A MODEL OF TUTORING: FACILITATING KNOWLEDGE INTEGRATION USING MULTIPLE MODELS OF THE DOMAIN

BY

RAMZAN ALI KHUWAJA

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science
in the Graduate School of the
Illinois Institute of Technology

Approved______________________
Advisor

Chicago, Illinois
December, 1994

ACKNOWLEDGMENT

My path to, and through, the period of my life at Illinois Institute of Technology owes much to many and it is here that I want to acknowledge the influence and support of many different people.

I would like to thank my research advisor, Professor Martha Evens, for her intellectual support, constant encouragement, understanding and patience throughout the period of this research. I feel extremely fortunate to have her as my advisor. I would like to express my sincere appreciation to Dr. Allen Rovick and Dr. Joel Michael for their expert advice and extremely valuable discussions during this research.

I would also like to thank my thesis committee, Dr. C. Robert Carlson, Dr. James Kenevan, and Dr. Allen Wolach, for their attention and suggestions on my thesis.

I thank all my colleagues in the project group for their invaluable contributions, comments, and cooperation.

I am forever indebted to my family for their loving care and understanding. I am dedicating my thesis to my father, the late Madad Ali Khuwaja. He was a great father, a great teacher, and a great man. I would like to express my deepest affection and love to my fiancee, Jabeen, for her understanding and confidence in me.

I would also like to thank the Ministry of Education, Pakistan for their support during the course of this research.

This work was supported by the Cognitive Science Program, Office of Naval Research under Grant No. N00014-89-J-1952, Grant Authority Identification Number NR4422554, to Illinois Institute of Technology. The content does not reflect the position or policy of the government and no official endorsement should be inferred.

R.A.K.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF FIGURES

Figure Page

Figure Page

ABSTRACT

This thesis describes a model of tutoring. This model is intended for CIRCSIM-Tutor (v.3) - an Intelligent Tutoring System (ITS) - that teaches the functioning of the baroreceptor reflex to the first year medical students.

This model is based on the behavior of human tutors in the keyboard-to-keyboard sessions. The major theme of this model is that, in a problem-solving environment, it helps the student integrate his/her knowledge into a coherent qualitative causal model of the domain and solve problems in the domain. The key feature of this model is that it uses multiple qualitative models of the domain in the process of facilitating knowledge integration.

The development of this model of tutoring has been approached by using an ITS development framework that views the development of an ITS as a modeling activity. There are three major phases of this methodology. These are the conceptual phase, the system phase, and physical phase. At each phase a different model of an ITS results.

The conceptual model, resulting out of the conceptual phase, deals in this research only with the domain and the pedagogy aspects of tutoring. The domain knowledge here is consists of multiple qualitative models that are used to support decision making. This decision making process considers three major functions: what to teach, when to teach, and how to teach.

The system model, resulting out of the system phase, provides a generic framework to represent three different types of knowledge. These are the planning knowledge, the curriculum knowledge, and the domain knowledge. The system model can also be viewed as consisting of a set of tutoring spaces. Each space is responsible for performing one type of major decision of the tutor while interacting with the student. For CIRCSIM-Tutor (v.3) the following tutoring spaces are used: the major-objective space, the exercise space, the unit space, and the lesson space. The second model resulting out of the system phase is the architecture of the system. Here an object-oriented methodology is used to develop some of the major components of this architecture.

These architectural components are coded using the Common Lisp Object System on the Apple Macintosh.

CHAPTER I

INTRODUCTION

"Reform" - this word, nowadays, is commonly heard in all educational settings ranging from primary (Anderson, 1992) to professional education (Calderhead, 1988; Jonas, 1978; McGuire et al., 1983; Rothstein, 1987). This reform movement is due to the unsatisfactory state of our current educational system. According to Woolf (1986) "education is in trouble" (p. 1). Anderson (1992) noted that "The situation with respect to low educational achievement has been raised to the status of a national crisis in America" (p. 1). Woolf (1986), citing the literature, further states that

> ... an NSF study says, "Most Americans are moving towards virtual scientific and technological illiteracy" (National Science Foundation, 1983). Naisbitt says, "The generation graduating from high school today is the first generation in American history to graduate less skilled than its parents ..." (Naisbitt, 1984). R. Buckminister Fuller says, "Classrooms are desensitizing, stultifying and boring." (Fuller, 1962) ... Andrew Molnar from NSF says that only 75% of the teachers in America are qualified to teach the courses they are teaching (Molnar, 1986) ... America will be short one million teachers within four years ... People graduate without the basic skills necessary to function at the college level (pp. 1-2).

The problems are great and there is no single "magical" method available to improve the current state of education. But what we can do is to adopt the "divide and conquer" strategy and solve problems that are linked to this disastrous state of education, as far as possible.

1.1 <u>Two Problems</u>

According to Anderson (1984) "Knowledge is not a basket of facts" but rather "the essence of knowledge is structure" (p. 5). A study by Bloom (1984) shows that conventional teaching, which means a teacher presenting material in front of 20 - 200 people, provides one of the least effective methods for educational delivery. In this conventional form "too often school science and mathematics is studied as a disembodied set of facts and principles" (Wheatley, 1991, p. 13). Here "the learner is assumed to be, in the John Locke tradition, a *tabula rasa*" (Wheatley, 1991, p. 14). In such a situation, as Anderson (1984) noted, the majority of students "are unlikely to make the inferences required to *weave* the information ... into a coherent overall mental model" (p. 10).

Besides leaving student's knowledge in a *non-integrated (unstructured)* form, the second problem created by the current educational system is that it does not provide opportunities for students to develop higher mental processes such as the ability to solve problems (Bloom, 1984). Michael (1993) says that "American education has, in general, failed to prepare students ... to become problem solvers" (p. 37). According to Bloom (1984)

> ... Such higher mental processes ... enable the student to relate his or her learning to the many problems he or she encounters in day-to-day living. These abilities ... are retained and used long after the individual has forgotten the detailed specifics of the subject matter taught in the schools. These abilities are regarded as one set of essential characteristics needed to continue learning and to cope with a rapidly changing world ... These higher mental processes are (also) important because they make learning exciting and constantly new and playful. (p. 13).

Bloom further observes that

> ... teachers in United States typically make use of text books that rarely pose real problems. These text books emphasize specific content to be remembered and give students little opportunity to discuss underlying concepts and principles and even less opportunity to attack real problems in the environments in which they live. The teacher-made tests (and standardized tests) are largely tests of remembered information ... (p. 13).

These two problems are related to each other. A student having knowledge in a non-integrated form would not be able to *correctly* solve problems, because according to Anderson (1980) problem solving requires a "goal-directed sequence of cognitive operations" (p. 257). A "basket of facts" lacks the needed structure to support the required sequence of operations.

1.2 Medical Education

A demand for reform is not new in medical education (Rothstein, 1987; McGuire et al., 1983; Jonas, 1978; Barrows & Tamblyn, 1980). A comprehensive report on the reform of medical education was published as early as 1910 by the Carnegie Foundation. This report was called the "Flexner Report." Jonas (1978) summarized the main features of this report, as follows

> <u>The most important task of the medical education process would be to teach the understanding and use of the scientific method.</u> Thus the didactic-lecture/rote-memorization method of teaching, and examinations would be of limited utility. Lectures would be used primarily for introductions to and summations of subject area. Examinations would be used more as learning experiences than as measures of performance. <u>The problem-solving method of instruction, i.e., the practical use of the scientific method, would be widely employed.</u> To the greatest extent possible, basic medical science teaching would be integrated with and relevant to clinical teaching. The use of the problem-solving method would, of course, greatly facilitate this integration (p. 213).

It can be noted that Flexner in his report was also most concerned with the two main problems mentioned in Section 1.1. After the publication of the Flexner report many different (in some cases quite radical) models of medical education was adopted by various medical colleges. Some of these models were: two-plus-two (Jonas, 1978), the systems approach (Rothstein, 1987), and problem-based learning (Barrows & Tamblyn, 1980). Some even went further to suggest the inverted curriculum approach (Barrows & Tamblyn, 1980). But as Jonas (1978) observes, the approach to teaching basic science instruction most condemned by the Flexner Report is still used in many medical schools: didactic lecturing, rote memorization, and frequent examinations.

A most recent comprehensive evaluation of the state of American medical education was published by the Association of American Medical Colleges (Rothstein, 1987). This report contains a set of 27 recommendations. At least 9 of the 27 recommendations pertain, directly or indirectly, to the teaching of the basic biomedical sciences (Michael, 1989). Again, the two problems of our current educational system mentioned above are prominent in these recommendations.

1.3 <u>Need to Facilitate Active Learning: A Solution</u>

In the conventional teaching format, instruction is viewed as a process of transmission (Wheatley, 1991). In this viewpoint, ideas and thoughts are communicated in the sense that meaning is packaged into words and “sent” to another who unpacks the meaning from the sentences. Here the learner is like an sponge. This passive view of learning is one of the major causes of turning students into memorizers and poor problem-solvers. According to Wheatley (1991)

> The workplace metaphor seems to describe activities in many classrooms. Students are engaged in exchanging performances for grades, much as a worker’s reward depends on their productivity. Students are “paid” for their products (assignments, tests) with praise and grades.... Learning is not a goal of such a work oriented environment, learning is a by-product of the “work” if it happens at all (p. 13).

Current research in cognitive science is changing our view of how people learn. As Resnick (1983) noted, in the last few years a new consensus on the nature of learning has begun to emerge. She further noted that “This emerging conception of learning has a direct bearing on how ... science ... can be taught most effectively” (p. 477).

Resnick (1983) described the evolving model of the learner in terms of a series of propositions. (1) Learners do not passively receive knowledge but rather actively build (construct) it. That is, as much as we would like to, we cannot put ideas in student’s heads, they will and must construct their own meaning. (2) To understand something is to know relationships. Human knowledge is stored in clusters and organized into schemata that people use both to interpret familiar situations and to reason about new ones. Bits of information isolated from these structures are forgotten or become inaccessible to memory. (3) All learning depends on prior knowledge. Learners try to link new information to what they already know in order to interpret the new material in terms of established schemata. (4) Successful problem-solving requires a substantial amount of qualitative reasoning. Good problem-solvers do not rush into applying a formula or an equation. Instead, they try to understand the problem situation; they consider alternative representations and relations among the variables. Only when they are satisfied that they understand the situation and all the variables in it in a qualitative way do they start to quantify it.

From these findings it can be argued that active forms of learning can provide a solution to learning a non-integrated knowledge base. Active learning can also promote the acquisition of problem-solving skills. But active learning requires the role of the teacher to be radically changed. Here the teacher should create an environment that encourages students to take responsibility for their own learning (Michael & Modell, 1993). This environment should be cooperative rather than competitive and here the teacher should be a *facilitator* rather than a transmitter of knowledge (Wheatley, 1991). Teachers also need to focus on the qualitative aspects of the problem situation (Resnick, 1983).

