVE		6098		1661
LIST		10A10		1662
		J10		1663
GENERATE BOTH		1091		1664
IN PARALLEL		J100	J34	1665
DESCRIBE DIM AS REL, IRREL	91	1197		1666
		10A5		1667
		J12		1668
CHECK IF DONE		1099		1669
		J125		1670
		11W3		1671
		J114	J5	1672
	97	0	0	1673
	98	0	0	1674
	99	+01	0	1675



*****				1676
*****				1677
P91 MAKE CONCLUSIONS ON DIM VALS	PS1	90		1678
		P90	0	1679
	9-0	0		1680
		A2		1681
		9-2		1682
		A1		1683
		9-1	0	1684
	9-1	0		1685
		A7		1686
		M10		1687
		A12		1688
		M1		1689
		K96	0	1690
	9-2	0		1691
		M1	0	1692
P90 MARK DIMS AND DIM VALS REL OR 1 REL				1693
(0)K96,(1)M1,(2)A12,(3)M1,(4) M1				1694
SAVE INPUTS	P50	J54		1695
MAKE 99=0		5099		1696
		J124		1697
		30H0		1698
GET OBJECT D		11W1		1699
		J82		1700
GET E DESIG		11W4		

IPL LIST

		G10		1701
SAVE		2097		1702
GET WH		11W3		1703
		J81		1704
SAVE		6096		1705
GET DIM VARIED D		11W2		1706
		G10		1707
SAVE		6098		1708
GET DIM LIST CHANGED		10A9		1709
		J10		1710
GENERATE BOTH		1091		1711
		J10J		1712
CLEAR MUI		11W1		1713
		G30		1714
PUT DIM VARIED DES		11W1		1715
IN M1		1198		1716
		G2		1717
PUT WH IN M1		11W1		1718
		1196		1719
EXIT		G2	J34	1720
91 SAVE DIM	91	60W4		1721
CHECK ALL VALS MARKED		10A5		1722
		R41		1723
		40H0		1724
CHECK IF ALL BUT 1 MARKED		J6J		1725
		J78		1726
ERASE OUTPUT		J71		1727
NO EXIT		JJ	0	1728
YES MARK DIM		11W4		1729
		1197		1730
		10A5		1731
		J12		1732
CHECK IF DUNE		1099		1733
		J125		1734
		11W0		1735
		J114	J5	1736
	97	0	0	1737

	98	0	0	1738
	99	+01	0	1739

P101 REMOVE IRREL DV	P101	90		1741
		P100	0	1742
	9C	0		1743
		A2		1744
		92		1745
		A1		1746
		91	0	1747
	91	0		1748
		M1	0	1749
	92	J		1750

IPL LIST

		M1	0	1751
P100 DELETE IRREL DV	P100	J42		1752
GET NH		J81		1753
		Z0W2		1754
GET ODL		J82		1755
		60W1		1756
GET CHANGE FROM LIST		10A10		1757
		J10		1758
		70J32		1759
		11W1		1760
GET TO L		10A11		1761
		J10		1762
EXIT IF WRONG INFO		70J32		1763
GEN		1091		1764
		R100		1765
EXCHANGE FROM RTD		11W1		1766
		40H0		1767
		40H0		1768
		10A10		1769
		J10		1770
		J74		1771
		11W1		1772
		10A11		1773
		J10		1774
		J74		1775
		10A10		1776
		J11		1777
		10A11		1778
		J11	J32	1779
SAVE DV (TO)	91	Z0W0		1780
CHECK REL,DIRREL		10A5		1781
		G10		1782
		70J4		1783
IRREL		10F1		1784
		J2		1785
YES EXIT		70J4		1786
IRREL,REMOVE FROM NH		11W2		1787
		11W0		1788
		J69	J4	1789

P171 REVERT HYP BACK TO ORIGINAL	P171	9-0		1791
		P170	0	1792
	9-0	0		1793
		A2		1794
		9-2		1795
		A1		1796
		9-1	0	1797
	9-1	0		1798
		M1	0	1799
	9-2	0		1800

UN

IPL LIST

		M1	0	1801
		J42		1802
PI70 REVERT S.C. TO ORIGINAL STATE	PI70	J81		1803
GET		20W2		1804
S.C.		60W0		1805
SAVE NAME M1		J82		1806
GET S.C. FROM DES		60M1		1807
GET LIST IOF CHANGED VALUES		10A11		1808
		J10		1809
		70J32		1810
SAVE (TO) VAL LIST		6097		1811
CHANGE FROM LIST		11W1		1812
		10A13		1813
		J10		1814
SAVE (FRM) VAL LIST		6098		1815
GENERATE BOTH		1091		1816
IN PARALLEL		R100		1817
		11W1		1818
COPY DDL		J74		1819
		60W1		1820
EXCHANGE (FRM TO) VAL		1197		1821
COPY FROM LIST		J74		1822
		10A13		1823
		J11		1824
		11W1		1825
		1198		1826
COPY (TO) LIST		J74		1827
		10A11		1828
		J11		1829
CLEAR M1		11W0		1830
		G30		1831
PUT DDL IN M1		11W0		1832
		11W1		1833
		G2		1834
PUT S.C. IN M1		11W0		1835
		11W2		1836
		G2	J32	1837
SUB PROCESS	91	J51		1838
REPLACE (TO VALUE)		11W2		1839
BY (FROM VALUE)		11W1		1840
ON SEARCH CRITERION		11W0		1841
		J67		1842
		J4	J31	1843
				1844
*****1 *****				1844
Q101 CAN CCNCEPT BE GIVEN	Q101	90		1845
		Q100	0	1846
	9-0	0		1847
		A2		1848
		9-2		1849
		A1		1850

IPL LIST

		9-1	J	1851
	9-1	0		1852
		M10	0	1853
	9-2	J		1854
		H5	0	1855
Q100 CHECK IF ALL DIM VALS MARKED	Q100	J81		1856
CHECK IF ALL DIM VALS MARKED		1091		1857
		J100	J8	1858
	91	10A5		1859
		G10		1860
		700	J8	1861

*****				1862
P121 FORM CONCEPT	P121	9-0		1863
		P120	0	1864
	9-0	0		1865
		A2		1866
		9-2		1867
		A1		1868
		9-1	0	1869
	9-1	0		1870
		M10	0	1871
	9-2	0		1872
		M1	0	1873
P120 COLLECT CONCEPT (0)REP,(1)M1	P120	J46		1874
GET S.C.		J81		1875
SAVE		20W6		1876
SAVE M1		20W5		1877
CONCEPT NAME		J90		1878
		J136		1879
		20W3		1880
DDL NODE		J90		1881
		J136		1882
SAVE		20W2		1883
GENERATE S.C.		11W6		1884
		1091		1885
		J100		1886
COPY S.C.		11W6		1887
		X73		1888
		20W6		1889
ATTACH COPY		11W2		1890
TC DDL UNDER BASIS ATT		11W6		1891
		10A31		1892
		J12		1893
CLEAR M1		11W5		1894
		G30		1895
PUT DDL IN M1		11W5		1896
		11W2		1897
		G2		1898
PUT CONCEPT IN M1		11W5		1899
		11W3		1900

IPL LIST

EXIT		G2	J36	1901
SUB P SAVE DIM VALUE	91	60W4		1902
CHECK IF REL,IRREL		10A5		1903
		G10		1904
NEITHER,EXIT		70J4		1905
		20W1		1906
PUT DIM VALUE		11W2		1907
ON DDL UNDER		11W4		1908
REL OR IRREL ATT		11W1		1909
		J12		1910
CHECK IF DIM VAL REL		11W1		1911
(REL) SYMB)		10F2		1912
		J2		1913
NO,EXIT		70J4		1914
YES ADD TO		11W3		1915
CONCEPT LIST		11W4		1916
		J66	J4	1917
*****				1918
E94 DESIGNATE CONCEPT	E94	9-0		1919
		E4	0	1920
	9-0	0		1921
		A2		1922
		9-2		1923
		A1		1924
		9-1	0	1925

	9-1	0		1926
		E1	0	1927
	9-2	0		1928
		M1	0	1929
E4 EXPERIMENTER DESIGNATE CONCEPT	E4	J51		1930
CLEAR		11W1		1931
M1		G30		1932
(0) E1, (1) M1		11W0		1933
GET SUBJECTS CONCEPT		G1		1934
SAVE NAME		60W0		1935
PUT CONCEPT DESIGNATION ON		11F5		1936
M1 COMPARE CONCEPT WITH		R8		1937
CORRECT CONCEPT		709-1		1938
COMPARE CORRECT CONCEPT WITH		11F5		1939
SUBJECTS CONCEPT		11W0		1940
		R8		1941
BUILD DETACHED DES.	91	G8		1942
		J91		1943
		J136		1944
SAVE H5		40H5		1945
CONCEPT DESIGNATION ATT.		1JA27		1946
		U20		1947
PUT DESCRIPTION		11W1		1948
ON STM		J6		1949
		G2		1950

