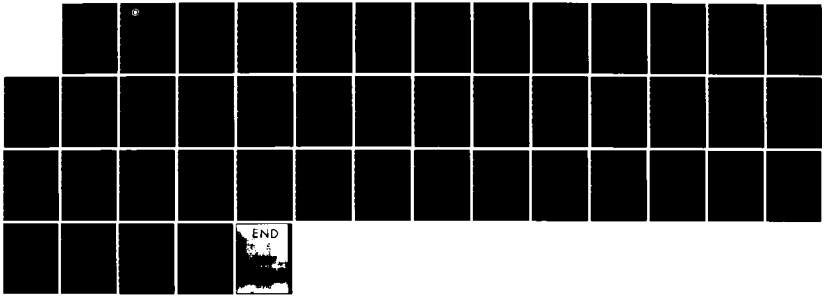
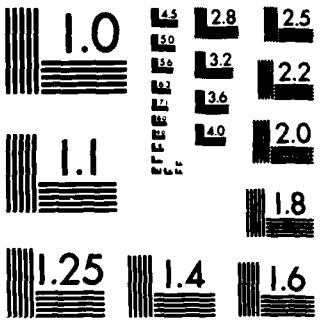
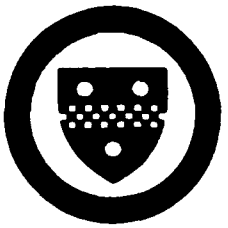
AD-A144 638 SPIRIT: AN EVOLUTIONALLY DESIGNED INTELLIGENT TUTORING SYSTEM(U) PITTSBURGH UNIV PA LEARNING RESEARCH AND DEVELOPMENT CENTER H E POPLE JUL 84 1/1

UNCLASSIFIED UPITT/LRDC/ONR/APS-15 N00014-82-K-0613 F/G 9/2 NL







University of Pittsburgh
LEARNING RESEARCH AND DEVELOPMENT CENTER

SPIRIT: AN EVOLUTIONALLY DESIGNED INTELLIGENT TUTORING SYSTEM

Amos Barzilay[1]
University of Pittsburgh
and
XEROX Palo Alto Research Center

Harry E. Pople, Jr.
University of Pittsburgh

July 1984

Technical Report No. UPITT/LRDC/ONR/APS–15

Preparation of this report was sponsored by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract No. N00014–82–K–0613, Contract Authority Identification Number, NR 154–497.

This report is issued by the Learning Research and Development Center, supported in part as a research and development center by funds from the National Institute of Education (NIE), United States Department of Health, Education, and Welfare.

[1]Now with Syntelligence, 800 Oak Grove Ave., Menlo Park, CA 94025.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER UPITT/LRDC/ONR/APS-15	2. GOVT ACCESSION NO. AD-A144636	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SPIRIT: An Evolutionally Designed Intelligent Tutoring System		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Amos Barzilay and Harry E. Pople, Jr.		8. CONTRACT OR GRANT NUMBER(s) N00014-82-K-0613
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Pittsburgh Learning Research and Development Center Pittsburgh, PA 15260		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Prog. Ele.: 61153N(42) Proj. #: RR042-06-0A Task #: NR 154-497
11. CONTROLLING OFFICE NAME AND ADDRESS Personnel and Training Research Program Office of Naval Research (Code 442PTJ) Arlington, Virginia 22217		12. REPORT DATE July 1984 13. NUMBER OF PAGES 30
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Intelligent tutoring system; instructional dialogue; expert system; student model.

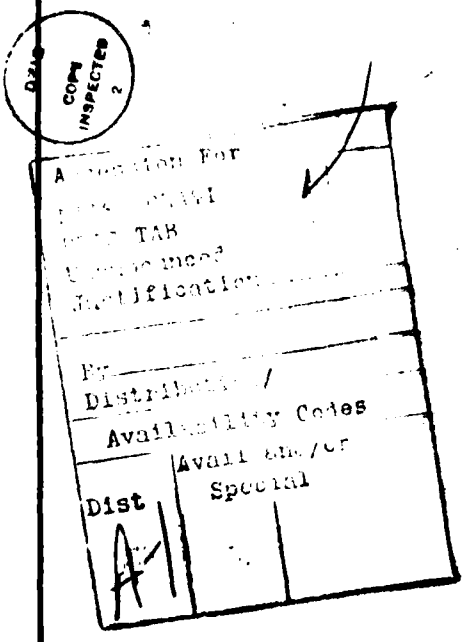
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

SPIRIT is an Intelligent Tutoring System for tutoring probability theory which has evolved through a continuous process of experimentation and tuning. The system manages a unique flexible tutoring style. On one hand, the system may behave as a tutor who mostly observes the student without interference, intervening only when things are really going wrong and on the other hand, it may behave as a tutor who manages a " questioning and answering" type of dialogue. Based on a belief constructed about the student's aptitude, the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

system frequently changes its tutoring style. SPIRIT integrates several artificial intelligence methods that include: a theorem prover; a production system; an object oriented system and procedural knowledge embedded in LISP code.



SPIRIT: An Evolutionally Designed Intelligent Tutoring System[1]

Amos Barzilay[2]
University of Pittsburgh
and
XEROX Palo Alto Research Center

Harry E. Pople Jr.
University of Pittsburgh

ABSTRACT
SPIRIT is a an Intelligent Tuturing System for tutoring probability theory which has evolved through a continuous process of experimentation and tuning. The system manages a unique flexible tutoring style. On one hand, the system may behave as a tutor who mostly observes the student without interference, intervening only when things are really going wrong, and on the other hand, it may behave as a tutor who manages a "questioning and answering" type of dialogue. Based on a belief constructed about the student's aptitude, the system frequently changes its tutoring style. SPIRIT integrates several artificial intelligence methods that include: a theorem *prover, a production system, an object oriented system and procedural knowledge embedded in LISP code.*

1. Introduction

Although much progress has been achieved in the last decade as people have started to apply AI methods to the design of tutoring systems, tutoring systems today are far less competent than the experienced tutor. One of the reasons for this is the lack of a well founded theory of teaching. As Sleeman and Brown ([1], p.3) put it:

> The tutoring strategies used by these systems are excessively ad-hoc reflecting unprincipled intuitions about how to control their behavior. Discovering consistent principles will be facilitated by constructing better theories of learning and mislearning...

Moreover, because of the complexity of the design of Intelligent Tutoring Systems, AI workers have often focused on one or two aspects in Intelligent Tutoring System (ITS), rather than developing a complete system that does a satisfactory tutoring job [2]. For example, Brown and Burton [3] in DEBUGGY and Sleeman and Smith [4] in

---

1. This research has been supported in part by the Office of Naval Research Grant No. SFRC N00014/82/k/0613.

2. Now with Syntelligence, 800 Oak Grove Ave. Menlo Park, CA 94025.

LMS focused on student modeling. Clancey [2] focused both on the tutoring knowledge and on student modeling but his system used MYCIN's [5] knowledge as its domain knowledge, which was not specially built to serve tutoring purposes.

The underlying approach of this research is that since we do not have a good theory of learning, an effective ITS should evolve through a process of experimentation and tuning. What the system should do can best be identified by experimenting and using the system. Therefore, the designer's focus should not be on the effectiveness of the various componets such as student modeling, tutoring strategies or domain expertise as stand alone parts but on the effectiveness of the system as a whole.

This approach was used in the design of SPIRIT; an ITS for tutoring probability theory that has evolved to be a successful system. During the process of developing the system the designer's focus often shifted from one aspect to another as deemed appropriate based on experiments that were continuously conducted with the system. The system integrates various AI methods such as a theorem prover, a production system, an object oriented system and conventional LISP programming; each method was selected to do the task for which it is best suited. All together the methods do a complete task.

2. The task: tutoring probability theory

SPIRIT tutors:

- Elementary concepts of probability theory and formulation of probability problems.
- Basic probability rules: multiplication and addition rules, conditional probability, rules of intersections and unions.
- Special cases of probability rules: independent events, mutual exclusive events, marginal probability.
- Bayes' rule.

A student interacting with the system is assumed to be familiar with the basic concepts of probability theory. Word problems in probability are posed to the student, and the system follows and guides the student while he is solving the problems, focusing on the recognition of the concepts presented by the problem and on the application of probability rules in problem solving.

The first stage of the research with SPIRIT was to investigate the behavior of the experienced human tutor in tutoring the relevant subset of probability theory. In preliminary experiments five tutoring sessions were recorded and analyzed. Most of the tutoring principles that were identified at that stage were implemented in a prototype of SPIRIT. However, through the process of experimentation with SPIRIT the system has been changed very significantly. Moreover, tutoring principles used by the human tutor in the preliminary experiments which were ignored in previous analyses became noticeable as more experiments were conducted with the evolving system. Before this evolution process will be described, SPIRIT's architecture will be reviewed.

3. SPIRIT's architecture

The major components of SPIRIT are depicted in figure 1. In the following, I will briefly describe the function of each component.