1.4 Tutoring: A Method of Facilitating Active Learning

One-on-one tutoring is a method that facilitates active learning in the student. In this method the student and the tutor collaborate in the process of instruction (Goodyear, 1991). Here the tutor provides individualized instruction and attention to the student (Wenger, 1987). Studies have shown that one-on-one tutoring is one of the most effective educational delivery methods (Bloom, 1984; Anderson et al., 1985; Woolf, 1986; Cohen et al., 1982). A study by Bloom (1984) shows that students involved in one-on-one tutoring seem to perform at about the 98th percentile as compared with students who are traditionally trained (via the group instruction method). Hence, if our educational system adopts tutoring as the educational delivery method then students will be active while learning. This would probably help them to build coherent models of domain from the subject matter content, which would not only help to enhance their understanding but also help to develop their problem-solving skills. So if tutoring is so advantageous why are students still learning in groups via the traditional classroom teaching method? As Galdes (1990) noted that “The answer is simple - there just aren’t enough skilled human tutors to do all of the necessary tutoring” (p. 2).

1.5 Machine Tutors: An Alternative to Skilled Human Tutors

One possible solution to the shortage of human tutors is to build computer-based tutoring systems that are as effective as human tutors. These computer-based tutors are often referred to as Intelligent Tutoring Systems (ITS).

It would be incorrect to suggest that intelligent tutoring systems will solve *all* the problems of current educational system (Woolf, 1986) but it is possible that these machines might solve some of the problems (e.g., see Section 1.1) that are responsible for the current disastrous state of education.

Research on ITS started in the 1970's (Clancey, 1992; Woolf, 1988a). Since then many systems have been built but only a few are in actual use (Clancey, 1992; Galdes, 1990). Studies have shown that tutoring by computers can be more effective than class room teaching (Anderson et al., 1989; Reiser et al., 1991). However, it appears that human tutors still outperform machine tutors (Merrill et al., 1992). These findings raise the following questions for the ITS community: Why are human tutors so effective, and how can we make machine tutors better? Also why are only a few of the tutors built so far in use (Galdes, 1990)?

The development of an ITS is a complex task (Woolf, 1988b) and requires a multidisciplinary approach (Kearsley, 1987). As Kearsley (1987) noted, the design and development of ITS "lie at the intersection of computer science, cognitive psychology, and educational research" (p. 3). In order to make this design and development process clearer, we will describe ITS from three different viewpoints: the conceptual view, the system view, and the physical view.

1.5.1 Conceptual View of ITSs. This view of ITS is concerned only with the conceptual issues underlying the target tutoring expertise. Two main issues dominate here: the nature of tutoring expertise and the variables influencing tutoring expertise.

Galdes (1990) claims that the goal of ITS research is "to build computer-based systems which emulate skilled human tutors" (p. 2). Although all researchers may not agree with this view of ITS development, studies have shown that human tutoring provides the best educational delivery method so far known (Bloom, 1984). Does this mean that tutoring by humans always provides the most effective method of instruction? Anania (1983), in a review of three studies of tutoring vs. group instruction, found that tutoring is not always better. Her review showed that group instruction is better than tutoring when the teachers are trained and the tutors are untrained. In a review of thirteen studies of tutoring vs. group instruction, Ellson (1976) found that only five of these studies clearly showed that tutoring was effective in improving cognitive skills. From this, Ellson concluded that

> The success of tutoring cannot be attributed to individual attention. If individual attention were the critical operating factor then all tutoring should be successful, but as the review has shown, only some is successful, perhaps less than half ... There is no magic in individual attention.

These studies raise the question of what makes human tutors effective? One explanation for the effectiveness of human tutoring is that it is the “skilled tutoring that provides the magic” (Galdes, 1990, p. 2). Skilled tutoring requires expertise in both the domain and in the process of tutoring (Khuwaja et al., in preparation (a)). A human tutor lacking either of these skills will not be able to perform optimally in a tutoring situation. Breuker (1988) has concluded that experienced users (domain experts) are not necessarily good tutors. This conclusion has also been drawn by Fletcher (1984). On the other hand, expert tutors lacking expertise in the domain are also not optimally effective because, as Jones et al. (1979) observed in their study, “no amount of (teaching) strategies can make up for the lack of knowledge in a subject matter.”

As mentioned above, a skilled human tutor acts as an expert in the domain (domain expert) and in the process of tutoring (expert tutor) while communicating with the student. The expert tutor is a composite role. In this capacity the human tutor *diagnoses* the student’s problems, *plans* the feedback and *communicates* it to the student. These three roles of the tutor have been identified since the early days of ITS research

(Wenger, 1987; Barr & Feigenbaum, 1982). But as Wenger (1987) observed, most of these early efforts “concentrate on some of these issues (roles) at the expense of others” (p. 14). The research methodology used in early studies was correct but those systems only began to investigate the tutor’s expertise in different domains (Galdes, 1990). Also it is relatively easy to isolate a role of the tutor for investigation and tempting to generalize findings over domains and tutoring situations. Almost all research efforts, so far, have ignored the question of *integration* between roles of the tutor or adopted a primitive view based on speculation.

If we want ITSs to be as effective as skilled human tutors then we must not only investigate the different roles of the skilled human tutor but also the processes integrating these roles. Stevens et al. (1982) have also emphasized this theme as

> In much of psychology, there has been a bias towards emphasizing highly general, domain-independent mechanisms that are supposedly central to the instructional process. Our work demonstrates that such a perspective is incomplete without a detailed consideration of domain-specific knowledge, its representation and its interaction with more general aspects of cognition (p. 13).

It is now well established that people use multiple representations of the physical world when interacting with it (Collins, 1985; Gentner & Stevens, 1983). A mental model in this context can be defined as “an internal representation of a physical system used by a person or a program to reason about processes involving that system” (Wenger, 1987, pp. 45-46). Although research on mental models is an active area in artificial intelligence (AI) and cognitive psychology, only recently have a few efforts been geared to emphasize its utility from a pedagogical standpoint. Most of these research efforts are concerned only with the learner’s viewpoint, i.e., what models the learner possesses and how these models can be tuned to yield the desired responses (Collins, 1985). An alternative to this educational approach is the question: How does the tutor use his/her multiple conceptual/mental models of the domain to remedy student misconceptions and to help learners build correct mental models of the domain?

Another variable only briefly considered in ITS research is the *environment* in which the tutor and the student communicate. This variable is of central importance to the researchers involved in the development of learning environments. It is true that in some human tutoring situations this variable is of little importance. For example, in one form of face-to-face tutoring the tutor and the student mainly use verbal means for communication. In this case only simple rules of verbal communication need to be followed. But this is not the case in all tutoring situations. In some tutoring situations the tutor and the student need to follow a complicated set of rules which, for example, determine the way different objects (e.g., charts, tables) are used while tutoring. These environments can also convey, implicitly, part of the problem-solving methodology that the tutor wants the students to discover. In these situations it is valuable to investigate how these environments are created by the tutor and how they impact on the various roles of the tutor.

The proper conceptual view of an ITS is literally incomplete without the consideration of appropriate variables influencing the *tutoring* expertise of the tutor. Ignorance of this aspect of the conceptual view of ITS is clear in the literature (e.g., see Galdes, 1990). Researchers are tempted by the “generality hypothesis” (White & Frederiksen, 1990) to generalize their findings, ignoring crucial variables (e.g., type of domain, educational setting, tutoring situation, goals of the tutor, knowledge level of students) affecting tutoring expertise. The result of this ignorance is theories having holes that can not be filled when these variables change! This is also one reason that after almost two decades of research, human tutors still out perform machine tutors (Merrill et al., 1992) and the majority of ITSs are inapplicable in real educational settings! It is true that we are still in the exploratory phase of ITS development and as Galdes (1990) noted, “currently, we are still building an empirical base of knowledge and descriptive theories of what skilled human tutors do” (p. 80). But if we ignore the true context of our research, by ignoring the variables making up the context, our knowledge base will be inconsistent and incomplete and retard the development of a unified theory of ITS development, if such a theory is possible.

Studies like these are essential in understanding the processes underlying the target tutoring expertise. The conceptual view allows us to concentrate on the theoretical aspects underlying tutoring expertise without getting bogged down in the system and physical views that are required to implement conceptual views in machine executable form on a computer. It would be incorrect to totally reject the influence of the system and the physical view on the conceptual view of ITS because ultimately we want a theory of tutoring expertise whose prescriptions can be implemented in a machine form.

1.5.2 System View of ITSs. An ITS is a program, a piece of software, and its purpose is to engage the student in an instructional activity. Unlike the conceptual view, which is concerned with natural tutoring expertise, the system view is concerned with the organization of tutoring expertise as a computational system. Here the behavior of the ITS results from the coordination of its components. The system view is influenced by the conceptual view. The characteristics of the system view, in turn, influence the physical view. The system view of the ITS can be conveniently divided into two subviews explained as follows.

1.5.2.1 System View 1: System Model. Through this view an ITS can be seen as an instructional system. Here the tutoring theory of the conceptual view is realized as a computational model, but still this view is not related to the actual implementation formalism needed to realize the system as a computer program.

An ITS viewed from this perspective deals with two major issues: curriculum and instruction (Halff, 1988). The curriculum issues involve the representation, selection, and sequencing of material to be presented to students, whereas instructional issues involve the presentation of that material to students. Halff (1988) noted that

> For teaching methods such as lectures, which are less dynamic than tutoring, both curriculum and instruction can be developed prior to delivery, with as much or as little accountability to instructional principles as the developers feel is needed. Tutoring systems afford no such luxury because a tutor, human or machine, is

> bound to tailor the selection, sequencing, and methods of delivering instruction to meet the ever-changing needs of individual students (p. 80).

Lesgold (1988) claims that "intelligent instructional systems developed to date have explicit representation of the target (domain) knowledge but at best only implicit representations of the curriculum knowledge, the scope and sequencing of lessons" (p. 118). A lack of this explicit representation of curriculum is one of the causes of sub-optimal performance of ITS.

Empirical studies have shown that human tutors perform a number of tasks that are hierarchically arranged (Goodyear, 1991). A proper execution of these tasks produces a successful management of an effective tutoring session. These tutorial tasks are usually classified at different levels of abstraction. One such classification comprises the major (core) objective level, exercise (or problem selection) level, unit level, lesson level, and discourse level. An ITS not only needs to explicitly represent its curriculum, but also needs to arrange it so that it can support a complete hierarchy of tutorial tasks.