IPL LIST

PUT SUBJECTS		11W1		1951
CONCEPT CN		11W0		1952
STM		G2		1953
		30H5	J31	1954
*****1 *****				1955
Q12 DETERMINE IF CONCEPT CORRECT	Q12	90		1956
		Q10	0	1957
	9-0	0		1958
		A2		1959
		9-2		1960
		A1		1961
		9-1	0	1962
	9-1	0		1963
		M1		1964
		A27		1965
		F1	0	1966
	9-2	0		1967
		H5	0	1968
Q10 CHECK IF CONCEPT CORRECT	Q10	J51		1969
GET DES.		J82		1970
ENTER ATT		11W1		1971
		G10		1972
COMPARE		11W0		1973
EXIT		J2		1974
		30H0	J31	1975
*****1 *****				1976
P181 ADD DIMENSION TO SEARCH C.	P181	90		1977
		P180	0	1978
	9-0	0		1979
		A2		1980
		9-2		1981
		A1		1982
		9-1	0	1983
	9-1	J		1984
		M1	0	1985
	9-2	0		1986
		M1	0	1987

P180 ADD UNTESTED DV TO WH	P180	J47	1988
SAVE M1		20W7	1989
GET FOCUS COPY		J81	1990
GET WH OFF DES		10A18	1991
		G10	1992
SAVE NAME		60W6	1993
PUT WH IN M1.N		G4	1994
GET DDL		11W7	1995
		J82	1996
GET (TO) LIST		10A11	1997
		J10	1998
GENERATE		1091	1999
		J100 J37	2000

IPL LIST

ADD (TO) LIST VALS TO WH	91	11W6	2001
		J6	2002
		J64 J4	2003
*****1 *****			2004
R190 COPY FOCUS IN DAV ORDER	R190	J45	2005
SAVE		20W0	2006
SEARCH CRITERION		J90	2007
SAVE NAME		20W2	2008
GET M13 PREF ORDER		10A8	2009
		G10	2010
		20W1	2011
	93	11W2	2012
(FOCUS)		11W0	2013
(DIM LIST)		11W1	2014
SELECT DIM		R52	2015
		7092	2016
GET COMMON VALUE ON FOCUS		R1	2017
ADD TO COPY		J65 93	2018
PGP EXTRA SS	92	30H0	2019
SET PROBS OF DIMS +		12W1	2020
	94	J60	2021
		J60	2022
		7095	2023
		12H0	2024
		J123	2025
		J8 94	2026
	95	30H0 J35	2027
	57	*01 0	2028
*****1 *****			2029
R52 SELECT PREFERRED DIM OF (0)	R52	J42	2030
SAVE LIST		60W0	2031
SELECT DIM		J16	2032
EXIT IF NONE		70J32	2033
SAVE DIM		20W1	2034
SET PROB		11W0	2035
OF DIM.		11W1	2036
OR DIM. VALUE MINUS		G9	2037
RETURN WITH SELECTED DIM		11W1 J32	2038
*****1 *****			2039
Q41 FIND UNTESTED DIM(S)	Q41	90	2040
		Q40 0	2041
	9-0	0	2042
		A2	2043
		9-2	2044
		A1	2045
		9-1 0	2046
	9-1	0	2047
		M10	2048
		K97 0	2049
	9-2	0	2050

IPL LIST

		M1		2051
		H5	0	2052
		J46		2053
Q40 FIND UNTESTED DIMS	Q40			2054
SAVE (0) NO. OF DIMS TO FIND		Z0W1		2055
(1)MEP,(2)M1		J81		2056
SAVE FOCUS COPY		Z0W6		2057
		30H0		2058
SAVE M1		60W5		2059
CLEAR M1		G30		2060
AND DIMS OF IWI		11W6		2061
HAVING PROPERTY A5		10A5		2062
		R40		2063
SAVE DDL		60W4		2064
DATA TERM		11W1		2065
CUT LIST		J200		2066
ERASE REMAINS		G30		2067
GET LIST OF CVALS		11W4		2068
COLLECT DIMS		R6		2069
MAKE DDL WITH DIMS		10A9		2070
		U20		2071
SAVE NAME		60W1		2072
PUT DIMS VALS		11W4		2073
ON DDL		10A11		2074
PUT DDL CN M1		J11		2075
		11W5		2076
		11W1		2077
PUT DDL CN M1		G2		2078
PUT FOCUS COPY ON M1		11W5		2079
		11W6		2080
		G2	J36	2081
*****1 *****				
Q42 FIND DIMS MARKED IRREL	Q42	90		2082
		Q43	0	2083
	9-0	0		2084
		A2		2085
		9-2		2086
		A1		2087
		9-1	0	2088
	9-1	0		2089
		M10		2090
		K97	0	2091
	9-2	0		2092
		M1		2093
		H5	0	2094
Q43 FIND DIMS MARKED IRREL	Q43	J46		2095
SAVE (0) NO. OF DIMS TO FIND		Z0W1		2096
(1)MEP,(2)M1		J81		2097
SAVE FOCUS COPY		Z0W6		2098
		30H0		2099
SAVE M1		60W5		2100

IPL LIST

CLEAR M1		G30		2101
AND DIMS OF IWI		11W6		2102
VALUE		10F2		2103
HAVING PROPERTY A5		10A5		2104
		R42		2105
SAVE DDL		60W4		2106
DATA TERM		11W1		2107
CUT LIST		J200		2108
ERASE REMAINS		G30		2109
GET LIST OF CVALS		11W4		2110
COLLECT DIMS		R6		2111

MAKE DDL WITH DIMS		10A9		2112
		U20		2113
SAVE NAME		60W1		2114
PUT DIMS VALS		11W4		2115
ON DDL		10A11		2116
PUT DDL ON M1		J11		2117
		11W5		2118
		11W1		2119
PUT DDL CN M1		G2		2120
PUT FOCUS COPY ON M1		11W5		2121
		11W6		2122
		G2	J36	2123
*****1 *****				
R81 FIND VAL OF ATT (0) OF LIST(1)	R81	J51		2124
ATT(0) ASSUMED SPECIFIC		11W1		2125
GET VAL OR VAL LIST	91	11W0		2126
		J10		2127
EXIT IF FOUND		70	J31	2128
NOT FOUND		11W0		2129
GET ATT (0) CLASS ATT		10A6		2130
		J10		2131
		70J31		2132
		J81		2133
GET SPECIFIC		11W1		2134
DESCRIPTION		J6		2135
		G10		2136
		70J31		2137
		60W1	91	2138
*****1 *****				
R81 MAKE (1) A VAL OF ATT(0)	R82	J52		2139
CN LIST (2)		11W0		2140
ATT(0) ASSUMED SPECIFIC		10A6		2141
GET CLASS ATT		G10		2142
NCNE,CONNECT SPEC		7091		2143
SAVE CLASS		2097		2144
MAKE DDL		11W2		2145
WITH CLASS ATT		J90		2146
		J136		2147
		60W2		2148

IPL LIST

		1197		2151
		J12		2152
MAKE SPECIFIC	91	11W2		2153
DESCRIPTION		11W1		2154
		11W0		2155
		J12	J32	2156
*****1 *****				
R40 MAKE LIST OF ELEMENTS	R40	J90		2157
OF L(1), HAVING ATT.(0)		J136		2158
SAVE OUTPUT AND ATT.		J51		2159
GENERATE L(1)		1091		2160
		J100		2161
EXIT WITH OUTPUT IN HO		11W0	J31	2162
SUB P SAVE S	91	6097		2163
ENTER ATT.		11W1		2164
CHECK IF ATT. IN DES OF S		R81		2165
NO, CONTINUE		70J4		2166
YES, ADD S TO OUTPUT L		51W0		2167
AND CONTINUE		11W0		2168
		1197	J65	2169
	97	0	0	2170
*****1 *****				
R42 MAKE LIST OF ELEMENTS FROM L(2)	R42	J90		2171
HAVING ATT (0) AND VAL (1)		J52		2172
GENERATE (2)		1091		2173

		J100		2176
RETURN WITH OUTPUTS		11W0	J32	2177
SAVE S	91	6097		2178
CHECK FOR ATT (0)		11W1		2179
		G10		2180
NO EXIT		70J4		2181
YES ,CHECK VAL		11W2		2182
		J2		2183
		70J4		2184
NO,EXIT		11W0		2185
YES ADD S TO OUT L		1197	J65	2186
	97	0	0	2187
*****1 *****				2188
R41 MAKE LIST OF ELEMENTS	R41	J90		2189
OF L(1), NOT HAVING ATT.(0)		J136		2190
SAVE OUTPUT AND ATT.		J51		2191
GENERATE LIST (1)		1091		2192
		J100		2193
EXIT WITH OUTPUT IN HO		11W0	J31	2194
SAVE S FROM L(1)	91	6097		2195
ENTER ATT (0)		11W1		2196
TEST IF ATT NOT ON DES OF S		R81		2197
YES CONTINUE		70	J8	2198
NO,ADD S TO OUTPUT LIST		11W0		2199
AND CONTINUE		1197		2200