INSERT FIGURE 1 HERE

Tutoring expert

The tutoring expert manages the dialogue between the system and the student, and makes most of the important decisions of what tutoring actions to take. There are two different styles of tutoring which the system uses; *tutor* and *mentor*. The *tutor* manages

a dialogue in which the student is strictly guided and is told exactly what the next move is. It first goes over the natural language problem text, possing to the student the important sentences and asking him to represent each sentence using probability notations.This process is called working in the "symbolic context". After all the important information is extracted from the problem text , the *tutor* guides the student along the solution process. This process uses an AND/OR tree generated by the probability expert which will be described below. If the student wishes, he may jump ahead and attack any relevant intermediate problem. However, at each point along the dialogue managed by the *tutor*, there is an enforced "understanding" between the system and the student about what to do next. The other style of tutoring is the *mentor*, which manages quite a different style of dialogue. The *mentor* usually does not tell the student what to do. The student uses the terminal as scratch paper. He can type whatever he wants. He may start by expressing symbolically the information given in the problem text as well as writing some formulas or algebraic expressions. The *mentor* analyzes each line the student types. From time to time the *mentor* may decide to intervene. It may correct mistakes, encourage, or transfer control to the *tutor*. Between the *tutor* and the *mentor* there is a smooth two-way interaction. The student cannot always tell with which component he is interacting. The system behaves somewhat as a human tutor who changes his style of tutoring on the basis of the circumstances. A summary of all that happens along the tutoring session is maintained and is used by the *tutor* and the *mentor*. In addition, some strategic decisions are made on the basis of a deeper analysis of the student which is done in the Student Model. One important decision may be to transfer control to one of the specialists (which will be described below) that can handle in depth a basic student's misconception.

Problem Model

Probability problems to be presented to the student in the tutoring process are represented within SPIRIT by means of a problem model. The problem model is built by the teacher. The events are described and the information needed with respect to a

problem is recorded in this model. For example, the problem model would specify if events are independent, mutually exclusive or exhaustive.

The probability expert

The probability expert can solve probability problems within the domain covered by the system. However, before the expert can solve a problem all the events referred to in the problem must be identified, and the special cases for which probability rules are extrapolated should be explicitly described. For example, the probability expert must be told if two events are independent because it can not deduce this information from the natural language problem text. The conceptualization used by the expert component is that of backward chaining through formulas. That is, first the formula that yields the final answer to the problem is constructed. Then, for each argument in this formula the probability expert attempts to find a formula that may be used to calculate the argument. The process terminates when reaching terms for which numbers given in the question can be substituted directly. This was the prevailing conceptualization used by students in the preliminary experiments. The expert produces an AND/OR tree by chaining the probability rules in the backward direction. Then the tree is 'solved', meaning that the desired answer in the root of the tree is 'proved'. This is done by propagating the numbers in the leaves of the tree upward until the final answer is constructed. This process is based on the classic notion of Backward Deduction System as described in [6]. The AND/OR tree is often helpful in allowing the system to follow the student's reasoning.

Error Analysis

The purpose of this component is to identify students' systematic mistakes and to find the misconceptions hidden behind the mistakes. This is the same objective underlying the development of DEBUGGY [3]. The error analyzer is composed of several LISP subroutines, each subroutine corresponding to a probability rule. Each subroutine attempts to map the formula the student wrote to the correct formula. The

discrepancies are identified and classified to several types corresponding to the assumed misconceptions behind the mistake. An effort was made to account for the vast majority of mistakes a student may commit. After the error analyzer identifies the mistake the student has made, a message is sent to the tutoring expert. Then, the tutoring expert can make the decision whether or not to intervene immediately to correct the mistake, and if so in what way.

Student Model

The student model analyzes the student's performance. It makes assumptions about the student's overall capability and accumulates knowledge about the student's performance with respect to various subskills the student is expected to acquire. The system makes an assumption about the capability of the student because not all students should be handled alike. A more competent student receives more complex explanations, and interacts mostly with the mentor. A weak student receives simplified explanations and interacts mostly with the tutor.
The system's beliefs are continuously revised and changed based on the circumstances. The student model is accessible to the tutoring expert and to the specialists which are described below, and it influences decisions made by these components.

Specialists

From time to time the tutoring expert may decide to treat in more depth a student's fundamental misconception that was revealed in the problem solving process. This is done by transferring control to a specialist. The specialists are LISP subroutines that interact with the student. A specialist queries the student model so as to teach the student in a suitable way. The information a specialist uses include: what the student has done before and its current belief about his capabilities. Some of the specialists that are currently implemented handle the following issues: independence, marginal probability, conditional probability and the union rule. The specialists may present

Venn diagrams, may pose questions and may relate things done previously to the current problem.

The Interface

The purpose of the interface is to analyze the student input, to parse it and to decide what the student intends to do. Sometimes the student is asked to do a specific thing. This usually happens when he interacts with the tutor. In this case, the system knows what to anticipate. When the student interacts with the mentor, a line typed by the student is handled by a component that makes assumptions about the student's intentions. The various things that a student may attempt while interacting with the tutoring expert are: expressing data symbolically, writing a formula, substituting numbers in a pre-written formula, writing an algebraic expression and writing a numerical value. In addition the student may request help either because he is confused or in order to solve an intermediate subproblem. He may also ask for information about either the probability values found or the formulas written up to this point. In addition, the terminal can be used as a calculator at almost any point in the dialogue.

4. The implementation framework

Since it was clear at an early stage of the research that SPIRIT would evolve and change, it was extremely important to choose an appropriate tool to facilitate the building and the refinment of the system. The component most naturally subjecte to changes and tuning is the tutoring expert. Tutoring knowledge has been often represented in explicit rules (e. g. Clancey [2], Oshera [7], Collins [8] and Lantz et al. [9]). There are some rule-based tools (e.g. EMYCIN [10]) that could have been used in SPIRIT. OPS5 [11], however, seemed to be particularly appropriate for the design of an ITS. OPS5 has the advantage possessed by most rule-based tools; rules are relatively independent of each other, and changes in one rule do not necessarily force changes other rules. As a result, the system's behavior can be changed relatively easily. In

addition to this advantage OPS5 has two other important advantages for the design of an ITS. First, the principles of conflict resolution strategy used in OPS5, specificity and recency suit very well (as will be demonstrated below) the tutoring domain. Second, the structure of working memory elements that are accumulated in the working memory, and of production rules which are sensitive to certain elements corresponds nicely to the structure of a tutoring session. Working memory elements may correspond to elementary pieces of interaction and represent the elements of knowledge the human tutor acquires in a tutoring dialogue. The production rules may correspond to various tutoring decisions that ought to be made in certain scenarios presented by the working memory. Thus, OPS5 provides a natural way of representing the dynamic knowledge accumulated during the dialogue and the static tutoring knowledge that corresponds to the human tutor's experience. Moreover, OPS5 is a tremendously efficient tool that can reduce system's response time which is a major factor for success in a highly interactive system such as an ITS.

In SPIRIT, productions correspond to tutoring decisons which manage the dialogue. Each working memory element describes an elementary interaction, and indicates what the student did and how the system responded. A move taken by the student is described by a working memory element which indicats what is the move, whether or not it is a correct move and if not what type of mistake has been made. After the system takes a tutoring action a working memory element which describes this action is introduced into the working memory. With respect to each tutoring decision, there are usually several productions, arguing for different tutoring actions. These productions actively compete for the right to fire (i.e. for the right to execute their action part). The conflict resolution strategy employed provides an intuitively appealing approach for resolving these conflicts.

5. SPIRIT's evolution

Because SPIRIT has continuously evolved, and was changed almost every day, there are no versions or distinct stages in the system's life cycle. However, this section divides the evolution process into three stages, for purposes of exposition. The first stage is called the base system and includes only the "tutor" tutoring style. The second stage includes both the tutor and the mentor partially integrated. The final system is the complete tutor-mentor integrated system including student's modeling and specialists.

5.1. The base system - follow the tree

The base system can be viewed as a combination of two sub-systems, called "symbolic" and "solve". The symbolic component sequentially presents to the student segments of the probability problem, each of which is a part of the text that expresses a probability concept. The student is asked to represent each segment by a probability expression. After the student has completed expressing symbolically the probability word problem, he moves to the solve stage, where he applies probability formulas.

The approach used by the base system is that of questioning and answering. Every system response includes a question and every student response includes an answer to a specific question. After all the segments have been symbolically represented the system starts the process of following the AND/OR tree, which is strictly a backward chaining approach. The major goal that drives the system is that of finding the solution to the probability problem, (rather than instructional goals). Although this approach may appear relatively simple, the base system uses several important tutoring strategies that were identified in the preliminary experiments involving some design complexities. The interesting issues are related to decisions determining what is the proper explanation, when to provide it, and when to probe the student. The extensive analysis of mistakes done by the error analyzer also contributed to the system's effectiveness.

Some of the important tutoring strategies used by the base system were the following:

1) Probe the student. By "probing" we mean pushing the student to be active and to do things even when he hesitates or prefers the system to tell him what to do. The system does not provide help automatically whenever help is requested. When the student types a "?" there are usually several productions whose left hand side (LHS) are satisfied, each arguing for a different tutoring action. Those with a more specific LHS have a stronger argument for providing substantial explanation. The production whose LHS is more specific would be satisfied only when there has previously been some action with respect to the current intermediate problem, so more substantial help should be provided. The production whose LHS is less specific would be fired only in the cases where the more specific productions are not satisfied, and not much has been done (if anything at all) that pertinent to the current intermediate problem. As a result, a student asking for help immediately, without making any effort first, would be probed, while a student who has been struggling for some time with the current intermediate problem, or who has already asked once or twice for help, would receive more extensive help. When the student asks for help for the first time after an intermediate problem is posed to him, the system always asks the student to make some effort and to type something, even if it is just a guess. The first time in the dialogue that the system probes the student, it explains that even though the student is not confident of his answer, it is better to guess so the system will have some knowledge about his thinking. In most cases, the base system provided substantial help after three repetitive help requests, or after a help request which was preceded by one or two attempts. As will be discussed in a subsequent section, this behavior of the system was often inappropriate.