The selection, sequencing and presentation of the curriculum to students by an ITS requires a sophisticated planning mechanism. The need for explicit representation of the curriculum adds further complications to this planning mechanism. This planning mechanism, regardless of the particular theory of tutoring on which it is based, should meet the ever changing needs of individual students. One of the currently popular opportunistic instructional planning mechanisms, which plans at the local discourse level, is MENO-TUTOR (Woolf, 1984). On the other hand, the IDE-INTERPRETER (Russell, 1988) attempts to represent plans at higher levels of abstraction, but this system lacks power at the local discourse level (Woo et al., 1991). Currently there have been attempts to combine local discourse planning with global (or higher level) planning (Woo et al., 1991). But again these systems lack explicit representation of the curriculum and fail to plan tutorial tasks at sufficiently high levels. At this point there is a need for a system model that uses a sophisticated planning mechanism (fueled by the tutoring prescriptions at the conceptual level) and also makes use of explicit curriculum and domain knowledge.

1.5.2.2 <u>System View 2: System Architecture</u>. This view brings the ITS a step closer to its realization in a machine executable form. Through this view an ITS can be seen as a software system. Software engineering principles shape this view of an ITS. But this view is still independent of the actual implementation formalism. Major concerns here include: the design of different modules and the communication between the modules.

1.5.3 <u>Physical View of ITSs</u>. An ITS seen through this view is a machine coded program. Here each module of the system view is transformed into a chunk of code that executes on a machine. Major concerns here are the selection of a hardware platform and the implementation formalism to realize the ITS as an executable software program.

1.6 <u>Goals of This Research</u>

This research is a part of the project that is primarily aimed at developing an Intelligent Tutoring System called CIRCSIM-Tutor (v.3). The knowledge domain of this system is cardiovascular (CV) physiology, specifically the baroreceptor reflex, that part of the cardiovascular system responsible for maintaining a more or less constant blood pressure (Berne & Levy, 1993). The main educational goals of this ITS are (1) that the students, using this system, acquire a qualitative, causal model of the cardiovascular system, and (2) that they learn a problem-solving method that enables them to solve any problem in the domain.

The primary goal of my research is to design and develop the Domain Knowledge Base, the Domain Problem Solver, and the Instructional Planner of CIRCSIM-Tutor (v.3). These components are designed to play two roles of a skilled human tutor - the domain expert and the pedagogy expert.

This research has developed a model of tutoring that, in a *problem-solving environment* helps the student to *integrate* his/her knowledge into a coherent qualitative causal model of the domain. The key feature of this model is that it uses multiple conceptual models of the domain in the process of facilitating knowledge integration. The primary tutoring style assumed by this model is Socratic dialogue (Wenger, 1987). This research adds to our knowledge in several areas.

(1) The design of an ITS development methodology. The three ITS viewpoints (see Section 1.5) correspond to the three major phases in the development of an ITS. I have used this methodology to develop the above mentioned components of CIRCSIM-Tutor (v.3).

(2) Analysis and development of the conceptual model of the domain and the pedagogy expertise. This includes an analysis of the interaction between the domain and the pedagogy expertise and its implication on the process of tutoring. Here I have also analyzed the influence of the tutoring environment on the process of tutoring.

(3) Development of the system model for the domain and the pedagogy expert.

(4) Development of the system architecture for CIRCSIM-Tutor (v.3).

(5) Implementation of the domain knowledge base, the domain problem solver, and the instructional planner.

1.7 Organization of the Thesis

This thesis consists of nine chapters. Chapter II describes a review of the literature. It concentrates on the early Intelligent Tutoring System (ITS) development approaches and various ITSs built as a result of use of these approaches. It first describes a classification of ITS. Next it reviews various ITS development approaches. Then a set of theories are described that view the tutor from different points of view (e.g., facilitator of knowledge integration). Next the mental model approach is described as the most recent and popular paradigm for knowledge representation in intelligent systems. This chapter then describes a set of important pedagogical issues relating to the design and development of an instructional system. Finally, this chapter ends with a brief description of one of the most popular knowledge based development methodologies, KADS. This methodology has had a significant influence on the ITS development methodology developed for this Research.

Chapter III describes the background of the research described in this thesis. A historical trace of the development of computer based medical systems to teach the functioning of the baroreceptor reflex is presented. This trace has been divided into two major periods: pre CIRCSIM-Tutor and CIRCSIM-Tutor eras. The pre CIRCSIM-Tutor era mainly deals with Computer Aided Instruction (CAI) systems (e.g., HEARTSIM) developed primarily at the Rush Medical College. During the CIRCSIM-Tutor era, a set of ITSs have been developed as a joint venture between the Illinois Institute of Technology and Rush Medical College. This chapter describes in detail the characteristics, capabilities, and research issues of most of the systems developed during these eras.

Chapter IV first describes the ITS development framework developed for this research. This framework combines the key features of a knowledge based system development methodology and an instructional system design methodology. Next, this chapter explains the usage of this methodology in the development of CIRCSIM-Tutor (v.3). Here knowledge acquisition is described as a modeling activity. This chapter ends with a detailed description of methods to capture raw expertise for this research.

Chapter V basically sets the stage for the remaining chapters of this thesis, beginning with a sketch of the theme of this research - a cognitive model of tutoring. This chapter describes the limitations of the tutoring models used in the earlier versions of CIRCSIM-Tutor. Next, it describes the scope and generality of the model of tutoring developed in this research. This model considers only the domain and the pedagogy roles of an effective human tutor. The pedagogy role here deals with both the pre-session and the in-session behaviors of the tutor. A detailed analysis of the pre-session behavior of our tutor in the keyboard-to-keyboard sessions is described. An analysis of the in-session behaviors of the tutor is left for the next chapter of this thesis.

Chapter VI describes the conceptual view of our model of tutoring. Here only the pedagogy and the domain expert roles are considered. This chapter starts with a detailed view of the in-session behavior of the pedagogy expert. Next, it describes the conceptual model for the domain expert. Then it describes an study that investigated the integration between these two roles of the tutor. This chapter ends with a description of the theoretical orientation of this model of tutoring. Here the behavior of the tutor is described using two metaphors - the jigsaw puzzle metaphor and the zoom-lens metaphor.

Chapter VII describes the system model for the CIRCSIM-Tutor (v.3). This model is an attempt to integrate the curriculum-based and the model-based themes of ITSs. The system model is also presented as a generic model that could be used to develop a system in any domain. Two views of this model are presented. In the first view the knowledge in an ITS is organized into three different dimensions: the planning dimension, the curriculum dimension, and the domain knowledge dimension. In the second view the system model consists of a set of tutoring spaces. This chapter then describes the contents of each tutoring space used in CIRCSIM-Tutor (v.3) in detail.

Chapter VIII describes the design of the architecture of CIRCSIM-Tutor (v.3) and the transformation of the system model into architectural components. The architecture of CIRCSIM-Tutor (v.3) is based on an object-oriented methodology. This chapter then describes the design and implementation of each architectural component developed during this research.

The thesis concludes in Chapter IX with a discussion of the significance of this research, describes some of its limitations, and gives suggestions for future research.

CHAPTER II

LITERATURE REVIEW: EARLY APPROACHES AND SYSTEMS

2.1 A Classification of Intelligent Computer-Based Educational Systems

It is now commonly accepted that research on Intelligent Tutoring Systems (ITS), also sometimes called Intelligent Computer Aided Instructional (ICAI) systems, started as a distinct approach with a dissertation by Carbonell (1970a) and with his system SCHOLAR (Collins et al., 1975; Carbonell, 1970b; Barr & Feigenbaum, 1982; Clancey, 1987a). Since this beginning, this research has developed in many directions (Kearsley, 1987; Galdes, 1990). Broadly speaking, two major schools of thought have evolved and produced two different types of system: learning environments and active teaching systems (Goodyear, 1991; Galdes, 1990). As Goodyear (1991) noted, the builders of learning environments have adopted “models which favor learner-directed exploration, negotiation of purpose, or the acquisition of higher-level cognitive skills” (p. 9), whereas the builders of active teaching systems “have adopted pedagogical models which could be described as instructional, goal-directed, system-led or content-driven” (pp. 8-9).

The educational philosophies underlying these two types of system form a continuum (Figure 2.1). At one extreme of this continuum lie the learning environments that support free exploratory (or discovery) learning, e.g., the LOGO system (Papert, 1980). The fundamental epistemological concept underlying LOGO is "it is more important to help children learn how to develop and debug their own theories than it is to teach them theories we consider correct" (Wenger, 1987, p. 125). A repeated conclusion from experience with these environments in practice is that students tend to require some additional assistance or guidance to take full advantage of their exposure. Another type of learning environment is called a reactive learning environment (Barr & Feigenbaum, 1982). These systems are designed to improve on the discovery learning approach by making more of the underlying domain and its structure visible to the student (Brown, 1983). With discovery learning, the student is on his own to discover any structure inherent in the learning environment, but, as Brown (1983) noted, "much of the time he (the student) will appear to be wandering aimlessly down the garden path in search of some insight." In contrast, reactive learning environments are designed to prevent too much wandering by the student.

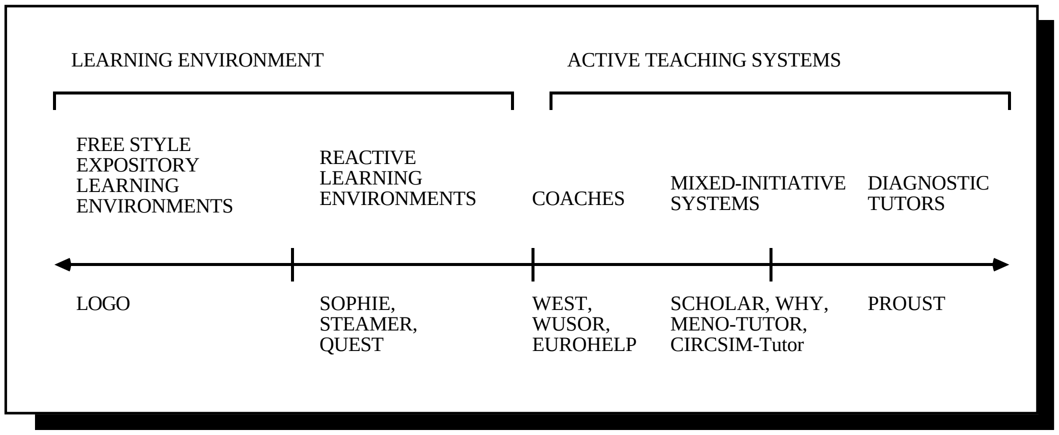


Figure 2.1 A Classification of Computer-Based Educational Systems

The second major category of ITS is active teaching systems (see Figure 2.1). These systems deliver instruction and monitor guided practice. In other words these systems make the structure of the domain visible and accessible and also help to lead the student through his/her domain knowledge. Active teaching systems are subclassified as: coaching systems (Breuker, 1988; Barr & Feigenbaum, 1982), mixed-initiative tutors (Barr & Feigenbaum, 1982; Wenger, 1987), and diagnostic tutors (Kearsley, 1987).