IPL LIST

		J65	J4	2201
	57	0	0	2202
*****1 *****				2203
R100 GENERATE SYMBS FROM	R100	10W1		2204
(1) AND (2) IN PARALLEL FOR (0)		J17		2205
SAVE LISTS (1),(2)		J21		2206
DOWN 1 ON (1)	9-1	11W0		2207
		J60		2208
		20W0		2209
EXIT IF END LIST		70J19		2210
DOWN 1 ON LIST (2)		11W1		2211
		J60		2212
		20W1		2213
EXIT IF END LIST		70J19		2214
ENTER SYMB FROM (2)	95	12W1		2215
ENTER SYMB FROM (1)		12W0		2216
EXECUTE SUB PROCESS		J18		2217
EXIT IF SUB-P RETURNS		70J19	9-1	2218
U6 GEN PSEUDO PROGRAM	U6	2099		2219
STRUCTURE SYMBS FOR SUB P (0)		91		2220
POP INITIAL ENTRY		3097		2221
ERASE LIST OF SUB LISTS		1098		2222
(CANT BE USED IN RECURSIVE ROUTN)		J75	J71	2223
PUT LOC L	91	4097		2224
SAVE LOC		6097		2225
ADD SUB L NAME		1098		2226
		J6		2227
		J66		2228
TC LIST		1197		2229
SCAN SUB LIST	95	J60		2230
		2097		2231
IF END EXIT +		70J4		2232
ENTER S TWICE		1297		2233
		4090		2234
LOCAL		J132		2235
		70	92	2236
SET SIGN + BEFORE XEQ SUB P		J4		2237
NOT LOC PRESENT S TO SUB P		0199		2238
CONTINUE OR QUIT		700	95	2239

SUB L, SEEN BEFORE	92	1098		2240
		J6		2241
		J77		2242
IF SEEN GO TO 95 (CONTINUE)		70	95	2243
NOT SEEN THEN GEN IT		1297		2244
		91		2245
EXIT TO POINT (POP LOC L)		3097		2246
IF - EXIT AGAIN		700	95	2247
	57	0	0	2248
LIST OF USED LOCALS	98	0	0	2249
	95	0	0	2250

IPL LIST

G11 (EQUIV TO J11 EXCEPT NO ERASE)	G11	J51		2251
		52H0		2252
		11W0		2253
		J62		2254
		7091		2255
		J60		2256
		20W0		2257
		11W1		2258
		21W0	J31	2259
	91	40H0		2260
		11W0		2261
		J65		2262
		11W1		2263
		J65	J31	2264
*****1 *****				2265
R6 COLLECT LIST OF (NEXT UPPER)	R6	J90		2266
OF LIST (0)		J136		2267
SAVE NAME OF OUTPUT		J50		2268
GENERATE LIST (0)		1091		2269
		J100		2270
EXIT WITH NAME OF OUT		11W0	J30	2271
SUB P GET VAL OF SYMMETRY ATT.	91	10A6		2272
I.E. NEXT UPPER		610		2273
NCNE, CONTINUE		70J4		2274
ADD IT TO OUTPUT LIST		11W0		2275
AND CONTINUE		J6	J65	2276
*****1 *****				2277
R73 COPY (C) WITHOUT ITS DESCRIPT	R73	J73		2278
MAKE OUTPUT NAME LOCAL		J136		2279
SAVE NAME		6097		2280
BLCT OUT DES ON OUTPUT		100		2281
		2197	0	2282
	97	0	0	2283
*****1 *****				2284
R39 MAKE STRUCTURE (0) SYMMETRIC	R39	40H0		2285
DOUBLE LIST NAME, SAVE BOTH		J51		2286
DCWN 1 ON LIST (0)	91	11W0		2287
		J60		2288
		20W0		2289
EXIT IF END		70J31		2290
ENTER SYM		12W0		2291
MAKE LIST NAME A VAL ON DES		11W1		2292
OF ITS SYMB		10A6		2293
UNDER ATT A6		J12		2294
ENTER SYMB		12W0		2295
RECURSE AND CONTINUE		R39	91	2296
*****1 *****				2297
R38 ERASE VAL LIST OF ATT.(0) ON (R38	J50		2298
1) ,GENERATE (1)		1091		2299
EXIT WHEN DONE		J100	J30	2300

IPL LIST

SUB P ,DOUBLE LIST NAME	91	40H0		2301
ENTER ATT.		11W0		2302
DELETE ATT AND VAL FROM LIST		J14		2303
ENTER ATT.		11W0		2304
RECURSE		R38	0	2305
*****1 *****				
G210 IS DES OF (1) IN (2) UNDER CL	6210	J44		2306
CLASS ATT		20W1		2307
SAVE (1) DES		20W4		2308
		20W3		2309
GEN SPEC ATTS OF CLASS ATT		11W1		2310
		1091		2311
		J100	J34	2312
CCMPARE DESES	91	20W2		2313
GET VAL GF SPEC ATT FROMR1		11W4		2314
		11W2		2315
		G10		2316
IF NONE CONTINUE		70J5		2317
GET VAL FROM ROUTINE 2		11W3		2318
		11W2		2319
CCMPARE THE TWO		G10		2320
		70J8	J2	2321
*****1 *****				
R3 (0)-(1) DIFFERENCE	R3	R73		2322
DELETE SYMS ON (0) FROM (1)		6097		2323
COPY (0) SAVE NAME		J6		2324
GENERATE (1)		1091	J100	2325
DELETE S ON (1) FROM (0)	91	1197		2326
		J6		2327
		J69	J4	2328
	97	0	0	2329
*****1 *****				
R4 MARK ELEMENTS OF L (2)	R4	J51		2330
WITH ATT (0) AND VAL (1)		1091		2331
		J100	J31	2332
MARK GENERATED SYMS WITH ATT	91	11W1		2333
AND VAL		11W0	J12	2334
*****1 *****				
R5 COPY LIST (1) AND CUT OFF	R5	J50		2335
COPY AFTER (0)TH SYMB,OUTPUT COP		R73		2336
SAVE COPY NAME IN H0		40H0		2337
LOCATE (0)TH S		11W0		2338
CUT,ERASE REMAINS		J200		2339
		G30	J30	2340
*****1 *****				
G1 FETCH 1ST LIST S OF LIST (0)	G1	40H5		2341
		J81		2342
PGP H5 EXIT		30H5	0	2343
*****1 *****				
G4 INSERT	G4	J51		2344

IPL LIST

(0) IN FIRST LIST LOC		11W1		2351
OF (1) NO PUSH DOWN	92	J60		2352
IF EMPTY PUSH DOWN		T0	91	2353
		41HC	92	2354
	91	20W1		2355
		11W0		2356
		21W1	J31	2357
*****1 *****				
G2 PUT (0) IN 1ST LIST POS. OF (1)	G2	J6		2358
SAVE LIST NAME		J50		2359
SAFE H5		40H5		2360

PUSH DOWN LIST		41W0		2362
LOCATE 1ST POS		11W0		2363
		J60		2364
		20W0		2365
		30R5		2366
PUT S IN		21W0	J30	2367
*****				2368
G8 PICK YES,NO SYMBS VIA H5	G8	1091		2369
		70J82	J81	2370
	91	0		2371
		F1		2372
		F2	0	2373
*****				2374
G10 GET 1ST VAL OFF VAL LIST	G10	R81		2375
OF ATT(0) OF LIST (1)		700	J81	2376
*****				2377
G12 GET MU,NI,(M1,0) FROM T1	G12	10F3		2378
DEPENDING ON (0)		J2		2379
		70J82	J81	2380
*****				2381
G9 SET VALUE OF ATT. (0) OF	G9	J10		2382
LISTV (1) MINUS		700		2383
		J123	J8	2384
*****				2385
R2 LOCATE LIST (0) EQUIVALENT	R2	J51		2386
ON LIST CF LISTS (1)	91	11W1		2387
		J60		2388
NEXT ON (1)		20W1		2389
END, QUIT		70J31		2390
IS 1(1) EQUIV (0)		12W1		2391
		11W0		2392
		R8		2393
NO,CONTINUE SCAN		7091		2394
YES RETURN LOC.		11W1	J31	2395
*****				2396
R8 (0) .C. (1)	R8	J51		2397
ENTER LIST (0)		11W0		2398
ENTER SUB P		109-1		2399
GENERATE AND TEST		J100	J31	2400