2) Make the student find the answer by himself. The system attempts to provide minimal help while still making sure that the student is progressing. This is done by choosing and providing the appropriate explanation. When the student makes a mistake, the system first responds by showing him why his answer is incorrect. After detecting repetitive mistakes the system begins to address the correct answer more

explicitly in its explanations. Although these become more and more elaborated, only in the extreme case is the answer provided. This strategy enables each student to use just the help he needs, not more nor less. An example of this strategy is the following. Suppose the student is expected to write the formula $p(A)=p(A \cap B)+p(A \cap -B)$, and in his first attempt he writes the formula $p(A)=p(A / B) + p(A / -B)$. The system's response would be the following: "You can not add conditional probabilities like this. You may get a larger than 1 probability. Try to think about a more basic formula which does not involve conditional probabilities. Try again to write a formula for p (A)." . This explanation says very little about the correct formula. If in the next attempt the student is wrong again the system's response would be more to the point. For a typical mistake, the system is capable of providing explanations in about three to four levels. The base system always provides first the shallowest explanation and then the more elaborated ones. The vast majority of the explanations provide two kinds of information, information about what is wrong in the recent mistake and information about the right answer. The shallower explanations focus on the recent mistake, while the more elaborated ones focus on the correct answer. A similar tutoring strategy was used in the WEST system [12] where four successive levels of hints where used. The first addressed the student's weakness and successive levels increasingly addressed the optimal move to be taken.

3) Relate things to what has been done previously. This strategy is used in two cases: (1) when a student reveals that he has a misconception which had been identified and dealt with before, (2) when a student reveals a misconception, although previously he did something suggesting that he did not have this misconception. Each time the student either makes a mistake or requests help, the system examines to see whether or not the concept raised by the current intermediate problem has been discussed or dealt with before. If the system identifies that the issue has been discussed, it refers the student to the appropriate intermediate problem that raised this issue. In the base system, issues are discussed only when the student makes a mistake or requests help related to the issue. This is in contrast to the way the principle of relating things was implemented in GUIDON [2]. GUIDON encourages the student when he properly

applies a domain rule that previously he did not know, but the case of a student failing to apply things that previously he knew, is not handled explicitly. An example of this strategy in SPIRIT is the following: Suppose the student has difficulties in expressing conditional probability, but he has correctly expressed conditional probability in another segment. The system in that case would refer the student to the segment that he has correctly expressed, and it would indicate that the two segments (the one done before and the current one) have the same structure. An example of this strategy in the solve stage would be referring the student to an intermediate problem in which he has applied the probability formula that is currently required. The productions that implement this strategy have priority over other productions that provide explanations or probing statements. Therefore, the system always prefers to refer the student to what he has done, rather than to take other tutoring actions.

4) Encouragement strategies. Several strategies that are not dependent on extensive student modeling were implemented in the base system. Negative feedback is avoided as much as possible. The system does not use statements such as "wrong" or "incorrect". Instead, the response to a mistake is always a constructive statement that tells the student what is wrong with his answer, or provides him with some hint about the correct answer. Positive feedback, on the other hand, is given whenever appropriate. The student is given the feeling that he has found the answer by himself even when he has received substantial help from the system.

5) Generalization strategies. The process of following the tree is explicitly described during the dialogue, thereby demonstrating and teaching the backward chaining approach to problem solving. Explanations are provided in general terms using the abstract form of formulas (using the symbols A or B instead of the current events' names in the specific problem).

Problems in the base system

The base system was tested by several subjects that expressed more or less the same

concerns about the system's performance. Subjects complained that while interacting with the system they lost the whole picture and they felt as though they were being quizzed. That is, the tutoring session seemed to them as one in which the tutor asks a series of unrelated questions. The subjects who could have performed better were prevented from solving the problem by skipping some of the stages or implicitly doing some intermediate moves. They also expressed dissatisfaction because the decision of what to do next was always being made by the system. Making this decision, they felt, was an interesting challenge, and so they easily got bored and did not express any desire for a long engagement with the system.

The subjects who did not perform well also felt as though they were quizzed, but did not mind the system deciding for them what to do next; their major problem was related to the interface. The strategy of "probe the student" very often tended to humiliate the weak subjects. Subjects generally engaged in a long process of thinking before they requested help. Even remarks from the experimenter who reminded them that a help facility was available did not change their hesitant behavior significantly. When a subject eventually requested help, the system's response often told him to try again or to give his best guess. This of course might be very discouraging.

The interaction with the system was done using a screen terminal and students used a text editor (a full screen editor) to scroll back and forth in case they wanted to see things done previously. This was a burden for all the students who used the system.

5.2. Second round: Adding the mentor

The concerns that subjects expressed with respect to the base system's approach seemed to be prominently related to the fact that the base system did not address the student's need for strategic and planning knowledge in problem solving ([13], [14]). The approach that was adopted in the second round was based on the assumption that a student should first know the general concepts of probability theory, and how to make probability inferences with this knowledge before he can sharpen his strategic

knowledge. While the tutoring of knowledge of concepts and of making inferences was believed to be done quite satisfactorily in the base system (excluding the interface problem), the system needed a new approach for handling the tutoring of strategic knowledge. This new approach was implemented by the *mentor* tutoring style.

The idea was that a student should be allowed to select his moves as he wishes, thereby practicing his strategic knowledge and revealing his ability to the system in this respect. The implementation of this idea in a tutoring system is a novel aspect of SPIRIT, undertaken to approach more closely the human tutor's behavior, as reflected in the tutoring protocols. The major theme of the second round was therefore to use the base system's approach for the students who do not have the knowledge of the basic probability concepts and rules, and to use the mentor approach for those who have the basic knowledge but need tutoring focused on strategic knowledge.

In the second round there were two systems; the tutor and the mentor. These two systems were not mutually exclusive; rather, the tutor was a subset of the mentor system. The tutor as an independent system was similar to the base system with some upgrading as described below. The mentor used the tutor as a sub-system. A student when starting to interact with the system had the choice between the two systems, and had only limited option to switch between them making a preliminary choice.

Improving the tutor

The strategy of probing the student was modified so that the student would not feel humiliated. The productions that argue for providing more elaborated explanations received more priority in conflict resolution. The system was changed so that the student would never be asked to try again more than once while working on any one intermediate problem. Also, the revised system always provided meaningful help if the student's help request came after some effort had been made with respect to the current intermediate problem.

The strategy of probing the student was still used by the system, but it was changed to be applied in a more moderate and graceful way. In order to alleviate the problem of getting lost as mentioned earlier, two new commands were made available to the student. The command *data* displays all the probability expressions as well as their values, that have been defined by the student. This command freed the student from the necessity to scroll back and forth in order to see the numbers that were given in the problem text. The command *formula* has the same purpose, but with respect to formulas that have been written up to the point where the student issues the command. The command *formula* displays all the correct formulas that the student has typed in. The tutor system behaved like the base system with the changes mentioned above.

The mentor

In the mentor, the problem text is presented to the student who is asked to solve it using the screen the same way he would use scratch paper. The mentor does not force the student to start the problem solving process in the symbolic stage nor to adopt the system's backward reasoning approach. Instead, the student makes his moves while the mentor analyzes his progress, and whenever appropriate it intervenes by taking some tutoring action. These actions can be performed either by the tutor, which is instructed to assume control temporarily or by the mentor. All tutoring actions (except encouragements) are evoked as a result of mistakes and help requests. Encouragements are provided from time to time as deemed appropriate. Thus, a student who does not make mistakes and does not request help could solve the problem with the intervention of only some system remarks such as "Good" or "Correct". In many cases, a mistake made by the student does not evoke the mentor's intervention. Thus, the mentor uses the strategy of letting the student discover his own mistake. The rationale behind this strategy is that if the system does not correct a student's mistake there is a chance that the mistake would lead the student to a dead end or to a point where the mistake becomes obvious. When the student, himself, identifies that he has made a mistake there is a higher chance that he would remember

not to make this mistake again. The mentor has to make a difficult decision of whether or not to let the student go in the wrong track, and if so, how far. There must be some point from which letting the student go further in the wrong track is a counter productive strategy. Deciding where this point is in each case is a difficult question. The rules that determine this point were constructed on the basis of experiments and on a trial and error process of tuning. There are two classes of cases in which the mentor may decide to intervene: an intervention may be triggered by the student's request for help, or it may be evoked by some strategy that specifies an intervention in a particular scenario reflecting by the various moves that the student has taken.

Interventions triggered by help requests

The mentor always intervenes if the student requests help while he is on the wrong track. When the student requests help and the move preceding the help request included a mistake, an assumption is made that the confusion is the result of the mistake, and the mentor transfers control to the tutor for a discussion related to the last mistake made. The tutor handles this mistake as described in the previous section. After the mistake has been fixed the tutor returns control to the mentor. The principle of selecting the most recently mentioned issue for discussion was also used both in GUIDON [2] and in WEST [12]. In WEST this strategy was called "focus strategy" and was compared to the "breadth strategy" which selects for discussion an issue that has not yet been discussed. When the student asks for help at the very beginning of the problem solving process, the mentor gives the student an opportunity to switch back to the tutor. The mentor also recommends in this case to start the problem solving by expressing the data symbolically. A help request which is made neither after a mistake nor in the beginning of the problem solving process is understood by the mentor as if the student were saying "I don't know what to do next". In this case the mentor looks to see what is the most recent thing that the student has done. Using the AND/OR tree the mentor then recommends what the backward reasoning approach would dictate to do next.