As the name suggests, coaching systems view the instructional system as a coach. Here the system's main task is "to look over the student's shoulder and decide whether to help the student around a particular pitfall or let him succumb to the pitfall so that he can learn to detect when he is on the wrong path" (Galdes, 1990, pp. 20-21). In other words a coaching system observes the student’s performance quietly and provides advice that will help the student to perform better (Kearsley, 1987).

Mixed-initiative tutors engage the student in a two-way conversation and attempt to teach the student via the Socratic method of guided discovery (Kearsley, 1987). As Galdes (1990) noted, the Socratic method here “refers more to a basic question-answering and dialogue approach (Resnick, 1977) than to the original method developed by Socrates (Jordan, 1963; Hayman, 1974; Plato, 1974)” (p. 20).

Diagnostic tutors, according to Galdes (1990), “debug a student’s work once the student has completed the problem or reaches an impasse” (p. 21). These programs are driven by a bug library that identifies the misconceptions that students may have in solving a problem (Kearsley, 1987).

Goodyear (1991) noted that the advocates of learning environments nowadays “tend to distance themselves from mainstream research, for example, by rejecting the most widely used descriptive term for an AI-based teaching system - intelligent tutoring system” (p. 9). From now on, for the purposes of this proposal, I will restrict the term ITS to mean just active teaching systems.

2.2 Research Approaches to the Design of Active Teaching Systems: Why Study Human Tutors?

Tutoring is a *method* of educational delivery. In order to use this method one needs to make many decisions, for example, (1) what type of knowledge needs to be tutored? Here possibilities are: factual knowledge, procedural knowledge, conceptual knowledge, basic principles, or a combination of these. (2) What is the nature of the educational setting? For example, will tutoring be a part of or peripheral to the course under consideration. If it is a part of the course then how much of the curriculum will it cover? On the other hand, if it is peripheral to the course then on what aspects of the curriculum will it concentrate? (3) What is the educational level for which this tutoring method needs to be used? This could range from primary to professional education.

These decisions, in a way, are variables determining the nature of tutoring used in an educational system.

Tutoring studies performed by, for example, Bloom (1984), Collins et al. (1975), Fox(1993), Putnam (1987), Littman et al. (1985), and Galdes(1990), have used different variables and there are no systematic criteria yet available to compare these studies! In such a situation it is difficult to generalize the effectiveness of a study under one set of conditions to a similar study having a different set of conditions.

A survey of literature indicates that many of the researchers agree that tutoring is the most effective educational delivery method. One reason for this general agreement is that certain studies (for example, performed by Bloom (1984)) have clearly demonstrated the effectiveness of tutoring compared to other methods of educational delivery. These studies have provided a basis for the first-order generalization that tutoring as a method of educational delivery, without regards to the variables on which it depends, is most effective. I absolutely agree that the field of ITS is now sufficiently mature to strive for a more detailed classification of variables on which tutoring depends. But for the purposes of this proposal I will stick with the first-order generalization of the effectiveness of tutoring mentioned above.

Studies have shown that tutoring by *humans* is more effective than currently known educational delivery methods (Bloom, 1984; Cohen et al., 1982). One goal of ITS research is to make machine tutors at least as effective as human tutors (Galdes, 1990). However, it appears that human tutors still outperform machine tutors (Merrill et al., 1992). In light of these findings, the obvious approach to improving ITS design is to study the behavior of the human tutors. But, as this section will report, this is only one of the approaches used.

2.2.1 <u>Multiple Expert Metaphor</u>. Research on tutoring has shown that human tutors, while in the process of tutoring, perform a number of tasks (Collins et al., 1975; Woolf & Cunningham, 1987; Littman, 1991; McArthur et al., 1990). These tasks can be grouped into the following major activities: act as an expert in the domain, act as an expert in the process of tutoring, act as an expert in the process of diagnosing the student's problems, and act as an expert in the process of communicating his/her responses to the student. None of the studies cited above explicitly tested the competence of their tutors before the experiments, and there is no agreed upon method of testing available. The word *expert*, used in listing these varied tutor activities I introduced, is only to emphasize the relative competence of the tutors with respect to the students participating in the experiments (i.e., these tutors are not necessarily experts compared to the talent available in the field of education).

The most widely accepted model for the design of the ITS is based upon the multiple expert metaphor (Breuker, 1990; Self, 1988; Wenger, 1987; Polson & Richardson, 1988). According to this metaphor, an ITS consists of a set of experts that communicate with each other and coordinate their activities to create effective tutoring behavior. The most commonly used components that take part in this process are the domain expert, student expert (or student modeler), pedagogical expert, and communication expert. The domain expert characterizes the knowledge and strategies needed for expert performance in a domain (Bonar, 1984). The student expert is also called a student modeler because it represents the tutor's estimate of the student's understanding of the material to be taught (Barr & Feigenbaum, 1982). In other words this expert "uncovers a hidden cognitive state (the student's knowledge of the subject matter) from observable behavior" (VanLehn, 1988, p. 55). The pedagogy or tutoring expert, using a theory of tutoring, provides assistance to the student, monitors and criticizes the student, and selects problems and remedial material for the student (Halff, 1988; Wenger, 1987; Clancey, 1987a). The communication expert processes the flow of communication in and out of the system. Wenger (1987) noted that "whereas the pedagogical module (expert) decides the timeliness and content of didactic actions, the ... (communication expert) takes care of their final form" (p. 21).

It is interesting to note that this multiple expert metaphor for the ITS development matches with the roles that are responsible for the activities the human tutors perform in a one-on-one tutorial setting. It is immaterial, from the point of view of this proposal, whether the study of human tutors motivated the multiple expert metaphor for ITS design or vice versa. But it is obvious that studying human tutors certainly will enhance our understanding of the process underlying their effective behavior and help us to improve the design and the behavior of ITS. Such empirical studies will also naturally support the system model (see section 1.5.2) of the ITS.

For the sake of consistency I will use the multiple expert metaphor to describe, also, the behavior of the human tutors. But in order to distinguish the behavior of the human tutor from the ITS, I will use the word *role* for the human tutor and the word *component* for the machine tutor (ITS). With this distinction in mind, the behavior of the human tutor can be described as a collection of four *roles* (domain expert, pedagogy expert, student expert, and communication expert) which he/she performs while interacting with the student in a tutoring situation. On the other hand, the conceptual model of the machine tutor is composed of four *components* (domain expert, pedagogy expert, student expert, and communication expert), which interact to create its behavior.

This discussion would certainly be incomplete without mentioning a recent criticism of the multiple expert metaphor. According to Breuker (1990)

> In the ITS literature (e.g., Wenger, 1987) a confusion of functions, types of knowledge and agents is very persistent. In general it is explained that an ITS consists of a “classical” triad (Self, 1988) of a domain, student and didactic expert. In fact the roles of these “agents” overlap: in particular in diagnosing. This overlap is not a serious problem, unless it is preserved in the functional decomposition, and worse: in the architecture of an ITS. Because this metaphor is used over and over again it may have been the cause for so many inarticulated ITS architectures (pp. 50-51).

This criticism of the multiple expert metaphor raises a very fundamental issue for ITS research. I believe that this metaphor is helpful at the conceptual level (see section 1.5.1), but I also agree with Breuker that this metaphor should not be carried over to the implementation level. This is the approach I have taken for this research. For further discussion of this issue see section 3.3.

2.2.2 <u>Interactivity Between the Components of Multiple Expert Metaphor: The Real Strength of an ITS</u>. As I said earlier, the multiple expert paradigm is the one most popularly used in the design of the ITS. But as Park et al. (1987) noted, because of the size and complexity of most ITS, researchers typically focus on only one or two components of the ITS at a time and ignore the others. For example SCHOLAR (Carbonell, 1970b) and WHY (Collins et al., 1975) emphasize knowledge representation (domain expert) and tutorial dialogues (pedagogy expert). BUGGY (Brown & Burton, 1978), DEBUGGY (Burton, 1982), and PROUST (Johnson & Soloway, 1984a) emphasize student modeling issues (student expert). MENO-TUTOR (Woolf & McDonald, 1985) emphasizes tutorial discourse strategies (pedagogy expert and communication expert). This partial and isolated approach was justifiable in these early systems because these research efforts only started to unfold the nature of the processes underlying effective but complex tutoring behavior.

The multiple expert metaphor is a powerful modularization tool which breaks a complex behavior into its components to facilitate understanding. But one drawback of this approach is that it puts too much emphasis on the notion that sum of the components makes up the whole. This idea is nicely explained by Burns & Parlett (1991)

> It is difficult to separate the dancer from the dance; nevertheless in Foundations of Intelligent Tutoring Systems (Polson & Richardson, 1988), the fundamental anatomy was used to discuss research issues within each of the separate components ... Because designing intelligent tutors is such an interdisciplinary activity, attending to the anatomy piece by piece often ignores the synergy that a wholly integrated system could achieve (p. 2).

Agreeing with Burns & Parlett (1991), I think now it is time for the ITS research community to focus on a new interpretation of the multiple expert metaphor, which not only emphasizes the individual components but also the interactivity between them which is “the real strength and centerpiece of individualized instruction necessary to ITSs”

(Burns & Parlett, 1991, p. 3). Similar arguments have been made by Goodyear (1991). According to Goodyear (1991), many researchers would now characterize "the central problems as being to do with integration or interdependence - that knowing about the BITS (Bits of an ITS) will not tell us how to put them together, or that the nature of each of the BITS is strongly determined by its relation to the whole" (p. 9). One obvious method of inquiry into this aspect of tutoring behavior is to study human tutors because they provide an excellent example in which all the roles are integrated to yield effective tutoring behavior.

2.2.3 Approaches to the Inquiry of Tutoring Behavior. A number of approaches have been used to develop the individual "experts" of an ITS. For the sake of clarity, I will discuss these approaches only in the context of developing the pedagogy expert of the ITS. As I said earlier, the pedagogy expert of an ITS using a model/theory of tutoring, provides assistance to the student, monitors and criticizes the student, and selects problems and remedial material for the student (Halff, 1988; Clancey, 1987a; Wenger, 1987). Broadly speaking three approaches have been used to develop a model/theory of tutoring for the pedagogy expert of the ITS (VanLehn, 1993).