IPL LIST

SUB P TEST IF S ON (1)	9-1	11W1		2401
		J6	J77	2402
*****				2403
R1 RETURN SYMB IF ON LISTS (0),(1)	R1	J51		2404
ENTER LIST1		11W1		2405
SUBPROCESS		109-1		2406
GENERATE LIST (1)		J100		2407
INVERT H5 AND EXIT		J5	J31	2408
DCUBLE SYMB FROM LIST (1)	9-1	40H0		2409
ENTER LIST (0)		11W0		2410
INVERT ARGS		J6		2411
SYMB (0) ON LIST (1)		J77		2412
REVERSE H5		J5		2413
IF - FOUND, P CONTINUE		700	J8	2414
*****				2415
U20 MAKE UP SPECIFIC	U20	J51		2416
ATT. DESCRIPTION LIST		J90		2417
(0) ATT., (1) VALUE		J136		2418
DOUBLE NAME		40H0		2419
OUTPUT (0) NAME OF DESCRIPTION		11W1		2420
		11W0		2421
		J11	J31	2422
*****				2423
G30 ERASE BODY OF LIST (0)	G30	J75	J71	2424

*****1 *****				2425
EXPERIMENTERS INST TO SUBJECT	L60	0		2426
		L41		2427
		L42		2428
		L43		2429
		L44		2430
		L45		2431
		L46		2432
		L47		2433
		L48		2434
		L49		2435
		L50		2436
		L51		2437
		L52	0	2438
*****1 *****				2439
PROBLEM DETAILS	L41	90		2440
		94	0	2441
	90	J		2442
MESSAGE TYPE		A60		2443
		920		2444
		A66		2445
		L9	0	2446
DDL CONTAINING MESSAGE	94	95	0	2447
	95	0		2448
FIRST PCB FLAG		A65		2449
		FI		2450

IPL LIST

EXTERNAL ENVIRONMENT		A61		2451
		921		2452
STRUCTURE OF ENVIRONMENT		A50		2453
		922		2454
TYPE OF CONCEPT		A63		2455
		923		2456
GCAL		A64		2457
ALL CTR VALUE (YOB)	All	Z		2458
		924	0	2459
	920	0		2460
PROBLEM DETAILS		F61	0	2461
	921	0		2462
		E0		2463
	922	J		2464
		M13	0	2465
	923	0		2466
		F80	0	2467
	924	0		2468
		F1	0	2469
HIDDEN ASSUMPTIONS	L5	926	0	2470
	926	0		2471
AWARE		A91		2472
		91		2473
DIMENSIONS IN WH		A93		2474
		92		2475
DIMENSIONS ACTUALLY VARIED		A94		2476
		93	0	2477
	91	0		2478
AWARE		F7	0	2479
	92	0		2480
		K99	0	2481
	93	0		2482
		K98	0	2483
*****1 *****				2484
START BY SHOWING FOCUS CARD	L42	90		2485
		94	0	2486
DESCRIPTION OF PROCESS	90	0		2487
MESSAGE TYPE		A60		2488

DDL CONTAINING MESSAGE	94	920	0	2489
	95	95	0	2490
TYPE OF PROCSSS		0		2491
		A30		2492
COMMUNICATION		96		2493
		A31		2494
INFORMATION CREATED		97		2495
		A32		2496
DESIGNATION INFO		98		2497
		A33		2498
VALUE OF A30	56	99	0	2499
		0		2500

IPL LIST

COING		F30	0	2501
VALUE OF A31	57	0		2502
FROM EXTERNAL		F41	0	2503
VALUE OF A32	58	0		2504
FOCUS OBJECT		F51	0	2505
	95	0	0	2506
	920	0		2507
PROCESS SPEC		F62	0	2508
*****1*****				2509
SELECT AN OBJECT	L43	90		2510
		94	0	2511
DESCRIPTION OF PROCESS	90	0		2512
MESSAGE TYPE		A60		2513
		920	0	2514
DDL CONTAINING MESSAGE	94	95	0	2515
	95	0		2516
TYPE OF PROCESS		A30		2517
		96		2518
COMMUNICATION		A31		2519
		97		2520
INFORMATION CREATED		A32		2521
		98		2522
DESIGNATION INFO		A33		2523
		99	0	2524
VALUE OF A30	96	0		2525
COING		F30	0	2526
	57	0	0	2527
VALUE OF A32	58	0		2528
SEARCH CRITERION		F52	0	2529
	95	0	0	2530
	920	0		2531
PROCESS SPEC		F62	0	2532
*****1*****				2533
FROM THE BOARD	L44	90		2534
		94	0	2535
DESCRIPTION OF PROCESS	90	0		2536
MESSAGE TYPE		A60		2537
		920	0	2538
DDL CONTAINING MESSAGE	94	95	0	2539
	95	0		2540
TYPE OF PROCSSS		A30		2541
		96		2542
COMMUNICATION		A31		2543
		97		2544
INFORMATION CREATED		A32		2545
		98		2546
DESIGNATION INFO		A33		2547
		99	0	2548
VALUE OF A30	96	0		2549
COING		F30	0	2550

IPL LIST

VALUE OF A31	97	0		2551
FROM EXTERNAL		F41	0	2552
VALUE OF A32	98	0		2553
OBJECT CHOICE		F53	0	2554
	99	0	0	2555
	920	0		2556
PROCESS SPEC		F62	0	2557
*****1 *****				2558
WHICH WILL BE DESIGNATED	L45	90		2559
		94	0	2560
DESCRIPTION OF PROCESS	9C	0		2561
MESSAGE TYPE		A60		2562
		920	0	2563
DDL CONTAINING MESSAGE	94	95	0	2564
	95	0		2565
TYPE OF PROCESS		A30		2566
		96		2567
COMMUNICATION		A31		2568
		97		2569
INFORMATION CREATED		A32		2570
		98		2571
DESIGNATION INFO		A33		2572
		99	0	2573
VALUE OF A30	96	0		2574
COING		F30	0	2575
INTERN EXTERN COMMUNE	97	0		2576
		F42	0	2577
VALUE OF A32	98	0		2578
OBJECT CHOICE		F53	0	2579
VALUE OF A33	99	0		2580
OBJECT		F71	0	2581
	920	0		2582
PROCESS SPEC		F62	0	2583
*****1 *****				2584
BY YES OR NO	L46	90		2585
		94	0	2586
	9C	0		2587
MESSAGE TYPE		A60		2588
		920		2589
DDL CONTAINING MESSAGE	94	95	0	2590
	95	0		2591
DESIGNATION ATT.		A7		2592
		931	0	2593
	920	0		2594
ATT SPEC		F63	0	2595
	931	0		2596
YES		F1		2597
NO		F2	0	2598
*****1 *****				2599
WHEN YOU HAVE THE CONCEPT	L47	90		2600

IPL LIST

		94	0	2601
DESCRIPTION OF PROCESS	9C	0		2602
MESSAGE TYPE		A60		2603
		920	0	2604
DDL CONTAINING MESSAGE	94	95	0	2605
	95	0		2606
TYPE OF PROCESS		A30		2607
		96		2608
COMMUNICATION		A31		2609
		97		2610
INFORMATION CREATED		A32		2611

		98		2612
DESIGNATION INFO		A33		2613
		99	0	2614
VALUE OF A30	56	0		2615
CODING		F31	0	2616
	57	0	0	2617
VALUE OF A32	58	0		2618
CONCEPT		F54	0	2619
	95	0	0	2620
	920	0		2621
PROCESS SPEC		F62	0	2622
*****1 *****				2623
FORM THE CONCEPT	L48	90		2624
		94	0	2625
DESCRIPTION OF PROCESS	9C	0		2626
MESSAGE TYPE		A60		2627
		920	J	2628
DDL CONTAINING MESSAGE	94	95	0	2629
	95	0		2630
TYPE OF PROC5SS		A30		2631
		96		2632
COMMUNICATION		A31		2633
		97		2634
INFORMATION CREATED		A32		2635
		98		2636
DESIGNATION INFO		A33		2637
		99	0	2638
VALUE OF A30	56	0		2639
CODING		F30	0	2640
	57	J	0	2641
VALUE OF A32	98	0		2642
CONCEPT		F54	0	2643
	95	J	0	2644
	920	0		2645
PROCESS SPEC		F62	0	2646
*****1 *****				2647
WHICH WILL BE DESIGNATED	L49	90		2648
		94	0	2649
DESCRIPTION OF PROCESS	9C	0		2650

IPL LIST

MESSAGE TYPE		A60		2651
		920	0	2652
DDL CONTAINING MESSAGE	94	95	0	2653
	95	0		2654
TYPE OF PROC5SS		A30		2655
		96		2656
COMMUNICATION		A31		2657
		97		2658
INFORMATION CREATED		A32		2659
		98		2660
DESIGNATION INFO		A33		2661
		99	0	2662
VALUE OF A30	56	0		2663
CODING		F30	0	2664
VALUE OF A31	57	0		2665
BOTH		F42	J	2666
VALUE OF A32	58	0		2667
CONCEPT		F54	0	2668
VALUE OF A33	95	0		2669
CONCEPT		F72	0	2670
	920	0		2671
PROCESS SPEC		F62	0	2672
*****1 *****				2673
YES OR NO	L50	90		2674
		94	0	2675