Sometimes the student may need help in finding some particular probability expression, which he believes to be required for solving the problem. He can receive this help by issuing a specific help request, i.e. by typing the probability expression followed by a "?". There are three possible cases. First, the student may request help for finding a probability expression which is irrelevant and is not required. The mentor, in this case, would tell the student that he does not need this expression. Second, the student may request help to find a probability expression for which either its value or a formula has already been found. In this case the mentor would remind the student that the analysis of this expression has already been made, and would also remind him of the commands *data* and *formula*. In the third case the specific help request is relevant, and the mentor would provide some hint about how to find this expression. If the student requests help by typing a "?", following the system's response to a specific help request the mentor would understand this request as saying "This hint is not sufficient for me. Please give me more help to find the probability expression that I asked you before.". Notice that this is a different interpretation for the command "?" than the one mentioned earlier. Thus, the mentor has the ability to give more than one interpretation to a command based on the particular context in which the command is issued.

Interventions which are not triggered by help requests

The mentor provides a mechanism that enables the implementation of strategies of the form: " if a student has taken moves A, B, C, etc. then, do the following...". Throughout the process of experimenting with the mentor, such strategies can be easily added and changed. An example of such a strategy is the following. If the student has more than one mistake in expressing conditional probability, the mentor would intervene and correct the student's related mistakes, thereby demonstrating to the student that he had a misconception that resulted in these mistakes. Some mistakes may evoke immediate intervention. An example of such a mistake is the mistake of confusing algebraic and set operators. This mistake triggers an intervention because

the underlying misconception may cause the student to use improper notation not understood by the system, thereby inhibiting further analysis of the student's moves. Other interventions are required in order to maintain flow of the dialogue. For example, if the student types in an algebraic expression which is an instantiation of an incorrect formula that was typed in earlier, the mentor would intervene telling the student that the formula he is using is incorrect. The mentor always forces the student to work explicitly, thereby eliminating the possibility of accidently finding the correct answer; this also helps the student acquire the habit of working in a thorough, organized manner. Thus, if the student types in an instantiation of a formula (i.e. the actual numbers are plugged in), and the formula used has never been written before, the mentor would ask the student to write the formula that he is using. This is done whether the implied formula is correct or not. The mentor may decide to intervene in order to encourage the student. For example, if the student has written an incorrect formula, and later he rewrites the formula correctly, the mentor would encourage him. The mentor would also intervene to eliminate cycling. That is, if the student takes some previous move for the second time, the mentor would mention it to him.

The problems in the second round

The system in the second round was able to handle two tutoring styles. However, when a student started to interact with the tutor he was forced to continue the interaction using this tutoring style throughout the entire problem solving process of at least one probability problem. The same is true for a student who started to interact with the mentor. The assumption that students can be classified into two classes, those who know the basics and need practicing in the application of strategic knowledge, and those that need tutoring in the basics, was over simplified. In the experiments with the second round it became apparent that students can not be classified into two distinct categories in a satisfactory way. Rather, the knowledge and skills that students have and acquire by using the system, represent a continuity from students who are very knowledgeable and skillful to those who are slow and need much more guidance. Moreover, it seemed that students needed the "tutor" style of tutoring in some parts of

the session but preferred the "mentor" tutoring style in other parts. In the experiments, some students were confident in the symbolic stage and therefore proceeded in the mentor. However, in the solve stage they encountered difficulties. Although the mentor often transferred control to the tutor for handling the many mistakes that these students made, control was kept most of the time by the mentor. Each time the tutor returned control to the mentor, students received the mentor's message "Please go on", which did not tell what to do next, and that discouraged the students who did not know how to proceed. For those students it was quite obvious that the preliminary decision for using the mentor, while being the right one for the symbolic stage, was wrong for the solve stage. Other observations showed the other aspect of the same problem. Some students started the dialogue with shallow knowledge. The tutor tutoring style seemed to be appropriate in the beginning of the dialogue. However, after some time, the students understood the system's approach of backward reasoning, and then wanted to skip some of the simple intermediate moves. This of course, was not allowed by the tutor, and therefore the students became frustrated.

A second major problem with the second round system was that some students felt that they did not learn enough through the problem solving exercise. The system did not provide any in-depth discussion of probability concepts. When the research with SPIRIT began, the system was defined as a tutoring system whose purpose was to do coaching rather than teaching. That is, an assumption was made that a student interacting with the system knows the basic concepts and what he needs is practice. Despite the fact that teaching was not SPIRIT's intended central task, it seemed from the experiments that some teaching capabilities are needed to be incorporated in the system. This requirement is not a simple one because, instructional goals are sometimes conflicting. In particular, the goal of coaching may contradict the goal of teaching basic concepts because elaborate discussions of concepts in the midst of the problem solving process may severely disturb the flow of reasoning and inhibit automatism [15].

The two major problems mentioned above demonstrated the lack of extensive student modeling. The knowledge about the student in the second round was represented by the list of working memory elements that describe the student's previous moves. This knowledge does not provide a specific estimate of the student's aptitude. Such an estimate is important in order to decide which tutoring style to use, and on what level to provide discussions of probability concepts. Another kind of student modeling is required to specify the student's strengths and weaknesses. This information is important in order for the system to make plausible decisions about the trade-off between coaching and teaching. That is, the decisions of when and what to discuss in-depth.

Some problems were observed in the mentor and in the tutor working as separate systems. Despite the commands *data* and *formula*, students interacting with the mentor sometimes lost context. An assumption was made in the mentor that backward reasoning is simple straight forward reasoning. For example, suppose a student has written the formula $p(B / A) = p(A \cap B) : p(A)$, and suppose also that he has found $p(A \cap B)$. At this point a help request is understood as saying "What should I do next?". The mentor would respond to the help request by saying: "Think about finding $p(A)$". In the experiments, when some students encountered this scenario they were in doubt about why the mentor was telling them this. That is, they did not understand the backward reasoning underlying the mentor's recommendation. These results implied that in the mentor the system's approach should be more explicitly articulated.

Some students interacting with the tutor, although feeling comfortable with the tutoring style, were sometimes inhibited from skipping simple intermediate moves. For these students a switch from the tutor to the mentor would not be appropriate because they generally preferred the tutor to the mentor. Thus a mechanism that would allow jumping ahead in the tutor also seems desirable.

5.3. SPIRIT: Reaching the current state

The problems discussed above motivated the incorporation of extensive student modeling, specialists that discuss in detail important probability concepts, and the integration of the tutor and the mentor yielding the current state of SPIRIT.

Student modeling

The purpose of student modeling is twofold: first to construct and maintain a belief about the student's aptitude, and second to represent knowledge about the student's weak and strong points. The first purpose is accomplished by general modeling capability and the second by sub-skills modeling.

The Student Model is implemented using an object oriented programing language called HOUSE that was developed in the Decision Sciences Laboratory of the University Of Pittsburgh by Casey Quayle. This language uses some of the ideas of SMALLTALK [16] and of FLAVORS [17].

The object oriented paradigm seems appropriate for the implementation of the Student Model because:

- The Student Model has a dual task. It represents data about the student, and it performs analysis on this data making assumptions about the student's ability. Thus, it seems appropriate to combine these two tasks in objects.

- The Student Model includes some entities corresponding to sub-skills the student is supposed to acquire, and that use uniform protocols. These entities are represented in objects, which respond to a uniform set of messages, thereby facilitating the implementation. This is, all the sub-skill objects respond to the same message format and perform similar processing.

- The use of objects suits the evolving characteristic of the system. For example, new sub-skills can easily be added, and a change in the representation of the data can be done only once, and not as many times as the number of sub-skills.

- The object oriented paradigm facilitated the implementation of the modular

architecture in SPIRIT. The Student Model is interfaced to the other system's components by messages. Thus, the other components do not have to assume any particular data structures in the Student Model, and any change in the data structures does not necessarily impose changes in the other components.

General Modeling

Each move that the student takes reveals some knowledge about the student's aptitude. At any point in the dialogue, the system's belief about the student's aptitude is based on the cumulative evidence derived from the moves that the student has taken so far. An additional move taken by the student changes this information and therefore can change the belief that was held before. Thus, system's belief must be revised after each move taken by the student. There are two measures associated with each possible move. One is a measure of difficulty and the second is a measure of correctness. The first measure specifies the degree of difficulty in making the move correctly. The second specifies the degree to which the student is close to the correct move. A move whose two measures are high supports the belief that the student's aptitude is high. The system's belief is constructed based on these two measures of all the moves taken so far. The system's belief about the student's aptitude is used for several purposes: it is used for selecting the proper tutoring style; the specialists use it for tailoring the level of discussions to the student's ability; and it is used in order to provide the student with an evaluation at the end of the tutoring session.

Sub-skills modeling

The system needs to know how the student has done with respect to each sub-skill involved in problem solving. The sub-skills modeling provides a mechanism by which the system can easily examine the student's previous moves with respect to each sub-skill. The sub-skills modeling component does not make decisions. It merely represents information in a way that facilitates the analysis done by the specialists. For a detailed description of the sub-skills modeling process see [18].

Specialists

The purpose of the specialists is to discuss in detail basic probability concepts. A discussion evoked by a specialist includes two parts, an abstract discussion of the concept and an analysis of the student's previous moves pertinent to the current discussion. The specialists serve the purpose of teaching the basics, which sometimes contradicts, as I mentioned earlier, the instructional purpose of pursuing automatism. Therefore, a specialist is called only when a student reveals that he lacks the basic knowledge in the corresponding concept. The system reaches this conclusion when several hints and explanations in various levels have failed in helping the student. In this case, a specialist is called and assumes control.

A specialist can be called more than once, but it does not present the abstract discussion a second time, it only reminds the student about the previous discussion, and analyzes his previous moves. The abstract discussion is tailored to the student's aptitude. A student who has high aptitude receives a more challenging presentation of the concept that often includes abstract questions. The discussion may also include topics which are not presented to the student who has low aptitude. For example, the marginal probability specialist tells the high aptitude student that the two arguments of the marginal formula are mutually.exclusive and guides him by asking him first to write the formula in set notation. The student who has low aptitude receives a more straightforward explanation aimed directly at the final formula. In the second part of the discussion, the specialist examines the student's model to see if he has taken other moves that are related to the current issue under discussion.