2.2.3.1 Approach Based on Experience And Common Sense. In this approach the model/theory of tutoring is based on the author's experience (VanLehn, 1993) or observations of tutors (Galdes, 1990). These theories consist of heuristics, such as "Do not interrupt too often," and " Follow up an abstract definition with several examples and at least one negative example" (VanLehn, 1993, p. 3). One of the major objections to this approach is, since these theories are not based on careful empirical work, these theories are probably incomplete and possibly incorrect. Examples of systems that used this approach are WEST (Burton & Brown, 1979), which teaches arithmetic skills, WUSOR (Goldstein, 1982), which teaches logic and probability, and GUIDON (Clancey, 1987b), which teaches diagnosis and therapy prescription for infectious diseases. Galdes (1990) has given two more reasons for not using this approach. (1) <u>This approach can contradict facts:</u> The initial version of PROUST (Johnson & Soloway, 1984b) was developed using this approach. An evaluation of this early version indicated that the system was a total failure (Johnson & Soloway, 1984b). One reason for this failure was that the method of diagnosis used by the system was not able to handle real student input. The new version of this system used a theory of programming bugs based on empirical studies of the kinds of programming errors real students make (Spohrer & Soloway, 1986). This example shows that theories based upon experience or common sense may not always be right. (2) <u>This approach may not give us the right context:</u> According to Galdes (1990) "A second problem with intuitions is that the "rightness" or "wrongness" often depends on the *context* where we want to apply it." (p. 30). For example in WEST (Burton & Brown, 1979) twelve pedagogical principles are used. One of these principles states "Do not tutor on two consecutive moves no matter what" (Burton & Brown, 1979). The purpose of this principle is "to keep the student's interest from being destroyed" (Galdes, 1990, p. 32). Here Galdes (1990) argues that this principle is fine for a game situation where students may not be highly motivated. But, is this principle also equally applicable for a system which teaches students how to deal with life threatening issues? The fact is that the right context for applying such principles can not be defined without empirical evidence.

2.2.3.2 <u>Approach Based on Rigorous Experimentation</u>. In this approach a careful experimentation is conducted to build a prescriptive theory of tutoring. This approach deals with simple hypotheses which when proven yield lists of heuristics such as "Immediate feedback is better than delayed feedback" (VanLehn, 1993, p. 4). This is a slow approach but probably the best one in the long run to understand the process of tutoring (VanLehn, 1993). But, as VanLehn (1993) noted, this approach tends "not to produce a unified theory of how to tutor, at least not very quickly" (p. 4).

2.2.3.3 <u>Approach Based on Studies of Human Tutors</u>. In this approach, researchers study human tutors to develop a model/theory of tutoring. I think this is the best approach to develop an ITS. Research has shown that tutoring by humans is the best method of educational delivery (Bloom, 1984). Hence basing the design of the ITS on these empirical studies will increase our chance of building a system that is as effective as human tutors. Some ITSs that were based upon the study of human tutors are SCHOLAR (Carbonell, 1970b), WHY (Collins et al., 1975), MENO-Tutor (Woolf & McDonald, 1985), PROUST (Johnson & Soloway, 1984a), EUROHELP (Breuker, 1990), and TP (Littman, 1991).

As we have seen in Section 2.2.1, at the conceptual level the complex behavior of the human tutor and the ITS can be decomposed into similar roles/components using the multiple expert metaphor. This means that the study of human tutors can be applied directly to the design of ITS at the conceptual level. Another advantage of studying the behavior of human tutors is that the integration between the multiple experts of an ITS, a largely ignored aspect of ITS, can be explored.

2.2.4 How "Expert" Should the Human Tutor Be? Although studying human tutors is the best approach to the design of the ITS, one important question one still needs to ask is: how expert should the human tutor be to participate in the experiment?

Goodyear (1991) states that "the pressing need in the ITS field is not for knowledge about optimal ways of teaching, but for knowledge about how to do any kind of teaching" (p. 13). A survey of the literature on tutoring systems indicates that most of the researchers would agree with Goodyear. The tutors in most of the empirical studies performed either lacked expertise in the process of tutoring or in the domain knowledge.

Collins et al. (1975) performed an empirical study of human tutoring to improve the performance of the SCHOLAR system. Four tutors and six students participated in this study. Collins et al. (1975) reported that the two of the tutors who participated in the experiment "have extensive teaching experience at the college level, though neither has taught geography (the domain of SCHOLAR system). The third and fourth tutors each taught only one session, and did not prepare nearly as extensively as the first two tutors.

The students were employees at BBN" (p. 54). To develop an ITS for the domain of introductory computer programming for the Lisp programming language, Anderson et al. (1985) performed an study of human tutors. McKendree et al. (1984) reported that in this study five of the tutors were psychology and computer science students "who had prior experience as teaching assistants or private tutors for Lisp." The tutors participating in the Fox (1993) study were graduate students in chemistry, physics, computer science and mathematics who had all worked as tutors before.

Most of the human tutoring studies reported in the literature, also, do not explicitly state the level of expertise of their subjects. Admittedly "it is certainly true that it is very difficult to get universal agreement on the quality of a teaching performance, even harder to get agreement on the validity of the knowledge on which that performance may be based" (Goodyear, 1991, p. 13). But, for example, studies performed by Anania (1983) and Ellson (1976) concluded that tutoring by humans is not always better than group instruction! Galdes (1990) proposed that the effectiveness of human tutors lies in the skills they have in the domain and in the process of tutoring. A human tutor lacking either of these skills will not be able to perform optimally in a tutoring situation (Anderson, 1988; Breuker, 1988; Fletcher, 1984; Jones et al., 1979). In the light of these findings it is imperative that human tutoring studies should explicitly state the level of expertise of their subjects and the standards by which this expertise are measured. This will help ITS research move towards a long term goal of making ITS's at least as effective as the best human tutors.

The literature on studies of differences between novice and expert teachers (e.g., Berliner, 1986; Leinhardt & Greeno, 1991) also have shown that expert teachers, compared to the novices, possess a variety of skills that make the process of teaching much more effective. Berliner (1986) provided a list of reasons for studying expert teachers. He further states that "the design of an intelligent tutoring system requires versatile output. That is, expert tutoring systems need a wide range of instructional options to choose from and some decision rules about when to use these options. Expert teachers are likely to have more of that kind of knowledge than ordinary teachers" (p. 6).

This section will be incomplete without mentioning some of the problems in studying human tutors. Berliner (1986) and Galdes (1990) contain a comprehensive coverage of these issues. I describe them briefly here for the sake of completeness. The first two problems deal with the methodology used in the observation of human tutors.

I) <u>Methods are Not Well-Defined</u>. Galdes (1990) argues that human experts possess tactical knowledge and since this knowledge can not be easily articulated, researchers involved in the study need to rely on other methods of getting this knowledge. According to her, "previous studies of human tutors demonstrated that no standard methods exist for uncovering this tacit knowledge" (p. 67). Analyses of data obtained through some methods also vary for two reasons. "First, the analysis method used depends on which specific variables the researcher wants to study ... Second, the analysis method varies according to the level of detail desired in the result." As a result each new study of human tutors needs to develop its own methodology.

II) <u>Methods are Time Consuming</u>. The knowledge acquisition is usually regarded as the most time consuming process in the design of an AI system (Wielinga et al., 1987). An ITS, being an AI system, is not an exception here.

III) <u>No Definite Criteria are Available for the Identification of Expert Tutors</u>. According to Berliner (1986) there are no definite and objective criteria available to identify an expert human tutor. This situation exists at all levels of educational practice.

IV) <u>The Confounding of Experience and Expertise</u>. Berliner (1986) states that

> Another problem we encountered occurred because mere experience is simply not believed by most people to correlate highly with expertise in pedagogy. Perhaps this is because our problems are so ill structured. Perhaps it is due to the lack of external criteria. But for whatever reasons, the problems of studying expertise in pedagogy are harder than in some other fields because of the widespread belief that we need to separate expertise from experience and to study how experience

changes people without necessarily turning them into experts. This is not easy (p. 9).

V) **<u>What Knowledge Systems are Used in Pedagogy?</u>** According to Berliner (1986), another problem in studying human teachers is in "understanding what domains of knowledge are used by expert teachers in accomplishing their tasks" (p. 9). Some researchers argue that the subject matter knowledge is important. Others argue that the teacher's personal knowledge of self is important (e.g., Lampert, 1984). "One scholar (Elbaz, 1981) has identified dozens of domains of knowledge that are drawn on by teachers to accomplish their tasks" (Berliner, 1986, p. 9).

2.3 The Teacher as a Facilitator for Knowledge Integration: The Current Theoretical Standpoint

This section describes the current theoretical standpoints of the teaching philosophy used in ITS research. This discussion is based upon the ideas of Ohlsson (1991) who described three theories of teaching that are relevant for ITS research and "form a sequence, in the sense that each theory is a response to an inherent problem in the preceding theory" (p. 26). These theories are described briefly as follows.

2.3.1 **<u>The Facilitation of Knowledge Transfer</u>**. Ohlsson (1991) called the "facilitation of knowledge transfer" theory as the first-order theory of teaching. This theory "views teaching as the communication of subject matter" (p. 25) and paints the traditional view of pedagogy. Although current ITS research does not subscribe to this point of view, much of the school practice still rests on this theory of pedagogy (Wheatley, 1991). From this viewpoint ideas and thoughts are communicated in the sense that meaning is packaged into words and sent to another who unpacks the meaning from the sentences. Here the learner is like an sponge (Wheatley, 1991). According to Ohlsson (1991) "this theory of teaching is radically inadequate" (p. 27). A teacher using this theory needs to make two types of decisions: what to say and how to say it. These decisions require the teacher to have "a codification of the subject matter, and he or she needs to know effective methods for the presentation of each unit of the subject matter" (Ohlsson, 1991, p. 28).

The first-order theory of teaching "implicitly assumes that the teacher knows the extent of the student's previous knowledge" (Ohlsson, 1991, p. 34). Current educational practice does not rest on individualization of instruction. Also from Lesgold's (1988) two fundamental laws of instruction we know that "not everyone who passes a test on a topic knows what appears to have been tested," and "not everyone who fails a test on a topic lacks the knowledge that appears to have been tested" (p. 118). It follows that in order for the teacher to have an accurate estimate of previous knowledge, he or she must diagnose the knowledge state of the individual student. This requirement gives rise to the need for the second-order theory of teaching.

2.3.2 The Facilitation of Knowledge Integration. According to Ohlsson (1991), the second-order theory of teaching says that "to teach is to correct the learner's mental representation of the subject matter" (p. 35). According to this theory the learner possesses some representation of the subject matter but this representation is either incomplete or incorrect or both. "The job of the teacher is to provide remediation for the discrepancies between the learner's representation and the complete and correct representation" (Ohlsson, 1991, p. 35). In other words, the goal of teaching is that the learner ultimately integrates his/her view of the domain into a correct, coherent, and desired model of the domain. This theory makes up the current view on which almost all ITS are based (Ohlsson, 1991). The major theme of this theory is the process of cognitive diagnosis, which can be described as the process of "inferring what the student does and does not know on the basis of his or her performance on some diagnostic task" (Ohlsson, 1991, pp. 34-35).