	9C	0		2676
MESSAGE TYPE		A60		2677
		920	0	2678
DDL CONTAINING MESSAGE	94	95	0	2679
	95	0		2680
EXP DESIGNATION OF CONCEPT		A27		2681
		941	0	2682
	941	0		2683
YES		F1		2684
NO		F2	0	2685
	920	0		2686
ATT SPEC		F63	0	2687
*****				2688
IF THE CONCEPT IS CORRECT	L51	90		2689
		94	0	2690
DESCRIPTION OF PROCESS	90	0		2691
MESSAGE TYPE		A60		2692
		920	0	2693
DDL CONTAINING MESSAGE	94	95	0	2694
	95	0		2695
TYPE OF PROCESS		A30		2696
		96		2697
COMMUNICATION		A31		2698
		97		2699
INFORMATION CREATED		A32		2700

IPL LIST

		98		2701
DESIGNATION INFO		A33		2702
		99	0	2703
VALUE OF A30	56	J		2704
DECIDING		F31	0	2705
	57	0	0	2706
VALUE OF A32	58	0		2707
CONCEPT		F54	0	2708
VALUE OF A33	95	0		2709
CONCEPT		F72	0	2710
	920	0		2711
PROCESS SPEC		F62	0	2712
*****				2713
THATS IT	L52	90		2714
		94	0	2715
DESCRIPTION OF PROCESS	9C	0		2716
MESSAGE TYPE		A60		2717
		920	0	2718
DDL CONTAINING MESSAGE	94	95	0	2719
		95	0	2720
MESSAGE TERMINATION		A81		2721
		96	0	2722
	56	J		2723
		F65	0	2724
	920	0		2725
TERMINATION		F65	0	2726
*****				2727
A1 PROCESS INPUT	A1	0	0	2728
A2 PROCESS OUTPUT	A2	0	0	2729
A3 PROCESS DESCRIPT (CL) ATT.	A3	0		2730
		A30		2731
		A31		2732
		A32		2733
		A33	0	2734
A4 HCM OBJ FOUND	A4	0	0	2735
A5 REL, IRREL	A5	J	0	2736
A6 (NEXT UPPER) SYMMETRY	A6	0	0	2737
A7 EXP DESIG OF CARD CHOICE	A7	90	0	2738
	9C	0		2739

		A6		2740
		91	0	2741
	91	J		2742
A9 DIM VARIED ATT.		A15	0	2743
	A9	90	0	2744
	9C	J		2745
		A6		2746
		91	0	2747
	91	0		2748
A10 DIM VALUE FROM		A12	0	2749
	A10	90	0	2750

IPL LIST

		9C	0	2751
		A6		2752
		91	0	2753
	91	0		2754
		A12	0	2755
	9C	0		2756
		A6		2757
		91	J	2758
	91	0		2759
A12 CLASS ATT. S.C. FORMATION		A12	0	2760
	A12	0		2761
		A9		2762
		A10		2763
		A11	0	2764
A15 SET MEMBERSHIP ATT. (CLASS)	A15	0	0	2765
A16 OBJ FOUND	A16	0	0	2766
A17 CONCEPT FOUND	A17	0	0	2767
A18 WORKING HYPOTHESIS	A18	0	0	2768
A19 CONCEPT (HOW FORMED)	A19	0	0	2769
A20 L100 ,VAL IS FOCUS COPY	A20	0	0	2770
A27 EXP DESIG OF CONCEPT	A27	90		2771
	9C	0		2772
		A6		2773
		91	0	2774
	91	0		2775
		A15	0	2776
A26 HOW OBJ FOUND	A26	J	J	2777
A29 BASIS FOR CONST. CONCEPT	A29	0	0	2778
A30 PROCESS TYPE	A30	0	0	2779
A31 PROCESS COMMUNICATION	A31	J	0	2780
A32 INFORMATION CREATED	A32	0	0	2781
A33 DESIGNATION INFORMATION	A33	0	0	2782
A50 E9 ATT VAL IS M13	A50	J	0	2783
A60 EXP MESSAGE TYPE	A60	0	0	2784
A61 EXTERNAL ENVIRONMENT	A61	0	0	2785
A63 TYPE OF CONCEPT	A63	0	0	2786
A64 THE GOAL	A64	0	0	2787
A65 FIRST PROBLEM ATT. (VAL=F1)	A65	0	0	2788
A66 HIDDEN ASSUMPTN NOT KNO TO SUBJ	A66	0	0	2789
A80 PROBLEM DETAIL SPECIFICATION	A80	0	0	2790
A81 MESSAGE TERMINATION	A81	0	0	2791
A91 AWARENESS ATTRIBUTE	A91	0	0	2792
A93 DIMENSIONS IN WORK HYP	A93	0	0	2793
A94 DIMENSIONS ACTUALLY VARED	A94	0	0	2794
A111 POSSIBLE VALUES OF SPEC ATT	A111	0	0	2795
A30J L100 TYPE VAL IS E9	A30J	J	0	2796
A302 L100 , VAL IS STRAT	A302	0	0	2797
F1 YES, RELAVANT	F1	0	0	2798
F2 NO, IRRELVANT	F2	J	J	2799
F3 FLAG (N1,D)	F3	0	0	2800

IPL LIST

F4 FLAG (M1,N)	F4	0	0	2801
F5 CCNTAINS CONCEPT	F5	J	0	2802
F6 CONTAINS NAME FOCUS CARD	F6	0	0	2803
F7 AWARE	F7	0	0	2804
F8 UNAWARE	F8	J	0	2805
F30 COING	F30	0	0	2806
F31 DECIDING	F31	0	0	2807
F32 CCNTEXTING	F32	0	0	2808
F40 TO EXTERNAL	F40	0	0	2809
F41 FROM EXTERNAL	F41	0	0	2810
F42 BOTH	F42	0	0	2811
F51 FOCUS CBJECT	F51	0	0	2812
F52 S.C.	F52	0	0	2813
F53 OBJECT CHOICE	F53	0	0	2814
F54 CCNCEPT	F54	0	0	2815
F61 PROB LETAILS	F61	0	0	2816
F62 PROCESS SPEC	F62	0	0	2817
F63 ATTRIBUTE SPEC	F63	J	J	2818
F64 FOCUS SPEC	F64	0	0	2819
F65 TERMINATION	F65	0	0	2820
F71 DESIGNATE OBJ	F71	0	0	2821
F72 LESIGNATE CCNCEPY	F72	0	0	2822
F80 TYPE OF CONCEPT	F80	0	0	2823
LTM	H3	J		2824
		S2	0	2825

N10 PROB E10	N10	+01	50	2827
N20 PROB E20	N20	+01	50	2828
N30 PROB E30	N30	+01	50	2829
N40 PROB E40	N40	+01	50	2830
N50 PROB E50	N50	+01	50	2831
N11 PROB E11	N11	+01	50	2832
N12 PROB E12	N12	+01	50	2833
N21 PROB E21	N21	+01	50	2834
N22 PROB E22	N22	+01	50	2835
N31 PROB E31	N31	+01	50	2836
N32 PROB E32	N32	+01	50	2837
N41 PROB E41	N41	+01	50	2838
N42 PROB E42	N42	+01	50	2839
N51 PROB E51	N51	+01	50	2840
N52 PROB E52	N52	+01	50	2841

POSITIVE INTEGAL CONSTANTS	K1	+01	1	2843
	K2	+01	2	2844
	K3	+01	3	2845
	K4	+01	4	2846
	K5	+01	5	2847
	K6	+01	6	2848
	K7	+01	7	2849
	K8	+01	8	2850

IPL LIST

	K9	+01	9	2851
	K10	+01	10	2852
	K56	+01	1	2853
K97 NUM UNTESED DIM TO FIND Q40	K57	+01	1	2854
K98 NUM DIM TO VARY IN P130	K58	+01	1	2855
K99 NUM DIM IN WH EST BY P190	K59	+01	5	2856

ELEMENTS	E11	0	0	2858
	E12	0	0	2859
	E21	0	0	2860
	E22	0	0	2861

	E31	0	0	2862
	E32	0	0	2863
	E41	0	0	2864
	E42	0	0	2865
	E51	0	0	2866
	E52	0	0	2867
*****1 *****				2868
E1 SUBJECTS OUTPUT CHANNEL TO EXP.	E1	0	0	2869
M1 SHORT TERM MEMORY (STM)	M1	0	0	2870
M10 MEMORY ENTRY POINT	M10	0	0	2871
M13 DIMENSION LIST	M13	9-1		2872
		E10		2873
		E20		2874
		E30		2875
		E40		2876
		E50	0	2877
	9-1	J		2878
		AB		2879
		9-2	0	2880
	9-2	J		2881
		9-3	0	2882
	9-3	9-0	0	2883
9-0 PREFERENCE ORDER OF DIMS	9-0	J		2884
		E10		2885
		N10		2886
		E20		2887
		N20		2888
		E30		2889
		N30		2890
		E40		2891
		N40		2892
		E50		2893
		N50	0	2894
*****1 *****				2895
E10 DIM VALUE LIST	E10	9-1		2896
		E11		2897
		E12	0	2898
	9-1	0		2899
		AB		2900
IPL LIST				
		92	0	2901
	92	0		2902
		93	0	2903
	93	9-0	0	2904
9-0 DIM VALUE PREF. ORDER LIST	9-0	0		2905
		E11		2906
		N11		2907
		E12		2908
		N12	0	2909
*****1 *****				2910
	E20	9-1		2911
		E21		2912
		E22	0	2913
	9-1	0		2914
		AB		2915
		92	0	2916
	92	0		2917
		93	0	2918
	93	9-0	0	2919
9-0 DIM VALUE PREF. ORDER LIST	9-0	0		2920
		E21		2921
		N21		2922
		E22		2923
		N22	0	2924