Tutor-mentor transitions

The framework of two separate systems was not satisfactory in the second round. The approach adopted in the final system was that the system should be able to change its tutoring style as appropriate throughout the dialogue. At certain points in the dialogue either the system or the student may wish to change the current tutoring style. The student does not know the internal system structure and he is not aware of the two distinct tutoring styles and of the frequent transitions between them. However, the student may initiate a transition from tutor to mentor by telling the system: "I do not want to answer your question so leave me alone and let me solve the problem my way.". The student can say this by responding to a specific question asked by the system by typing the command "alone". Although the student does not know about the concepts of "tutor" and "mentor" he would never type the command "alone"

while interacting with the mentor because only the tutor asks specific questions. Thus, the student sees one system that changes its tutoring style from time to time, taking into account his preferences.

The system initiates transitions when it seems appropriate based on the student's aptitude and his recent moves. In the tutor, there are three points, called transition points, where the system considers whether or not to switch to the mentor. The first point is in the beginning of the dialogue, the second is when the student completes the symbolic stage, and the third point is after the final solution is found and before the next problem is presented. At these points, the system initiates a transition from the tutor to the mentor if the student's aptitude is sufficiently high. Much more flexibility in intitiating transitions is available in the mentor, which transfers control to the tutor whenever it seems appropriate. The tutor handles the mistake that caused the transition and then returns control back to the mentor. Thus, most of the transitions from the mentor are temporary. The mentor is also capable of transferring control to the tutor for handling the rest of the dialogue (or until the tutor would initiate its own transition) at certain points when the student's aptitude is judged relatively low.

Alleviating some of the other problems

In the second round the mentor improperly assumed that backward reasoning is a straightforward approach and that the student can easily use it. The mentor in the final system makes backward reasoning explicit rather than implicit.

Another problem in the second round was that the tutor did not allow skipping of moves. In the final system the tutor allows this by enabling the specific help request, which was used previously only in the mentor. When a student wishes to skip some moves, he types the probability expression that he wants to explore followed by a "?". The tutor then makes this expression the current focus of the discussion, and proceeds as though it had reached this expression through the usual backward chaining process.

6. Discussion

SPIRIT has been used by undergraduate students taking an introductory course in probability theory. Their reaction to the system is very encouraging, and we intend to report results of experiments in the near future. However, the system has some

drawbacks and another round might improve the system even more.

SPIRIT's shortcomings seem to be rooted in the inadequacy of its tutoring model. One problem is the feeling expressed by both subjects and observers of the experiments that the system does not focus on meaningful learning [19]. Meaningful learning occurs when the material that is learned is related to some general structure or principle. Often, after meaningful learning, individuals are better able to transfer their knowledge to new kinds of problems. In contrast, subjects who interacted with SPIRIT, did not show the desired ability to transfer knowledge they acquire while solving one kind of problem to solving another kind of problem. In the experiments, students solved two typical Bayesian problems followed by a problem that required the application of the addition rule. They were able to transfer knowledge they had acquired in solving the first problem to the second one because both problems have a similar structure. However, they failed in the third problem, being unable to apply effectively the strategy of backward reasoning they used before.

By comparing protocols generated by SPIRIT to the human tutor's protocols, it appears that the system does not emphasize enough the general principles and concepts of probability theory, but rather focuses on the task of getting the final answer to the current probability problem. The human tutor pursued two goals which sometimes conflict with each other. The first goal is making the student understand the concepts and the foundations of probability theory. That is, the tutor would like the student to be able to derive and to build an abstract representation of the real world in terms of probability concepts as is represented by the English sentences in the probability problem. The second goal is to teach the student how to solve similar probability problems. While the first goal is expressed by the cognitive outcome that we want to achieve, the second is expressed in terms of acquiring an applicable skill. It seems that it is possible to achieve one goal without achieving the other. For example, the student may be able to solve probability problems by playing in a systematic way with the formulae, but without having the representation of the world in terms of probability concepts. SPIRIT focuses on the second goal and does not suffuciently pursue the first one.

A second major problem with SPIRIT, which is related to what we have discussed above, is the problem of losing context, and losing or forgetting pieces of knowledge provided by the system. That is, subjects often forget what the final question is, and

what the intermediate problem is, which is the focus of their attention. In an earlier stage of the system, one of the reasons for that problem was hypothesized to be the fact that SPIRIT had (incorrectly) assumed that the student knew the domain independent problem solving method of backward chaining. This assumption was embedded in several aspects of the system's behavior. Later, this assumption was refuted and the system, currently, explicitly teaches backward reasoning. Despite this, although experiments conducted after SPIRIT's behavior has been changed showed some improvement in keeping context, losing context still remains a major complaint. This unexpected result implies that something more basic is wrong in SPIRIT. In Barzilay [18] it was hypothesized that one of the reasons for SPIRIT's shortcomings is its inability to tie various issues together as the human expert tutor does. Based on this hypothesis, Barzilay proposed a framework for enhancing the tutoring model in SPIRIT. This framework uses a lattice data structure of probability concepts to be used by the tutoring exert in making decisions about which concepts to discuss at what points in the dialogue, and to what other concepts they should be tied.

SPIRIT's success is partly due to OPS5 that facilitated the evolution of the system. However, despite the advantages of OPS5 discussed earlier, OPS5 has some disadvantages that were overcome by integrating OPS5 with other AI methods. The important disadvantages were:

- OPS5 is weak in list processing.

- No processing is allowed in a production's LHS, so one logic rule often needs to be implemented in several interrelated productions making the explicit knowledge in the logic rule segmented and less explicit. Also, debugging becomes tricky because of the interrelated productions.

- Productions can not be organized hierarchically. The "context" mechanism suits a linear organization (e.g. as in R1, [20]), but does not suit the hierarchical structure in SPIRIT very well. A tangled hierarchy is even more difficult to implement using this mechanism.

- Complex data structures are not supported. The only knowledge structure which can be directly accessed by OPS5 is the list of working memory elements.

7. Conclusions

We presented a brief description of SPIRIT by taking three snapshots of the system's behavior along the evolution process. Our philosophy is that an effective system evolves through a process of experimenting and tuning. A system needs a long tuning process in order to intelligently make tutoring decisions, such as deciding when things are really going wrong and that an intervention is required, or when it is better to let the student struggle by himself. Although, presently, the decisions that SPIRIT makes are not always the ones that the well experienced tutor would, we believe we have made quite a lot of progress since we started to experiment with SPIRIT. For example, the idea of achieving a flexible dialogue style by employing the *tutor* and the *mentor* came to our mind after experiments with SPIRIT in its early stages. Despite the fact that this idea meant a significant change in SPIRIT's behavior, the implementation was completed in a very short time.

The methods of production system and object oriented programming proved to be very helpful in the evolutionary design of SPIRIT. All together SPIRIT is implemented by three programming paradigms: procedure oriented (LISP), rule oriented (OPS5) and object oriented (HOUSE). Thus, the research in SPIRIT demonstrates the need for AI programming environments that support several paradigms. We hope that our work will help AI designers in the difficult task of choosing the right tools for the right tasks.

SPIRIT is one of the very few ITS that actually do a satisfactory tutoring job. Students in the University of Pittsburgh used the system as an assistant in the introductory probability course. Most of them expressed real enthusiasm and after passing the course attributed some of their success to the system.

SPIRIT covers only a small subset of probability theory. However, in terms of complexity and size, the system is quite large. It employs about 120 OPS5 productions and some 100 LISP subroutines. It has been developed over a period of 18 months and it runs (with a reasonable response time) on a VAX 11/780.

References

1. Sleeman, D. and Brown, J. S., *Intelligent Tutoring Systems*, Academic Press, London, (1982).

2. Clancey, W. J., *Transfer of rule-based expertize through a tuorial dialogue.* Rep. no. STAN-CS-79-769. Dep. of Computer Science Stanford University CA., (1979).
3. Brown, J. S. and Burton, R. R. Diagnostic Models For Procedural Bugs In Basic Mathematical Skills. *Cognitive Science* 2 (1978) 155-192.
4. Sleeman, D. H. and Smith, M. J., Modeling Student's Problem Solving. *Artificial Intelligence.* 16 (1981) 171-188.
5. Shortliffe, E. H. *MYCIN, A Rule-Based Computer Program For Advising Physicians Regarding Antimicrobial Therapy Selection.* Ph.D. dissertation in Medical Information Sciences. Stanford University, California (1974).
6. Nilsson, N. J. *Principles of Artificial Intelligence* Tioga Puublishing Company, Palo Alto CA (1980)
7. O'Shera, T. A Self-Improving Quadratic Tutor. In: Sleeman, D. and Brown, J. S. (Ed), *Intelligent Tutoring Systems.* Academic Press, London, (1982).
8. Collins, A. Processes in acquiring knowledgd. In Anderson, R. C. and Spiro, R. J. (Ed) *Schooling and the acquisition of knowledge* Hillsdale, N.J. Lawrence Erlbaum, (1976) 339-363
9. Lantz, B. S., Bregar, W. S. and Fareley, A. M. An Intelligent CAI System For Tutoring Equation Solving. *Journal Of Computer Based Education.* 10 , 1-2 (1983) 35-42.
10. Van Melle, W., *A Domain-independent production rule system for consultation programs.* Ph.D. dissertation, Stanford University, Computer Science Department, STAN-CS-80-820, (1980).
11. Forgy, C. L. *OPS5 Manual.* CS78-3612, Dep. Of Computer Science Carnegie Mellon University, Pittsburgh (1981).
12. Burton, R. R. and Brown, J. S. An Investigation Of Computer Coaching For Informal Learning Activities. In: Sleeman, D. and Brown, J. S. (Ed), *Intelligent Tutoring Systems.* Academic Press, London, (1982) chap. 4.
13. Greeno, J. G., A Study of problem solving. In Glaser, R. (Ed.) *Advances In Instructional Psychology* 1 Hillsdale , N.J. Erlbaum. (1978).
14. Riley, M. S., Greeno, J. G. and Heller, J. I. Development of Children's Problem-Solving Ability in Arithmetic. In Ginsburg, H. P. (Ed.) *The development of mathematical thinking.* New York, Academic Press, (1983).
15. Fitts, P. M. Factors In Complex Skill Trainning. In Glaser, R. (Ed) *Trainig research and education* New York. John Wiley and Sons, (1962).
16. Goldberg A., Robson, D. and Ingalls, D. *Smalltalk-80: the language and its implementation.* Reading MA, Addison-Wesley, (1982).