In accordance with this theory, the instruction generated by the teacher should be tailored to the knowledge state of the learner. A presentation of the subject matter about a particular misconception, for example, is not enough because different students need different presentations for the same misconception. The problem with this theory is that "the description of a deviation between the student's mental representation of the subject matter and the correct representation does not entail any conclusion about which tutorial message might cause the learner to correct that deviation" (Ohlsson, 1991, p. 42). Ohlsson (1991) concluded that a learning theory can only drive the content of a needed tutorial message. This conclusion gives rise to the third-order theory of teaching.

2.3.3 The Facilitation of Knowledge Construction. The third-order theory, according to Ohlsson (1991), paints a future view of teaching theory. According to this theory, teaching is the process of facilitating knowledge construction. In other words "to teach is to help the learner improve his or her current world view" (p. 43). This theory, although most promising, is not yet sufficiently fully developed to help in building ITSs. Ohlsson (1991) defined productive learning sequences as a sequence of learning events (e.g., to replace a vague idea with a clearer one, to generalize an idea) which leads to a better view of the subject matter. "To teach is, in this theory, neither to communicate the subject matter, nor to remedy cognitive errors, but to arrange situations in which productive learning sequences are likely to occur" (Ohlsson, 1991, p. 44).

2.3.4 Jigsaw-Puzzle Metaphor. In this section I will elaborate on the second-order theory of teaching because it is most commonly used in the current generation of ITSs. The research reported in this proposal is also based on this theoretical standpoint. I will also introduce the jigsaw-puzzle metaphor to further facilitate the understanding of this theory.

The goal of the teacher, in the second-order theory, is to correct the learner's mental representation of the subject matter. Here the teacher provides remedial material so that the discrepancies between the learner's representation and the complete and correct representation can be removed. The teacher using this theory needs to know how the subject matter knowledge is encoded in the learner (at some particular moment in time)? Ohlsson (1991) listed a set of four types of actions that a teacher needs to make:

(1) “Which subject matter topic is to be taught?” (2) “How does an expert represent that topic?” (3) “Which deficiency does the mental representation of this student suffer from at this moment in time?” and (4) “How can that deficiency be remediated?”

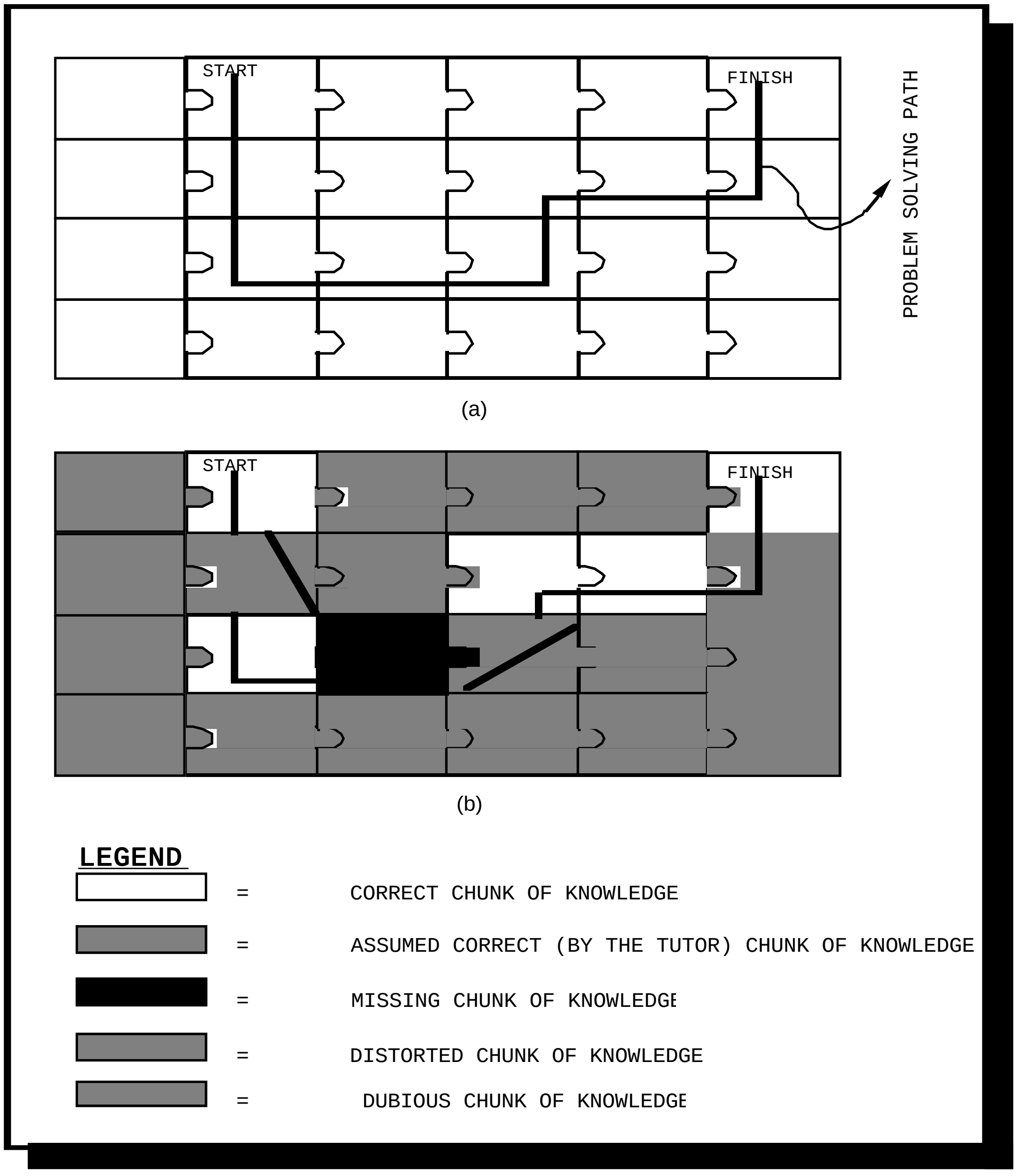

START
FINISH
PROBLEM SOLVING PATH
(a)
START
FINISH
(b)
LEGEND
= CORRECT CHUNK OF KNOWLEDGE
= ASSUMED CORRECT (BY THE TUTOR) CHUNK OF KNOWLEDGE
= MISSING CHUNK OF KNOWLEDGE
= DISTORTED CHUNK OF KNOWLEDGE
= DUBIOUS CHUNK OF KNOWLEDGE

Figure 2.2 Jigsaw-Puzzle Metaphor, (a) Mental Snapshot of the Tutor, (b) Mental Snapshot of the Student

The activities of the teacher using this theory can be visualized as if he/she is solving a jigsaw-puzzle (see Figure 2.2). This puzzle is analogous to the learner's mental representation of the subject matter (Figure 2.2 (b)). The puzzle solving process starts with a partially completed puzzle, representing the learner's mental representation at that point in time. This process is partially guided by the teacher's knowledge of the domain (Figure 2.2 (a)). The job of the tutor is to identify (i.e., perform cognitive diagnosis) on pieces of the puzzle (chunks of knowledge), which are missing or/and correctly and incorrectly placed.

After this piece by piece identification process, the teacher needs to put all the pieces together correctly (integrate them into a model of the domain) to complete this process of puzzle solving. Unlike a person solving a jigsaw puzzle, the teacher in real life does not have physical access to the student's mental state of the subject matter knowledge. He/she can only, using various teaching actions (e.g., asking questions, providing a summary), help the student integrate his/her knowledge into a coherent, correct and desired mental model of the domain. Although second-order teaching theory is more sophisticated than any metaphor of it can portray, the jigsaw-puzzle metaphor captures the essential ingredients of this theory. Another reason for introducing the jigsaw-puzzle metaphor is that it will help me explain, in latter chapters, the tutoring model developed in this research. I will refer to the second-order theory as the knowledge integration theory of tutoring.

2.4 Mental Models and Their Multiple Conceptualization: Current Wave in Knowledge Representation for Physical Systems

From the beginning, ITS research has concentrated more on the subject matter knowledge than other aspects of the teacher's expertise (Wenger, 1987). According to the multiple expert metaphor (see section 2.2.1), this knowledge is contained in the domain expert component of the ITS. Anderson (1988) called this component the backbone of the ITS. He further states that "a powerful instructional system cannot exist without a powerful body of domain knowledge" (p. 21). Artificial Intelligence provides a number of paradigms to represent this knowledge. In this section we will concentrate on one paradigm of knowledge representation, one that is currently a principal focus of ITS research, qualitative representation of mental models for physical systems.

2.4.1 <u>Knowledge Representation Hypothesis vs. Situated Nature of Knowledge: AI Might be Coming to a Decision Point</u>. Artificial intelligence systems, including ITSs, deal with knowledge, hence one major concern of these systems is to represent knowledge. In fact knowledge representation is one of the oldest and most active areas of AI research (Brachman & Levesque, 1985; Reichgelt, 1991). Some of the most popular formalisms of knowledge representation are formal logic-based representations (Moore, 1985), procedural representations (de Kleer et al., 1985), production systems (Davis et al., 1985), semantic networks (Brachman, 1985), frame-based representations (Minsky, 1985), qualitative reasoning (Brown & de Kleer, 1984; Forbus, 1984; Gentner & Stevens, 1983), and connectionism (Rumelhart et al., 1986).

Besides the many differences between these formalisms, it has been agreed that they (except connectionism) are based on a methodological assumption that Brian Smith (1985) has called the knowledge representation hypothesis. This hypothesis states that

> Any mechanically embodied intelligent process will be comprised of structural ingredients that a) we as external observers naturally take to represent a propositional account of the knowledge that the overall process exhibits, and b) independent of such external semantical attribution, play a formal but causal and essential role in engendering the behavior that manifests that knowledge (p. 33).

In other words, any system, whether it be human or artificial, that shows intelligent behavior, is assumed to contain a knowledge base as a substructure. The knowledge base is a more or less direct encoding of the knowledge that the system has available. The knowledge base is manipulated by a separate substructure, which is often called the inference engine. The inference engine processes the symbols in the knowledge base in order to generate intelligent behavior.

The connectionist approach, on the contrary, challenges this hypothesis. This approach, unlike other approaches that deal with the modeling of mind, model the brain to capture the intelligent behavior (Rumelhart et al., 1986). Most of the knowledge representation formalisms used in ITS research are consistent with the knowledge representation hypothesis.

Is there any relationship between knowledge representation formalisms (or more generally the knowledge representation hypothesis) and current teaching/tutoring theories? So far not much attention has been devoted to this inquiry, but with the current up coming theories of situated cognition (Brown et al., 1989a; Brown et al., 1989b; Brown, 1990), I think we in the ITS community sooner or later need to reinvestigate the fundamental assumptions behind the computational approaches to tutoring.