*****1 *****				2925
	E30	9-1		2926
		E31		2927
		E32	0	2928
	9-1	0		2929
		A8		2930
		92	0	2931
	92	0		2932
		93	0	2933
	93	0	0	2934
9-0 CIM VALUE PREF. ORDER LIST	9-0	J		2935
		E31		2936
		N31		2937
		E32		2938
		N32	0	2939
*****1 *****				2940
	E40	9-1		2941
		E41		2942
		E42	0	2943
	9-1	J		2944
		A8		2945
		92	0	2946
	92	0		2947
		93	0	2948
	93	9-0	0	2949
9-0 CIM VALUE PREF. ORDER LIST	9-0	0		2950
IPL LIST				
		E41		2951
		N41		2952
		E42		2953
		N42	0	2954
*****1 *****				2955
	E50	9-1		2956
		E51		2957
		E52	0	2958
	9-1	0		2959
		A8		2960
		92	0	2961
	92	0		2962
		93	0	2963
	93	9-0	0	2964
9-0 CIM VALUE PREF. ORDER LIST	9-0	0		2965
		E51		2966
		N51		2967
		E52		2968
		N52	0	2969
	5			2970
	5	S1		2971

IPL POST LIST

NAME	LOC.	REFERENCES											
A1	2728	89	195	317	538	695	881	1141	1168	1194	1231	1269	1323
		1357	1393	1406	1419	1432	1445	1458	1471	1484	1497	1518	1530
		1541	1558	1570	1586	1624	1635	1682	1746	1796	1850	1868	1924
		1961	1982	2045	2087								
A2	2729	94	883	1143	1166	1192	1229	1267	1321	1355	1391	1404	1417
		1430	1443	1456	1469	1482	1495	1516	1528	1539	1556	1568	1594
		1633	1680	1744	1794	1848	1866	1922	1959	1980	2043	2085	
A3	2730	141	282	308	462	472	516	558	623	726	754	778	806
		833	858	922	952	976	1003	1029	1056	1090	1114		
A4	2735												
A5	2736	160	653	913	1242	1298	1667	1722	1731	1781	1859	1903	2061
		2104											
A6	2737	47	390	483	530	546	647	1180	1182	2131	2143	2272	2293
		2740	2746	2752	2757	2773							
A7	2738	576	1156	1614	1638	1685	2592						
A8	0	1289	2009	2879	2900	2915	2930	2945	2960				
A9	2744	711	1211	1250	1279	1709	2069	2112	2762				
A10	2750	1215	1254	1283	1336	1662	1757	1769	1776	1813	1823	2763	
A11	2458	715	901	1304	1340	1761	1773	1778	1808	1828	1997	2073	2116
		2764											
A12	2761	717	1376	1439	1574	1640	1687	2749	2755	2760			
A15	2765	293	1413	1504	1562	2743	2776						
A16	2766	1452											
A17	2767	1478											
A18	2768	1426	1991										
A19	2769	1491											
A20	2770	1400											
A26	2777	898	1465										
A27	2771	633	1946	1965	2681								
A29	2778												

IPL POST LIST

NAME	LOC.	REFERENCES											
A30	2779	404	426	732	760	784	812	839	864	928	958	982	1009
		1035	1062	1096	1120	2492	2517	2541	2566	2607	2631	2655	2696
		2731											
A31	2780	371	734	762	786	814	841	866	930	960	984	1011	1037
		1064	1098	1122	1892	2494	2519	2543	2568	2609	2633	2657	2698
		2732											

A32	2781	382	736	764	788	816	843	868	932	962	986	1013	1039
		1066	1100	1124	2456	2521	2545	2570	2611	2635	2659	2700	2733
A33	2782	376	738	766	790	818	845	870	934	964	988	1015	1041
		1068	1102	1126	2498	2523	2547	2572	2613	2637	2661	2702	2734
A50	2783	234	245	2453									
A60	2784	209	213	2443	2488	2513	2537	2562	2588	2603	2627	2651	2677
		2692	2717										
A61	2785	2451											
A63	2786	2455											
A64	2787	2457											
A65	2788	181	252	337	2449								
A66	2789	2445											
A80	2790	249	334										
A81	2791	2721											
A91	2792	584	676	2472									
A93	2793	2474											
A94	2794	2476											
A111	2795	299											
A300	2796	241											
A302	2797	261	275	329									
C3	709	503	506	533	571	650							
C8	691	687											
C10	201	193											

IPL POST LIST

NAME	LOC.	REFERENCES
C11	192	186
C12	177	189
C13	205	203
C20	466	460
C21	459	449 482
C22	616	614
C23	613	457 646 688
C31	1513	493 723 751 83C 1000 1087
C36	511	509

408

C37	508	451	529					
C38	535	455	545					
C39	542	536						
C41	1621	594	597	661	664	947	1079	1080
C50	229	221						
C51	269	223						
C52	285	225						
C60	322	315	442					
C61	314	188						
C101	302	364						
D	855	515	519					
D1	973	124						
D2	1052	130						
D3	1111	134						
D4	775	499	505					
D40	798	776						

IPL POST LIST

NAME	LCC.	REFERENCES						
E	0	1361	2463					
E1	2869	1535	1546	1599	1927			
E2	1149	1139						
E3	1602	1592						
E4	1930	1920						
E10	2896	2873	2885					
E11	2858	2897	2906					
E12	2878	2898	2908					
E20	2911	2874	2887					
E21	2860	2912	2921					
E22	2861	2913	2923					
E30	2926	2875	2889					
E31	2862	2927	2936					
E32	2863	2928	2938					
E40	2941	2876	2891					
E41	2864	2942	2951					

E42	2865	2943	2953																
E50	2956	2877	2893																
E51	2866	2957	2966																
E52	2867	2958	2968																
E93	1551	919																	
E94	1519	1026																	
E95	1138	187																	
F1	2798	255	340	578	636	1153	1784	1966	2372	2450	2469	2597	2684						
F2	2759	180	1912	2103	2373	2598	2685												

IPL POST LIST

NAME	LCC.	REFERENCES																	
F3	2800	1397	1423	1449	1475	2378													
F4	2801	1410	1436	1462	1488	1501													
F5	2802	145	1608	1936	1939														
F6	2803	651	894	1146															
F7	2804	586	678	2479															
F8	2805																		
F30	2806	741	769	821	848	937	967	1018	1044	1105	2501	2526	2550						
		2575	2640	2664															
F31	2807	407	429	793	873	991	1071	1129	2616	2705									
F32	2808	465																	
F40	2809																		
F41	2810	743	850	2503	2552														
F42	2811	939	1046	2577	2666														
F51	2812	448	745	772	876	2539													
F52	2813	824	1108	2525															
F53	2814	450	454	796	852	941	2554	2579											
F54	2815	456	994	1021	1048	1074	2619	2643	2668	2708									
F61	2816	220	2461																
F62	2817	222	2508	2532	2557	2583	2622	2646	2672	2712									
F63	2818	224	2595	2687															
F64	2819																		
F65	2820	226	2724	2726															

F71	2E21	943	971	2581									
F72	2822	1050	1076	2670	2710								
F80	2823	2467											
G1	2346	1177	1179	1232	1240	1331	1366	1550	1606	1934			

IPL POST LIST

NAME	LOC.	REFERENCES											
G2	2359	265	1158	1161	1221	1223	1260	1262	1382	1386	1551	1586	1589
		1616	1619	1717	1720	1834	1837	1898	1901	1950	1953	2077	2080
		2120	2123										
G4	2350	348	1187	1994									
G8	2369	1611	1942										
G9	2382	1293	2037										
G10	2375	214	235	276	330	335	372	377	383	405	427	577	585
		634	677	899	902	1181	1183	1290	1299	1377	1584	1654	1660
		1701	1707	1782	1860	1904	1972	1992	2010	2137	2144	2180	2273
		2317	2321										
G11	2251	391											
G12	2378	1509											
G30	2424	150	152	478	522	552	642	1151	1218	1247	1257	1374	1579
		1604	1714	1831	1895	1932	2059	2066	2101	2109	2344		
G210	2307	309	473	517	556	624							
K1	2843	590	683										
K2	2844												
K3	2845												
K4	2846												
K5	2847												
K6	2848												
K7	2849												
K8	2850												
K9	2851												
K10	2852												
K96	2853	1642	1689										
K97	2854	2049	2091										
K98	2855	589	1235	2483									