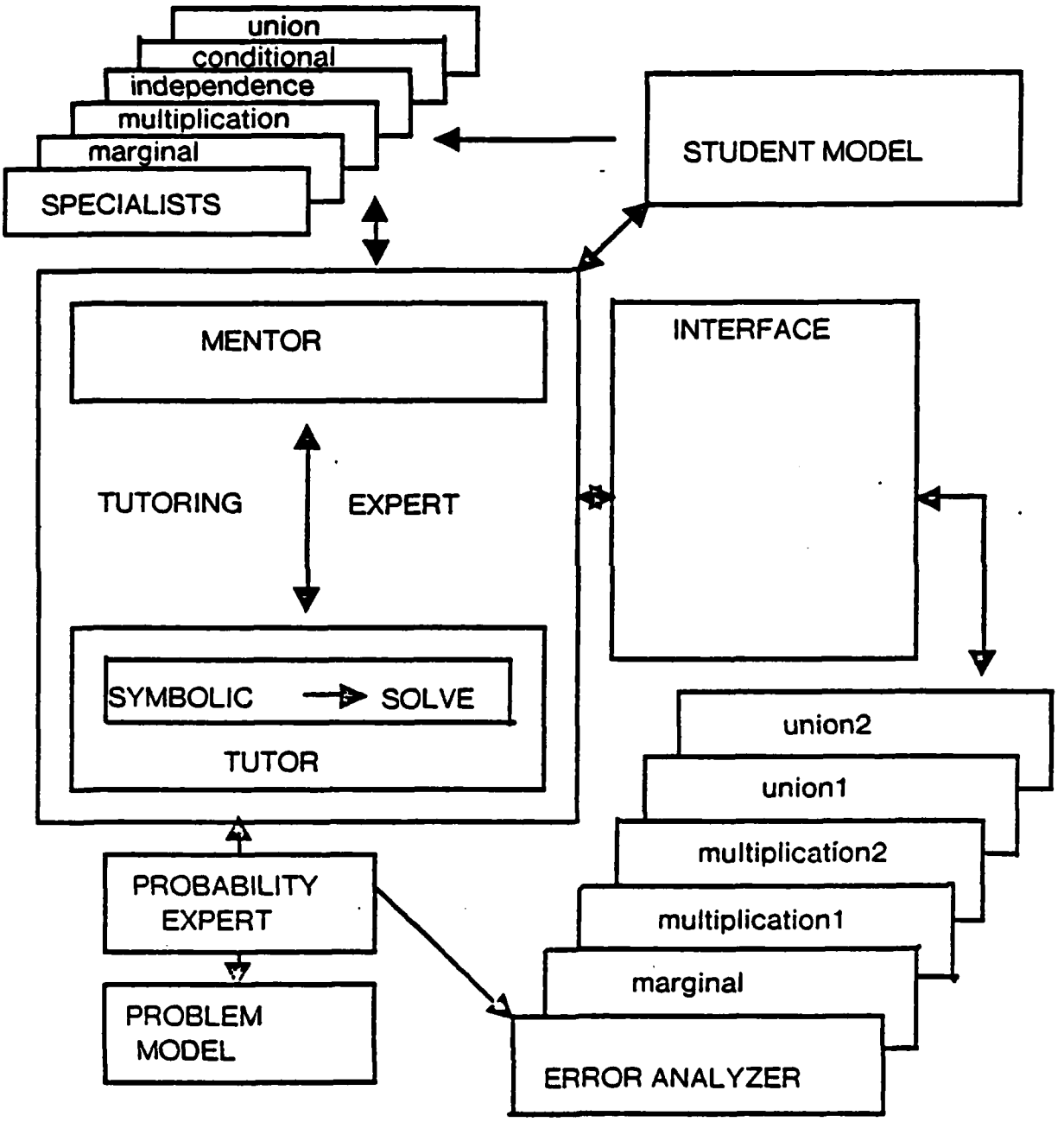
17. Cannon, H. I. *Flavors: a non-hierarchical approach to object-oriented programming* Personal Communication, (1982).

18. Barzilay, A. *An Expert System For Tutoring Probability Theory.* A Ph.D. dissertation, Graduate School of Business, University of Pittsburgh, (1984).

19. Greeno, J. G. Conceptual Entities. In Gentner, D. and Stevens A. L. *Mental Models*, LEA, Hillsdale, NJ. (1983), 227-251.

20. McDermott, J. *R1: A rule based configurer of computer sysytems.* Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA. (1980).

Figure 1: SPIRIT's architecture



1 Robert Ahlers
Code N711
Human Factors Laboratory
NAVTRAEQUIPCEN
Orlando, FL 32813

1 Dr. Ed Aiken
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Meryl S. Baker
Navy Personnel R&D Center
San Diego, CA 92152

1 Code N711
Attn: Arthur S. Blaiwes
Naval Training Equipment Center
Orlando, FL 32813

1 Dr. Nick Bond
Office of Naval Research
Liaison Office, Far East
APO San Francisco, CA 96503

1 Dr. Richard Cantone
Navy Research Laboratory
Code 7510
Washington, DC 20375

1 Dr. Fred Chang
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Susan Chipman
Code 442PT
Office of Naval Research
800 N. Quincy St.
Arlington, VA 22217

1 Chief of Naval Education and Training
Liason Office
Air Force Human Resource Laboratory
Operations Training Division
WILLIAMS AFB, AZ 85224

1 Dr. Stanley Collyer
Office of Naval Technology
800 N. Quincy Street
Arlington, VA 22217

1 CDR Mike Curran
Office of Naval Research
800 N. Quincy St.
Code 270
Arlington, VA 22217

1 DR. PAT FEDERICO
Code P13
NPRDC
San Diego, CA 92152

1 Dr. Jude Franklin
Code 7510
Navy Research Laboratory
Washington, DC 20375

1 LT Steven D. Harris, MSC, USN
RFD 1, Box 243
Riner, VA 24149

1 Dr. Jim Hollan
Code 14
Navy Personnel R & D Center
San Diego, CA 92152

1 Dr. Ed Hutchins
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Norman J. Kerr
Chief of Naval Technical Training
Naval Air Station Memphis (75)
Millington, TN 38054

1 Dr. Peter Kincaid
Training Analysis & Evaluation Group
Dept. of the Navy
Orlando, FL 32813

1 Dr. William L. Maloy (02)
Chief of Naval Education and Training
Naval Air Station
Pensacola, FL 32508

1 Dr. Joe McLachlan
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. James McMichael
Navy PersonnelR&D Center
San Diego, CA 92152

1 Dr William Montague
NPRDC Code 13
San Diego, CA 92152

1 Library, Code P201L
Navy Personnel R&D Center
San Diego, CA 92152

1 Technical Director
Navy Personnel R&D Center
San Diego, CA 92152

6 Commanding Officer
Naval Research Laboratory
Code 2627
Washington, DC 20390

1 Office of Naval Research
Code 433
800 N. Quincy SStreet
Arlington, VA 22217

6 Personnel & Training Research Group
Code 442PT
Office of Naval Research
Arlington, VA 22217

1 Psychologist
ONR Branch Office
1030 East Green Street
Pasadena, CA 91101

1 Office of the Chief of Naval Operations
Research Development & Studies Branch
OP 115
Washington, DC 20350

1 LT Frank C. Petho, MSC, USN (Ph.D)
CNET (N-432)
NAS
Pensacola, FL 32508

1 Dr. Bary Poock
Operations Research Department
Code 55PK
Naval Postgraduate School
Monterey, CA 93940

1 Dr. Gil Ricard
Code N711
NTEC
Orlando, FL 32813

1 Dr. Robert G. Smith
Office of Chief of Naval Operations
OP-987H
Washington, DC 20350

1 Dr. Alfred F. Smode, Director
Department N-7
Naval Training Equipment Center
Orlando, FL 32813

1 Dr. Richard Snow
Liaison Scientist
Office of Naval Research
Branch Office, London
Box 39
FPO New York, NY 09510

1 Dr. Richard Sorensen
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Frederick Steinheiser
CNO - OP115
Navy Annex
Arlington, VA 20370

1 Dr. Thomas Sticht
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. James Tweeddale
Technical Director
Navy Personnel R&D Center
San Diego, CA 92152

1 Roger Weissinger-Baylon
Department of Administrative Sciences
Naval Postgraduate School
Monterey, CA 93940

1 Dr. Douglas Wetzel
Code 12
Navy Personnel R&D Center
San Diego, CA 92152

1 Mr John H. Wolfe
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Wallace Wulfeck, III
Navy Personnel R&D Center
San Diego, CA 92152