Most of the AI community, as Brachman & Levesque (1985) noted, agree that “it is in the knowledge itself that the power lies” (p. xv). As a consequence it is believed that the power of an ITS resides in the detailed explicit knowledge base that represents the system’s understanding of the process of tutoring in its domain. This belief promotes the assumption that knowledge is a “integral, self-sufficient substance, theoretically independent of the situations in which it is learned and used” (Brown et al., 1989a, p. 32). Situated cognition theories reject “the idea that human memory is a place where representations are stored ... knowledge does not consist of structures that exist independently in the mind ... knowledge, as a capacity to behave, develops during the course of interactivity” (Clancey, 1992, pp. 23-24).

Much research investigating situated features of cognition remains to be done (Brown, 1990) before it can radically challenge and change current educational practices and the direction of ITS research. Researchers like John Seely Brown, Allan Collins, William Clancey, who are also the pioneers of ITS research, are most enthusiastic about this new approach to the study of cognition and learning (Brown et al., 1989a; Collins et al., 1989; Clancey, 1992). The following paragraph from (Brown, 1990) reflects their vision of future ITS research.

> At present, the ITS community confronts a profound and exciting challenge. As a result of our central concern with learning, we find ourselves at the heart of an emergent epistemology. It is, therefore, members of this community who are most likely to find themselves uncovering better and much needed, models of the architecture of cognition, because it is this community that is most closely coupled to - or situated in - the fully blooded complexity of human learning activity. Thus, if we meet this challenge correctly, it may well be that, instead of ITS being merely one subset of the overall schema of AI, we will, instead, find that it is AI that becomes one subset of the overall schema of ITS (pp. 280-281).

2.4.2 <u>Qualitative Models of Physical Systems: Mental Models</u>. A knowledge representation formalism that is concerned with the cognitive modeling of physical systems is currently getting greater attention in AI and cognitive psychology (Wenger, 1987). It is thought that the knowledge used in this formalism underlies our ability to mentally simulate and reason about dynamic processes (Anderson, 1988). The models of physical systems resulting from a use of this formalism are called mental models or conceptual models.

As mentioned earlier, the research on mental models interests both the AI and the cognitive psychology communities. However, the goals of these two communities are not necessarily the same (Anderson, 1980; Winston, 1984). As a result, a term may mean different things to different people in these communities. The major goal of this section is to clarify the different terms used in the context of mental model research before I review some of the major efforts in this area.

I will use Norman's (1983) classification of concepts used in mental model research. According to Norman we need to distinguish between four different things. (1) The *target system* is the system that a person is learning or using. (2) The *conceptual model* that is developed to provide an accurate, consistent and complete representation of the target system. (3) *Mental models* are "naturally evolving models. That is, through interaction with a target system, people formulate mental models of that system"

(Norman, 1983, p. 7). (4) The scientist's conceptualization of a mental model is "a model of a model" (Norman, 1983, p. 8).

In the case of ITS, the target system is the domain with which it is concerned. "Conceptual models are invented by teachers, designers, scientists, and engineers" (Norman, 1983, p. 7). In a way a conceptual model is a model of a model, i.e., it is based upon the mental model of the person who designed it. Norman (1983), based upon his research work, listed a number of observations on mental models. (1) "Mental models are incomplete." (2) "People's abilities to run their models are severely limited." (3) "Mental models are unstable." (4) "Mental models do not have firm boundaries: similar devices and operations get confused with one another." (5) "Mental models are unscientific." (6) "Mental models are parsimonious: often people do extra physical operations rather than the mental planning that would allow them to avoid those actions." and (7) Mental models are qualitative (Collins, 1985). It is impossible to capture a mental model of a person. We only, at best, can get a conceptualization of that model.

Mental models mean different things for different researchers. For example, Wenger (1987) defined a mental model as "an internal representation of a physical system used by a person or a program to reason about processes involving that system" (pp. 45-46). From Norman's (1983) point of view this definition could correspond to a mental model, a conceptual model or a conceptualization of a mental model. White & Frederiksen (1990) defined a mental model as "a knowledge structure that incorporates both declarative knowledge ... and procedural knowledge ..., and a control structure that determines how the procedural and declarative knowledge are used in solving problems" (p. 100). Notice here again, from Norman's point of view, this definition could refer to a mental model, conceptual model or conceptualization of a mental model. In fact, most AI researchers do not distinguish between mental and conceptual models. In this proposal, while discussing my research work, I will preserve this distinction.

2.4.3 Mental Models and Their Instructional Significance. Despite the fact that research on mental models is becoming popular, only a few efforts, to date, have been geared to explore their application for instructional purposes. This section reviews some of these efforts.

2.4.3.1 Learning as Model Tuning. WHY (Collins et al., 1975) is regarded as a classical system in ITS history. This system deals with causal reasoning in the domain of meteorology. Research on this system has contributed to our understanding of many aspects of ITS development, but here I will only concentrate on its contribution to the research on mental models. In an early version of WHY, knowledge of the domain was represented in a hierarchy of scripts to capture stereotypical sequences of events. These fixed sequences were designed to capture both the temporal and causal relations between events in a way that is easily amenable to inspection (Stevens et al., 1982). But soon designers of the system realized that this representation of knowledge is limited to linear relationships between events. “We believe that representing knowledge about physical processes requires multiple representational view points ... Our script structures provide one of these ... This representational viewpoint is important, but equally important is the functional viewpoint” (Stevens et al., 1982, p. 15). This perspective/viewpoint considers the various elements involved in domain processes, and their functions in the interactions that give rise to various events in the domain. Although both viewpoints describe the same phenomena, their respective emphases are different. Since they highlight different aspects of processes, they will lead to different presentations of the subject and to the perception of different misconceptions (Stevens & Collins, 1980).

This project in its later years became more theoretical (Wenger, 1987). One of the themes that emerged from this research was that people maintain multiple representations of the domain or system that they study or interact with. Learning in this scenario is "largely a process of refining models so that they correspond better with the real world"

(Stevens & Collins, 1980). A possible list of operations used to refine models is adding, replacing, deleting, generalizing and differentiating parts of the model (Stevens & Collins, 1980). Models that people possess are runnable. These models allow us to consider alternative possibilities and to derive predictions about novel situations. Students' underlying misconceptions "derive from simplifications or distortions in their models" (Stevens & Collins, 1980, p. 183).

Recently a very interesting development has taken place in this line of research. This development is concerned with the componential view of mental models (Collins, 1985; Collins & Gentner, 1983). There are two important aspects to this view. First, the various models people possess are hierarchically organized. These models "support one another in a reductionistic fashion where some relations in one model can be explained in terms of another, lower-level model" (Wenger, 1987, p. 47). Second, each level of these hierarchies can be viewed as a combination of component models. A subset of component models forms a view of the subprocesses of the domain under consideration.

Notice that this line of research is very consistent with the second-order theory of tutoring (see section 2.3.2 & 2.3.4). In light of this research, the student can be viewed as possessing a hierarchy of mental models. Each level of this hierarchy contains a number of component models of the domain. The process of tutoring here can be viewed as probing the student models at successive deeper levels. Student misconceptions can here be associated with either missing or faulty component models. The remediation process by the tutor can then involve strategies specific to the types of component models in question. The jigsaw-puzzle metaphor (see section 2.3.4) can still be used here to explain the tutor's task. But here, instead of two dimensions, the tutor is dealing with a three dimensional puzzle! Each piece of the puzzle is more like a component model of the domain. Tutoring in this scenario is indeed more difficult but if it is done properly, we hope, it will be more effective.

Although the notion of learning as model tuning is a useful idea, it puts a heavy burden on the diagnostic process. Also, identifying all of the relevant component models is not an easy task in complex domains. No matter how much effort one puts into collecting component models of a domain, it is always possible that the student begins a tutoring session with a domain model that is never seen by the tutor. In such a situation even a powerful diagnostic method will fall short of fully effective performance.

2.4.3.2 Learning as Model Progression: A Developmental Approach. QUEST (White & Frederiksen, 1990; White & Frederiksen, 1986) is a learning environment (see section 2.1) and its domain of application is electrical circuits. The goal of this system is that the student develops runnable models of electrical circuits so as to be able to predict the status of components and perform a small set of troubleshooting operations. The theme of this research is to investigate successions of mental models that correspond to increasing levels of expertise about the principles of the domain. QUEST views experts as using a set of mental models. The transition from novice to expert is seen as one of model evolution, in which students progress through a series of upwardly compatible models, each adequate to solve a particular subset of the problems. The theory states that the student should be able to progress though a series of models in learning about the domain. The qualitative, causal models used in this system “serve to drive circuit simulations, and to generate causal explanations of circuit behavior ... The central feature of the approach is that, at each stage of learning, the model driving the computer simulation represents the mental model that the student is to acquire” (pp. 99-100). The mental models used in QUEST are in actuality the conceptual models (Norman, 1983) of the domain.

This research also introduced some concepts of the typology of mental models. The three dimensions that White & Frederiksen (1990) proposed were: perspective, order, and degree of elaboration. The model perspective represents a lateral progression in the multi-dimensional space of models. Lateral progression serves to represent alternative means of understanding the domain. "The perspective of a model refers to the nature of the model's reasoning in explaining a circuit's operation" (p. 105). The order of a model reflects the order of the derivatives that are used to describe changes. In other words, order relates to whether the model is based on binary values of states (e.g., on/off, open/close), and properties (e.g., conductive/non-conductive), or based on first-order or second-order derivatives - zero, first and second order models respectively. Elaboration is the third dimension in which models can vary. This dimension controls the degree of constraints to vary the sophistication of model. Expertise in this learning scenario represent the proper fusion of models of various perspectives, orders, and degrees into a coherent and global mental model of the domain.

White & Frederiksen (1990; 1986) place less emphasis on diagnostic and remedial activities. For them it is more important to "refine the progressions of mental models and associated problems" (Wenger, 1987, p. 97). Notice that this is just opposite to the research approach taken by Stevens & Collins (1980). Wenger (1987) has most appropriately stated an important concern regarding this research effort, as follows

> The central question of integration of multiple conceptualizations, mentioned by White & Frederiksen as a key to deep understanding, is not yet addressed seriously - in particular, the issue of the meaning of concepts in electricity in the context of the student's model of the world. <u>The conceptual integration almost inevitably involves an interactive process whereby analogies to existing concepts are tuned and implicit assumptions exposed</u> (p. 97).

2.4.3.3 <u>ITSIE: A Learning Environment Based on Multiple Models</u>. ITSIE (Intelligent Training Systems in Industrial Environments) is a learning environment based on multiple qualitative models (Sime & Leitch, 1992). This system uses ideas from the method of cognitive apprenticeship (Collins et al., 1989) and can be used in two modes: free exploration or apprenticeship. The authors of this system claim that the advantage of the approach used in this system "is that it enables the teaching of multiple models of a physical system, teaching not only the models themselves, but also their strengths and limitations and how each can serve a different purpose during problem-solving" (p. 122). Unlike WHY and QUEST the development of this system seems not to be driven by an explicit theory.