IPL POST LIST

NAME	LOC.	REFERENCES
K99	2856	1198 2481
L9	2470	583 675 2446
L41	2440	178 2427
L42	2485	2428
L43	2510	2429
L44	2534	2430
L45	2559	2431
L46	2585	2432
L47	2600	2433
L48	2624	2434
L49	2648	2435
L50	2674	2436
L51	2689	2437
L52	2714	2438
L60	2426	154
M1	2870	77 151 153 166 198 631 1148 1171 1173 1200 1237 1272 1274 1326 1328 1363 1396 1409 1422 1435 1448 1461 1474 1487 1500 1521 1564 1576 1601 1641 1644 1688 1691 1749 1751 1799 1801 1873 1929 1964 1985 1987 2051 2093
M3	2824	170 266 320
M10	2871	80 82 149 195 263 273 321 541 886 1197 1234 1360 1399 1412 1425 1438 1451 1464 1477 1490 1503 1523 1533 1544 1561 1573 1627 1639 1686 1853 1871 2048 2090
M13	2872	139 159 2465
M10	2827	2886
M11	2832	2907
M12	2833	2909
M20	2828	2888

IPL POST LIST

NAME	LOC.	REFERENCES
N21	2834	2922
N22	2835	2924
N30	2829	2890
N31	2836	2937

N32	2837	2939								
N40	2830	2892								
N41	2838	2952								
N42	2839	2954								
N50	2831	2894								
N51	2840	2967								
N52	2841	2969								
P20	1174	1164								
P21	1163	721								
P50	1364	1353								
P51	1352	828								
P60	1505	1389	1402	1415	1428	1441	1454	1467	1480	1493
P61	1388	722								
P62	1401	724	920							
P63	1414	490	750							
P64	1427	496	673	752	804	950	1088			
P65	1440	829								
P66	1453	831								
P67	1466	999								
P68	1479	1001								
P69	1492	1027								

IPL POST LIST

NAME	LOC.	REFERENCES			
P70	1547	1514	1526	1537	
P71	1525	918			
P72	1536	1025			
P90	1694	1678			
P91	1677	603	948		
P95	1647	1631			
P96	1630	600			
P100	1752	1742			
P101	1741	581			
P120	1874	1864			

P121	1883	998	
P130	1238	1227	
P131	1226	801	
P140	1275	1265	
P141	1264	802	
P150	1329	1319	
P151	1318	803	
P170	1802	1792	
P171	1791	572	949
P180	1588	1978	
P181	1577	670	1086
P190	1201	1190	
P191	1189	487	749
P400	1628	1622	
P500	1577	1554	1566

IPL POST LIST

NAME	LCC.	REFERENCES	
P501	1553	946	1053
P502	1565	606	
Q10	1569	1957	
Q12	1556	1054	
Q40	2053	2041	
Q41	2040	656	1081
Q42	2082	658	
Q43	2095	2083	
Q50	889	879	
Q51	878	524	856
Q100	1856	1846	
Q101	1845	974	
R	1134	1084	
R1	2404	2017	
R2	2386	1370	
R3	2324	895	905

R4	2333				
R5	2339				
R6	2266	1210	1249	2068	2111
R8	2357	1609	1937	1941	2993
R38	2298	161	2305		
R39	2285	140	142	2296	
R40	2158	2062			
R41	2189	654	1243	1723	
R42	2173	2105			

IPL POST LIST

NAME	LOC.	REFERENCES				
R52	2030	1295	2015			
R73	2278	1203	1385	1888	2324	2340
R81	2125	914	2166	2197	2375	
R82	2141					
R100	2204	1286	1343	1765	1817	
R190	2005	1185				
S1	138					
S2	114	2825				
S3	185	156				
U1	25	157	345	396	424	439
U2	89	45	106			
U3	94	43	104			
U4	99	56				
U5	86	73				
U6	2219	174	304	693		
U20	2416	712	1157	1212	1251	1615 1947 2070 2113
X20	164	158				
Z	720	115	2458			
Z1	800	120				
Z2	827	121				
Z3	917	122				
Z4	945	123	549			

25	297	128		
26	1078	133	622	639
27	748	116	471	475

IPL POST LIST

NAME	LOC.	REFERENCES
28	1024	129

END IPL PCST LIST

APPENDIX C
NARRATIVE DESCRIPTION OF MARK IV, MOD 2

- C11 Control Contexter Program is to decide what type of a message is coming in and assign it to the proper decoder. Receive message from experimenter (automatic copy of input) and save the message. Determine the message type and save its value. Use the value of the message type to select a proper contexter. Execute the selected contexter.
- C50 Include problem details in dummy description list. (The message was a problem-detail type.) Save the message (automatic extract name of dummy description list from message list body) information. I shall *call* the problem _____. I shall call the external world _____. Describe the problem by my name for the external world. The structure of the relationship between dimension values and dimensions of the external world are given by M13. Describe my name for the external world by this structure. Describe the problem by the problem details which I had saved.
Is this the first problem solution attempt? If yes, I shall call my strategy _____. Describe the problem by the name of the strategy. If no, recall the name of the most recent strategy used (obtained from long-term memory). Describe the problem by the name of the strategy. Use the name of the problem as a memory tracer.
- C51 The message is a doing type. Save the message information (automatic extraction of ddl from message body). Get the description of the problem which names my strategy. Create a name for this behavior. Describe the name by the message information. Add the name of this behavior to the strategy.
- C52 The message is a designation type. Save the message information (automatic extraction of ddl from message). The possible values of the experimenter's designation replies will be yes or no. Add this information to my general store of information.
- C60 Recall my most recent strategy and recall the skeleton outline of the current problem. Is this the first problem?
No—execute the most recent strategy, then remember it.
Yes—create a name for the new strategy and save the name. Create linkage to beginning of this phase and save linkage. Now I must compare desired behavior with existing behavior and fill in the strategy. To do this I must (*Sub Process*).
Sub Process
Compare descriptions of behavior on skeleton strategy with existing behaviors. Can I find a match?
No—error.
Yes—save name of behavior, add name of this behavior to the new strategy list.
Does this behavior receive information from the external world?
No—(Go to 9-4).
Yes—Is external information a designation?
Yes—Determine type of designation.
No—Determine what type of external information is received.
Now that information type has been determined, I must get contexter for that type. Add name of contexter to the phase (automatic next upper linkage insertion). (Insertions of contexter implies end of phase so that the behavior can be executed.) Now

execute the behavior phase just created. Save results of decision (all phases end with a decision output of yes or no). Save name of phase just executed. Was the last behavior on the phase list a decision-type behavior?

No—Exit sub-process.

Yes—Link the no response to the first phase. Link the yes response to the correct phase. Create new yes link.

Was result a previous decision a

No—Execute phase linked to the No.

Yes—Exit from sub-process.

[9-4] Is behavior-type a decision?

No—Exit sub-process

Yes—Has phase already been executed?

[Note either a contexter or a decision, continue implies phase execution.]

No—(Go to phase execution).

Yes—Execute just the decision behavior, erase temporary storage.

Go to 9-16.

C20 Create Z7

Information has been received from the external world. I must save where I am in the problem sequence so that I can return to this point later. Is the next behavior in the plan described like a Z7 behavior?

Yes—Exit this contexter.

No—Create a name which is described the same as Z7.

Place this name in the saved problem sequence. Use name of the present contexter to get name of phase, save the name of the phase. Get name of Z7 behavior. Add behavior which creates a working hypothesis. Add behavior to remember the working hypothesis (P191). Add behavior which will trace working hypothesis (C31). Add behavior to remember how the working hypothesis was formed. Add a decision behavior which decides if I can proceed to the phase (D4). Describe the phase to show that Z7 has been added to the phase. Describe the phase to show that D4 has been added to the phase.

C36 Verify subject choice.

Save where I am in the problem solution. Obtain next behavior in the sequence. Is it described as a D-type routine?

Yes—Exit contexter.

No—Create a name for a behavior which is described as a DO routine. Add the DO behavior to the phase list (automatic next upper added to DO). Describe the phase list by the addition of DO.

C38-9 Create reaction to object designation. Save where I am in the problem sequence. Save the current memory entry point. Use name of contexter to obtain name of phase. Save the phase name. Create and save name for a routine described the same as a Z4 routine. Is the next behavior in the solution sequence described the same as Z4?

Yes—Delete the matching name from the problem sequence.

No—Add symbol for Z4 behavior to the phase list.

Describe phase list by addition of Z4 behavior.

Get experiment's designation of the object choice. Was it designated a yes?

Yes—Establish a routine to remove irrelevant dimensions, as the appropriate behavior, save name, (Go to 9-4).