1 H. William Greenup
Education Advisor (E031)
Education Center, MCDEC
Quantico, VA 22134

1 Special Assistant for Marine
Corps Matters
Code 100M
Office of Naval Research
800 N. Quincy St.
Arlington, VA 22217

1 DR. A.L. SLAFKOSKY
SCIENTIFIC ADVISOR (CODE RD-1)
HQ, U.S. MARINE CORPS
WASHINGTON, DC 20380

1 Technical Director
U. S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Mr. James Baker
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Beatrice J. Farr
U. S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Milton S. Katz
Training Technical Area
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Marshall Narva
US Army Research Institute for the
Behavioral & Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Harold F. O'Neil, Jr.
Director, Training Research Lab
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Commander, U.S. Army Research Institute
for the Behavioral & Social Sciences
ATTN: PERI-BR (Dr. Judith Orasanu)
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Joseph Psotka, Ph.D.
ATTN: PERI-IC
Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 Dr. Robert Sasmor
U. S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

1 U.S. Air Force Office of Scientific Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, DC 20332

1 Dr. Earl A. Alluisi
HQ, AFHRL (AFSC)
Brooks AFB, TX 78235

1 Mr. Raymond E. Christal
AFHRL/MOE
Brooks AFB, TX 78235

1 Bryan Dallman
AFHRL/LRT
Lowry AFB, CO 80230

1 Dr. Genevieve Haddad
Program Manager
Life Sciences Directorate
AFOSR
Bolling AFB, DC 20332

1 Dr. T. M. Longridge
AFHRL/OTE
Williams AFB, AZ 85224

1 Dr. John Tangney
AFOSR/NL
Bolling AFB, DC 20332

1 Dr. Joseph Yasatuke
AFHRL/LRT
Lowry AFB, CO 80230

12 Defense Technical Information Center
Cameron Station, Bldg 5
Alexandria, VA 22314
Attn: TC

1 Military Assistant for Training and Personnel Technology
Office of the Under Secretary of Defens for Research & Engineering
Room 3D129, The Pentagon
Washington, DC 20301

1 Major Jack Thorpe
DARPA
1400 Wilson Blvd.
Arlington, VA 22209

1 Dr. Robert A. Wisher
OUSDRE (ELS)
The Pentagon, Room 3D129
Washington, DC 20301

1 Dr. Patricia A. Butler
NIE-BRN Bldg, Stop # 7
1200 19th St., NW
Washington, DC 20208

1 Dr. Paul G. Chapin
Linguistics Program
National Science Foundation
Washington, DC 20550

1 Edward Esty
Department of Education, OERI
MS 40
1200 19th St., NW
Washington, DC 20208

1 Dr. Arthur Melmed
724 Brown
U. S. Dept. of Education
Washington, DC 20208

1 Dr. Andrew R. Molnar
Office of Scientific and Engineering
Personnel and Education
National Science Foundation
Washington, DC 20550

1 Dr. Everett Palmer
Mail Stop 239-3
NASA-Aaes Research Canter
Moffett Field, CA 94035

1 Dr. Ramsay W. Selden
National Institute of Education
1200 19th St., NW
Washington, DC 20208

1 Dr. Mary Stoddard
C 10, Mail Stop B296
Los Alamos National Laboratories
Los Alamos, NM 87545

1 Dr. Edward C. Weiss
National Science Foundation
1800 G Street, NW
Washington, DC 20550

1 Dr. Frank Withrow
U. S. Office of Education
400 Maryland Ave. SW
Washington, DC 20202

1 Dr. Joseph L. Young, Director
Memory & Cognitive Processes
National Science Foundation
Washington, DC 20550

1 Dr. Erling B. Andersen
Department of Statistics
Studiestraede 6
1455 Copenhagen
DENMARK

1 Dr. John R. Anderson
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

1 Dr. John Annett
Department of Psychology
University of Warwick
Coventry CV4 7AJ
ENGLAND

1 Dr. Michael Atwood
ITT - Programming
1000 Oronoque Lane
Stratford, CT 06497

1 Dr. Alan Baddeley
Medical Research Council
Applied Psychology Unit
15 Chaucer Road
Cambridge CB2 2EF
ENGLAND

1 Eva L. Baker
Director
UCLA Center for the Study of Evaluation
145 Moore Hall
University of California, Los Angeles
Los Angeles, CA 90024

1 Dr. Jonathan Baron
80 Glenn Avenue
Berwyn, PA 19312

1 Mr. Avron Barr
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Menucha Birenbaum
School of Education
Tel Aviv University
Tel Aviv, Ramat Aviv 69978
Israel

1 Dr. John Black
Yale University
Box 11A, Yale Station
New Haven, CT 06520

1 Dr. John S. Brown
XEROX Palo Alto Research Center
3333 Coyote Road
Palo Alto, CA 94304

1 Dr. Glenn Bryan
6208 Poe Road
Bethesda, MD 20817

1 Dr. Bruce Buchanan
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Jaime Carbonell
Carnegie-Mellon University
Department of Psychology
Pittsburgh, PA 15213

1 Dr. Pat Carpenter
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

1 Dr. Micheline Chi
Learning R & D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

1 Dr. William Clancey
Department of Computer Science
Stanford University
Stanford, CA 94306

1 Dr. Michael Cole
University of California
at San Diego
Laboratory of Comparative
Human Cognition - D003A
La Jolla, CA 92093

1 Dr. Allan M. Collins
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

1 Dr. Lynn A. Cooper
LRDC
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

1 Dr. Emanuel Donchin
Department of Psychology
University of Illinois
Champaign, IL 61820

1 Dr. Thomas M. Duffy
Department of English
Carnegie-Mellon University
Schenley Park
Pittsburgh, CA 15213

1 ERIC Facility-Acquisitions
4833 Rugby Avenue
Bethesda, MD 20014

1 Dr. Anders Ericsson
Department of Psychology
University of Colorado
Boulder, CO 80309

1 Dr. Paul Feltovich
Department of Medical Education
Southern Illinois University
School of Medicine
P.O. Box 3926
Springfield, IL 62708

1 Mr. Wallace Feurzeig
Department of Educational Technology
Bolt Beranek & Newman
10 Moulton St.
Cambridge, MA 02238

1 Univ. Prof. Dr. Gerhard Fischer
Liebiggasse 5/3
A 1010 Vienna
AUSTRIA

1 Professor Donald Fitzgerald
University of New England
Armidale, New South Wales 2351
AUSTRALIA

1 Dr. Dexter Fletcher
University of Oregon
Department of Computer Science
Eugene, OR 97403

1 Dr. John R. Frederiksen
Bolt Beranek & Newman
50 Moulton Street
Cambridge, MA 02138

1 Dr. Michael Genesereth
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Dedre Gentner
Bolt Beranek & Newman
10 Moulton St.
Cambridge, MA 02138

1 Dr. Don Gentner
Center for Human Information Processing
University of California, San Diego
La Jolla, CA 92093

1 Dr. Robert Glaser
Learning Research & Development Center
University of Pittsburgh
3939 O'Hara Street
PITTSBURGH, PA 15260

1 Dr. Marvin D. Glock
217 Stone Hall
Cornell University
Ithaca, NY 14853

1 Dr. Joseph Goguen
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

1 Dr. Daniel Gopher
Faculty of Industrial Engineering
& Management
TECHNION
Haifa 32000
ISRAEL

1 Dr. Bert Green
Johns Hopkins University
Department of Psychology
Charles & 34th Street
Baltimore, MD 21218

1 DR. JAMES G. GREENO
LRDC
UNIVERSITY OF PITTSBURGH
3939 O'HARA STREET
PITTSBURGH, PA 15213

1 Dr. Barbara Hayes-Roth
Department of Computer Science
Stanford University
Stanford, CA 95305

1 Dr. Frederick Hayes-Roth
Teknowledge
525 University Ave.
Palo Alto, CA 94301

1 Dr. Joan I. Heller
Graduate Group in Science and
Mathematics Education
c/o School of Education
University of California
Berkeley, CA 94720

1 Dr. James R. Hoffman
Department of Psychology
University of Delaware
Newark, DE 19711

1 Melissa Holland
American Institutes for Research
1055 Thomas Jefferson St., N.W.
Washington, DC 20007

1 Glenda Greenwald, Ed.
Human Intelligence Newsletter
P. O. Box 1163
Birmingham, MI 48012

1 Dr. Earl Hunt
Dept. of Psychology
University of Washington
Seattle, WA 98105

1 Robin Jeffries
Computer Research Center
Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304

1 Dr. Marcel Just
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

1 Dr. Steven M. Keele
Dept. of Psychology
University of Oregon
Eugene, OR 97403

1 Dr. Scott Kelso
Haskins Laboratories, Inc
270 Crown Street
New Haven, CT 06510

1 Dr. David Kieras
Department of Psychology
University of Arizona
Tuscon, AZ 85721

1 Dr. Walter Kintsch
Department of Psychology
University of Colorado
Boulder, CO 80302

1 Dr. Stephen Kosslyn
1236 William James Hall
33 Kirkland St.
Cambridge, MA 02138

1 Dr. Pat Langley
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, PA 15213

1 Dr. Jill Larkin
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

1 Dr. Alan Lesgold
Learning R&D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15260

1 Dr. Jim Levin
University of California
at San Diego
Laboratory for Comparative
Human Cognition - D003A
La Jolla, CA 92093