No—Establish a routine to revert dimensions as the appropriate behavior, save name.

Is the subject aware of dimension varied?

No—(Go to 9-4).

Yes—Was one dimension varied?

Yes—(Go to 9-4).

No—Add routine to retrace memory one level to the Z4 routine. Then exit contexter.

9-4 Add to Z4 a behavior to retrace memory one level. Add to Z4 a behavior to mark the dimension values. Add to Z4 a behavior to mark the dimensions. Add a routine to recall the experimenter's designation of the object. Add appropriate dimension handling routine to Z4. Exit contexter. [Note: This routine is not consistent with the other routines.]

[Note: The interpreter must check for contexter and if so will put CIA in (0), along with other inputs.]

C22 Set up reaction to concept designation.
 Save where I am in the problem sequence, obtain the next behavior in the problem sequence and determine if its designation matches that of a Z6 behavior.
 Yes—Delete behavior from problem sequence.
 No—Obtain experimenter's designation of the concept.
 If it is a
 Yes—Exit from contexter.
 No—Create a symbol described as a Z6 behavior.
 Add new Z6 to the execution list.
 Use name of contexter to get name of phase.
 Describe phase by addition of Z6 behavior. Get the focus object and determine if any unmarked dimensions exist.
 Yes—Set to add routine to which finds unmarked dimensions.
 No—Set to add routine which finds irrelevant dimensions.
 Save name of appropriate routine.
 Add routine to Z6 which retraces memory one level (C41). Add routines to retrace memory one level. Add appropriate routine from above (Q41, Q42). Add routine to add dimension to working hypothesis. Add routine to Z6 which remembers modified working hypothesis. Is the subject aware?
 Yes—Correct focus object dimensionality.
 No—Set dimensionality to unity.
 Set routines which add dimension to above dimensionality. Exit contexter.

Z0 Problem specification
 P21—Copy focus.
 P61—Remember copy of focus object.
 P31—Put name of focus copy in MEP.
 P62—Remember set membership of focus copy.

P21 Copy focus object.
 Get the name of the focus object from working memory, save the name.
 Get a dimension value, use dimension value to get dimension. Use dimension to get the list M13. Make a copy of focus object in DAV order (R190). Give this copy a name, place name in M1,N. Exit subroutine.
 [Note 1. This should be mostly within the subject. I think we should change program so that E routine

gives an undescribed focus with internal name so that subject does not have to copy it.]
 [Note 2. This routine is really unnecessary if we implement Note 1.]

P60-1 Attach (M1,X) to value list of class attribute (2) of list (3) (0) = Flag.
 Input M1 3
 Output M1, A220.
 Note that whatever is named in contents of MEP gets described. Remember _____. The item of information in working memory is stored as a value if an attribute describing the symbol contained in the memory entry point.

C31 Trace
 The name contained in working memory is placed in the memory entry point so that one can trace the problem path in memory.

P62 Remember _____. Remember that the focus object is a member of the set, i.e. the ddl in working memory is stored as the value of an attribute describing the focus object.

Z7 Create a working hypothesis
 P191—Create working hypothesis from the copy of the focus object (CFO).
 P63—Remember working hypothesis.
 C31—Place WH in MEP.
 P64—Remember how working hypothesis formed.

P191 Make a working hypothesis having K99 dimensions. If the subject is not aware, this is a subject characteristic routine which creates the WH.
 If the subject is aware, this routine is a legal doing-type routine.
 Get the focus object from MEP, create a copy. Remove those dimension values beyond the K99th. Find the dimensions corresponding to these dimension values. Create a description list containing these dimensions. Save the ddl. Attach the dimension values under a "from" attribute. Store ddl in M1,D. Store WH in M1,N. Exit subroutine.

P63 Remember working hypothesis.

C31 Place name of working hypothesis in memory entry point.

P64 Remember how working hypothesis formed.

- Z1 Create search criterion.
 P131—Select dimensions to vary.
 P141—Select dimension values.
 P151—Vary dimension values.
 P64—Remember how working hypothesis was varied.
- P131 Select dimensions to vary.
 Get the name of the WH from working memory. Collect dimension values not designated and save this list. I will vary only K98 of these so discard the rest.
 Make a list of the dimensions corresponding to the dimension values. Create a ddl for the dimensions. Add the list of dimension values to the ddl under a "from" attribute. Put name of WH in M1, N. Put name of ddl in M1, D. Exit from routine.
- P140 Find new dimension values.
 Get the list in M1, D describing what to vary, and save its name. Obtain value of the "from" attribute. Obtain the dimensions attribute. Generate dimensions and their values in parallel.
Sub Process. I know the dimension to vary and its original value. Obtain list of dominance values for the dimension. Set probability of dimension value negative (ignore dominant attribute value of this dimension value). Choose most dominant dimension value. If none, exit. Save this dimension value. Check to see if it has been designated.
 Yes—Pick up another dimension value.
 No—Make this DV a value of the "to" attribute of the ddl saved earlier. Restore DAV values of the "from" and "to" dimension values (seems peculiar should be in Z6 problem clean-up). Exit sub process.
- P151 Vary the dimensions of the working hypothesis. Save the name of the working hypothesis. Get the ddl containing the change information and save its name. Get the list of "from" dimension values. Get the list of "to" dimension values. Generate the "from" and "to" dimension values in parallel
Sub Process Get the name of WH, then replace "from" DV to the "to" DV

[Note: Contents of M1 before P151 were WH and ddl, P151 manipulates the contents of WH but does not replace the contents of M1. Perhaps we should standardize the end of a routine to fill M1, N, M1, D.

- P64 Remember how WH modified. Describe working hypothesis by the "from-to" list.
- Z2 Select an object.
 P51—Find object matching search criterion. [Note: the search criterion is the working hypothesis after a dimension has been varied.]
 P65—Remember the object.
 C31—Place name of object in MEP.
 P66—Remember how object found.
- P51 Save the inputs. Get and save the name of the search criteria. Search the external environment for object matching the search criterion (K2). Save the name of the matching object. Get value of change attitude ("from-to" list). Place name of change list in M1, D. Place name of object found in M1, N. [Note: We may want to systematize this by always using *name* for those symbols going into M1, N and *descriptions* for those symbols going into M1, D.]
- D0 Verify object choice
 Q51
- Q51 Check object choice.
 Save inputs. Get name of object chosen. Recall the focus object. Make list of symbols. Determine which dimension values of the object are not on the focus object. Save name of this list. Get "from-to" list from the "how found" attribute. Get the value of the "to" attribute. Find which values the focus object and "to" list do not have in common. Generate these values.
Sub Process Check to see if dimension values are marked.
 No—Exit.
 Yes—Pop H0.
 Set output of Q routine ±.
 [Note: This program determines what focus object and card choice do not have in common. Then it checks to see that the non-common dimension values are not already marked relevant or irrelevant. If unmarked, this object is O.K.]

Q101 Get name of working hypothesis.
Generate body of WH.
Sub Process Get value of designation attribute of the dimension value.
If unmarked—exit.
If marked—to next dimension value.
If all dimension values marked, then concept can be presented.

Z5 Form a concept.
P121—Form concept.
P67—Remember concept.
C31—Transfer concept.
P68—Remember how formed.

P121 Form concept.
Get the search criterion (WH) and save it. Create name of concept. Create descriptions of concept and save it. Generate body of working hypothesis. (Sub process)
Sub Process Save dimension value, determine if dimension is unmarked. If yes, exit. If no, save value of relevancy attribute and save dimension value. Was dimension value relevant?
If yes—add value to body of concept.
If no—exit sub process.
Make working hypothesis the value of the basis attribute of the ddl. Place description in M1,D, place concept in M1,N.

Z8 Have concept designated.
P72—Present concept to experimenter.
E94—Experimenter action.
P69—Remember designation.
[Note: P72 is definition of present verb.]

Z6 Corrective action:
C41—Pop SC to WH.
C41—Pop WH to CFO
Q41—Find untested dimensions (or Q42)
P181—Add dimension.
C31—Place working hypothesis name in MEP.
P64—Remember what added.

Q41 Find untested dimensions.
Get and save copy of the focus object (in MEP). Make list of all dimension values of CFO not marked *DELETE* all but K97 of the symbols. Find dimension corresponding to unmarked dimension values. Create description. Make list of dimensions the value of the dimension attribute (what). Add list of unmarked dimension values as value of the "to" attribute of the ddl. Place name of CFO in M1,N. Place description in M1,D.

Q42 Same as Q41 except that R42 used instead of R41. R42 checks for match of both attribute and its value, hence to list irrelevant dimension values one needs attribute value.

P181 Add dimensions to SC (WH in M1,N, new DV list in M1,D). Get value of hypotheses attributes of CFO and save it (WH). Hold it in M1,N. Get the description, obtain the "to" list from the description. Generate symbols on "to" list.
Sub Process Add symbol on "to" list to the working hypotheses. Place working hypotheses in M1,N. Place description in M1,D.