1 Dr. Michael Levine
Department of Educational Psychology
210 Education Bldg.
University of Illinois
Champaign, IL 61801

1 Dr. Marcia C. Linn
Lawrence Hall of Science
University of California
Berkeley, CA 94720

1 Dr. Don Lyon
P. O. Box 44
Higley , AZ 85236

1 Dr. Jay McClelland
Department of Psychology
MIT
Cambridge, MA 02139

1 Dr. James R. Miller
Computer*Thought Corporation
1721 West Plano Highway
Plano, TX 75075

1 Dr. Mark Miller
Computer*Thought Corporation
1721 West Plano Parkway
Plano, TX 75075

1 Dr. Tom Moran
Xerox PARC
3333 Coyote Hill Road
Palo Alto, CA 94304

1 Dr. Allen Munro
Behavioral Technology Laboratories
1845 Elena Ave., Fourth Floor
Redondo Beach, CA 90277

1 Dr. Donald A Norman
Cognitive Science, C-015
Univ. of California, San Diego
La Jolla, CA 92093

1 Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, VA 22311

1 Prof. Seymour Papert
20C-109
Massachusetts Institute of Technology
Cambridge, MA 02139

1 Dr. James W. Pellegrino
University of California,
Santa Barbara
Dept. of Psychology
Santa Barabara , CA 93106

1 Dr. Nancy Pennington
University of Chicago
Graduate School of Business
1101 E. 58th St.
Chicago, IL 60637

1 Dr. Richard A. Pollak
Director, Special Projects
MECC
2354 Hidden Valley Lane
Stillwater, MN 55082

1 DR. PETER POLSON
DEPT. OF PSYCHOLOGY
UNIVERSITY OF COLORADO
BOULDER, CO 80309

1 Dr. Steven E. Poltrock
Bell Laboratories 2D-444
600 Mountain Ave.
Murray Hill, NJ 07974

1 Dr. Mike Posner
Department of Psychology
University of Oregon
Eugene, OR 97403

1 Dr. Lynne Reder
Department of Psychology
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

1 Dr. Fred Reif
Physics Department
University of California
Berkeley, CA 94720

1 Dr. Lauren Resnick
LRDC
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 1521

1 Dr. Jeff Richardson
Denver Research Institute
University of Denver
Denver, CO 80208

1 Mary S. Riley
Program in Cognitive Science
Center for Human Information Processing
University of California, San Diego
La Jolla, CA 92093

1 Dr. Andrew M. Rose
American Institutes for Research
1055 Thomas Jefferson St. NW
Washington, DC 20007

1 Dr. Ernst Z. Rothkopf
Bell Laboratories
Murray Hill, NJ 07974

1 Dr. William B. Rouse
Georgia Institute of Technology
School of Industrial & Systems
Engineering
Atlanta, GA 30332

1 Dr. David Rumelhart
Center for Human Information Processing
Univ. of California, San Diego
La Jolla, CA 92093

1 Dr. Michael J. Saaet
Perceptronics, Inc
6271 Variel Avenue
Woodland Hills, CA 91364

1 Dr. Roger Schank
Yale University
Department of Computer Science
P.O. Box 2158
New Haven, CT 06520

1 Dr. Walter Schneider
Psychology Department
603 E. Daniel
Champaign, IL 61820

1 Dr. Alan Schoenfeld
Mathematics and Education
The University of Rochester
Rochester, NY 14627

1 Mr. Colin Sheppard
Applied Psychology Unit
Admiralty Marine Technology Est.
Teddington, Middlesex
United Kingdom

1 Dr. M. Wallace Sinaiko
Program Director
Manpower Research and Advisory Services
Smithsonian Institution
801 North Pitt Street
Alexandria, VA 22314

1 Gail K. Slemon
LOGICON, Inc.
4010 Sorrento Valley Blvd.
P.O. Box 85158
San Diego, CA 92138

1 Dr. Edward E. Smith
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

1 Dr. Elliott Soloway
Yale University
Department of Computer Science
P.O. Box 2158
New Haven, CT 06520

1 Dr. Kathryn T. Spoehr
Psychology Department
Brown University
Providence, RI 02912

1 Dr. Robert Sternberg
Dept. of Psychology
Yale University
Box 11A, Yale Station
New Haven, CT 06520

1 Dr. Albert Stevens
Bolt Beranek & Newman, Inc.
10 Moulton St.
Cambridge, MA 02238

1 DR. PATRICK SUPPES
INSTITUTE FOR MATHEMATICAL STUDIES IN
THE SOCIAL SCIENCES
STANFORD UNIVERSITY
STANFORD, CA 94305

1 Dr. Kikumi Tatsuoka
Computer Based Education Research Lab
252 Engineering Research Laboratory
Urbana, IL 61801

1 Dr. Maurice Tatsuoka
220 Education Bldg
1310 S. Sixth St.
Champaign, IL 61820

1 Dr. Perry W. Thorndyke
Perceptronics, Inc.
545 Middlefield Road, Suite 140
Menlo Park, CA 94025

1 Dr. Douglas Towne
Univ. of So. California
Behavioral Technology Labs
1845 S. Elena Ave.
Redondo Beach, CA 90277

1 Dr. Kurt Van Lehn
Xerox PARC
3333 Coyote Hill Road
Palo Alto, CA 94304

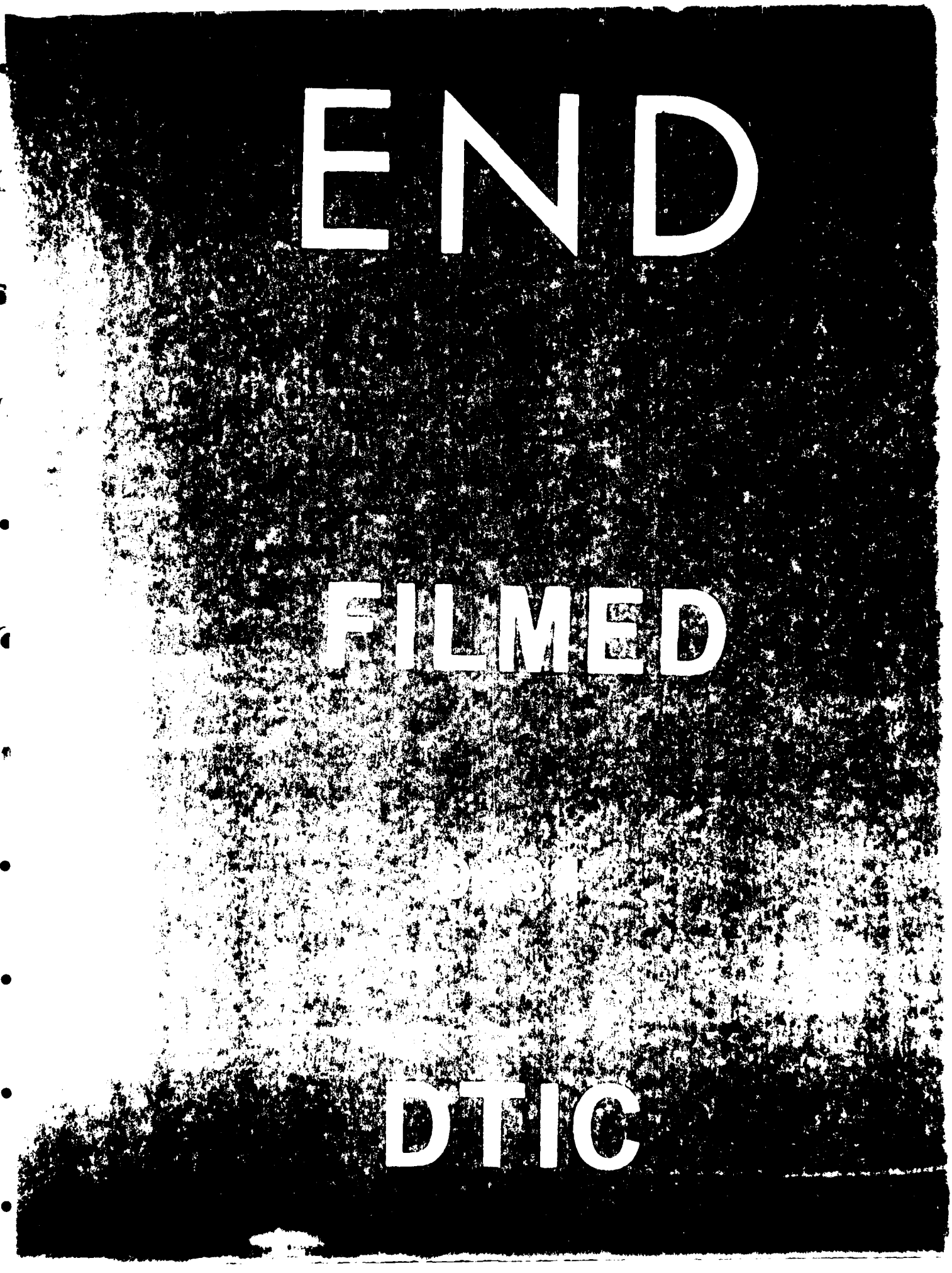
1 Dr. Keith T. Wescourt
Perceptronics, Inc.
545 Middlefield Road, Suite 140
Menlo Park, CA 94025

1 William B. Whitten
Bell Laboratories
29-610
Holadel, NJ 07733

1 Dr. Thomas Wickens
Department of Psychology
Franz Hall
University of California
405 Hilgarde Avenue
Los Angeles, CA 90024

1 Dr. Mike Williams
IntelliGenetics
124 University Avenue
Palo Alto, CA 94301

1 Dr. Joseph Wohl
Alphatech, Inc.
2 Burlington Executive Center
111 Middlesex Turnpike
Burlington, MA 01803


END
FILMED
DTIC