This research has defined five modeling dimensions. Each model in this system is characterized by the position it holds in each of these dimensions. (1) Scope: "it describes the area of the domain covered by the model." (2) Generality: "it is the degree to which it represents generic knowledge." (3) Abstraction: "it relates to the quality space used to represent values in the model." (4) Approximation: "It relates to the accuracy of the model." (5) Granularity: "It relates to the level of detail of the model" (Sime & Leitch, 1992, p. 117).

The models used in this system are conceptual rather than mental models of the domain. From the examples provided (Sime & Leitch, 1992) it seems that the system makes use of a primitive model of the student. This system, like WHY and QUEST, uses multiple models of the domain, but, unlike the research conducted in the context of WHY, it advocates the use of a simple student modeling capacity.

2.4.3.4 <u>Qualitative Modeling</u>. This section describes, very briefly, two seminal research efforts on qualitative modeling. These provide very general and theoretical frameworks that can be used in ITS research for the formation, as well as the use, of causal models of the domain. It is interesting to note that both of these research efforts were motivated by two classical ITS research projects - SOPHIE (Brown et al., 1982) and STEAMER (Stevens & Roberts, 1983).

SOPHIE is regarded as a classical ITS. The domain of this project is the troubleshooting of electric circuits. This project went through three successive phases. SOPHIE-I (Brown et al., 1982) is a reactive learning environment. SOPHIE-II (Brown et al., 1982) is an articulated expert system that provides qualitative explanations of meaningful measurements and decisions. Although SOPHIE-II provides multiple qualitative explanations, it, like SOPHIE-I, is based upon a general-purpose electronic simulator called SPICE, which is qualitative. Experiments carried out using SOPHIE-II revealed that both novices and experts preferred to reason in qualitative and causal terms. SOPHIE-III (Brown at al., 1982) employed human-like reasoning. Further work on this project by de Kleer and Brown (1983) led to a principled framework to represent and use qualitative, causal models of physical systems. This modeling process is used in a computer program ENVISION. Wenger (1987) noted that: "In contrast with most psychological approaches ... (de Kleer & Brown's work) does not attempt to capture the way people in general use mental models; rather, it aims to formalize the informal, qualitative reasoning of experts with a language and calculus for constructing causal models" (p. 74). Recently this work has led to a proposal for a qualitative physics that can be used as an alternative to mathematically-based physics in certain applications (Brown & de Kleer, 1984).

STEAMER (Stevens & Roberts, 1983) attempts to teach a qualitative appreciation of a complex system - the steam propulsion plant of a ship - through an interactive inspectable simulation based on a mathematical model. Multiple levels of graphical abstractions are used to convey different degrees of conceptual fidelity. Like SOPHIE, STEAMER has also inspired research into qualitative reasoning and explanation, leading to the formulation of qualitative process theory (Forbus, 1984). Forbus investigated the way people think about physical processes and this resulted in an attempt to encode causality as perceived by people in general - a form of naive physics. With respect to knowledge representation, SOPHIE and STEAMER led to the formation of two complementary approaches: one based upon physical systems and their components, and the other based upon physical processes.

2.4.3.5 ABEL: Multiple Representation of Medical Knowledge. This section describes a medical expert system in which domain knowledge is represented at multiple levels. Although this system is not used for instructional purposes, its knowledge representation scheme presents an interesting case, which could be utilized in an ITS design.

ABEL is a medical expert system that provides expert consultation in cases of electrolyte and acid-base disturbances (Patel et al., 1984; Patel, 1988). The causal knowledge of ABEL is organized in a multi-level representation. Patel argues that a physician's knowledge is expressed at various levels of detail. These levels are classified as clinical, intermediate and pathophysiological. At the shallowest level the causal knowledge of ABEL is expressed in terms of diseases and their clinically observable manifestations. At the deepest level this knowledge includes detailed biochemical and pathophysiology mechanisms. The intermediate and pathophysiological levels are successive elaborations of information at the clinical level.

The notion of elaboration used in ABEL is very important from a pedagogical standpoint. In fact the elaboration of domain knowledge forms a major theme of my research reported in this proposal.

2.5 Pedagogy: Issues for Intelligent Tutoring Systems

One of the experts in the multiple expert metaphor (see Section 2.2.1) deals with the pedagogical issues (e.g., curriculum, instruction) involved in the working of an ITS. This expert, called the pedagogy expert, is the theme of this section. This expert contains a theory of tutoring. In other words it has a set of specifications of *what* instructional material the system should present and *how* and *when* it should present it. As Wenger (1987) noted, the idea of representing pedagogical knowledge explicitly in an ITS is relatively recent. Researchers have paid more attention to the representation of subject matter than to pedagogical knowledge.

2.5.1 Curriculum and Instruction: The Core Issues. All instructional systems (including ITSs) deal with two major issues: curriculum and instruction. Halff (1988) says that the *problem of curriculum* can be broken into two problems: "formulating a representation of the material, and selecting and sequencing particular concepts from that representation" (p. 81), whereas the *problem of instruction* deals with the actual presentation of the selected concepts to the student.

Halff (1988) defines these terms in the context of ITSs. Instructional designers (Romiszowski, 1981, 1984) define these terms slightly differently. For example, Popham & Baker (1970b) define the *curricular question* as dealing with the ends: "an educator who is involved with curricular questions is exclusively concerned with determining the objectives of the educational system" (p. 82), whereas the *instructional question* deals with the means, i.e., "the procedures for accomplishing those objectives" (p. 82).

The word *curriculum* itself is used by authors rather more freely. For example, according to Wenger (1987) "a curriculum is quite clearly a plan" (p. 397), while for Imbeau et al. (1988) and Romiszowski (1981, 1984) the word curriculum means the subject matter, organized into structures of different shapes (e.g., linear, pyramidal, spiral).

Since the topic of this section concerns both ITS developers and instructional designers and these two communities do not necessarily use the same vocabulary for these concepts (Park et al., 1987), I will explain the differences in terminology as this section proceeds.

The representation of the subject matter (knowledge representation in the ITS jargon) is one of the most important issues involved in curriculum because its structure is supposed to support the selection, sequencing, and presentation issues of pedagogical activity. In the ITS field, the problem of the representation of subject matter belongs to the domain expert and selection, sequencing, and presentation of the subject matter are the responsibilities of the pedagogy expert. This separation of curriculum issues in an ITS, although it supports the modularity hypothesis (Halff, 1988), is, in fact, a mixed blessing (see Section 2.2.1).

2.5.2 Instructional System Design (ISD). The field of instructional system design has a long history as a discipline that is concerned with understanding and improving one aspect of education: the process of instruction (Reigeluth, 1983b). The field of ITS is relatively new and it would not be wise to omit the exploration of ISD as a source of ideas for the design of the pedagogy expert in an ITS. In this section I will briefly discuss some of the major issues of ISD and their relevance to ITS research.

There are many viewpoints of ISD (Gagne & Briggs, 1979; Pirolli & Greeno, 1988; Reigeluth, 1983a; Reigeluth, 1987). In this section I will mainly be describing the viewpoint proposed by Romiszowski (1981, 1984). According to Romiszowski (1981), ISD is a three phase process. In the first phase precise and useful objectives (also called goals or aims) of the instructional system are established. Notice that this phase is concerned with the first problem of curriculum - subject matter representation. The second phase is concerned with planning viable routes to achieve already established objectives. Notice that this phase is concerned with the selection, sequencing, and even presentation problems of curriculum and instruction (see Section 2.5.1). The third phase is concerned with the testing of already planned routes to the established objectives. In other words ISD is concerned with “analysis, synthesis, and evaluation” (Romiszowski, 1981, p. 4) phases. Romiszowski (1981) has further divided each of these phases into four layers to make this design process more modular.

Here I will focus on one aspect of the analysis phase - developing objectives for an instructional system. This phase is analogous to the knowledge analysis (or acquisition) in ITS. The purpose of this section is not to revisit the issue of knowledge representation (see Section 2.4) but to show the interdependence between domain expertise and pedagogy expertise.

It should be noted that all four levels of the analysis phase are mainly concerned with the development of objectives for an instructional system (Romiszowski, 1981). If we equate the process of collecting objectives with knowledge acquisition in ITS, it is not surprising that ITS developers also spend more time on the knowledge acquisition and representation phase than any other aspects of the system and emphasize these problems more (Anderson, 1988).

Words like goals, aims, and purposes are sometimes used synonymously with objectives (or educational objectives). But according to Romiszowski (1981) unlike the other terms, an educational objective, "is a precise statement of intent" (p. 43). He further states that

> An aim may be stated in *input* terms (e.g., to teach history), or *process* terms (e.g., to solve maths problems). An objective is always stated in *output* (or product) terms. It is also stated more precisely. For example, the aim "to solve maths problems" would transform into an objective (or rather a set of objectives) thus: (1) Given maths problems of the following types (specify). (2) The students should solve them. (3) To the following standard of speed and accuracy (specify) (p. 43).

This definition of an educational objective is in the behaviorist tradition. In the ITS field the word goal is more popular (Lesgold, 1988). Also it is not necessarily restricted to output or product terms. The objectives of an instructional system form a hierarchy. This hierarchy has more commonly the following levels: course objectives, unit objectives, lesson objectives, and exercise objectives.

Two more popularly used methods of developing educational objectives are task analysis and topic (or subject matter) analysis (Gagne & Briggs, 1979; Pirolli & Greeno, 1988; Romiszowski, 1981).

According to Romiszowski (1981) "a task is a coherent set of activities (steps, operations, or behavior elements) which leads to a measurable end result. The steps of a task are therefore interrelated" (p. 83). The task analysis procedure breaks down the target task into its components (or steps). This procedure also yields an operational sequence (which may or may not be the best instructional sequence). The steps of the task are then converted into intermediate objectives. Gagne & Briggs (1979) have developed a special kind of task analysis method called the learning-task analysis. This method yields learning hierarchies. A learning hierarchy represents an intellectual skill (Aronson & Briggs, 1983) in a graphical form. In a learning hierarchy the target skill is at the top and below it are all the essential prerequisites (called learning prerequisites).

Essential prerequisites are those subordinate skills that must have been previously learned to enable the learner to reach the target objective. A learning hierarchy provides a guide for planning the sequence in which the objectives should be achieved. Even though the task analysis that results in constructing a learning hierarchy proceeds from the top down, the instruction is sequenced from the bottom up.

Another method of generating objectives for an instructional system is based on subject matter analysis. This method is concerned with the structure of concepts and principles in the subject matter domain rather than with the behaviors that are involved in the performance of a task. According to Romiszowski (1981) a subject matter "is made up of a coherent set of elements (or teaching points or rules) which are interrelated into a sequence or a *structure* (a complex set of interrelationships which are not perhaps sequentially structured)" (p. 85). Usually the subject matter experts perform this task of analyzing the domain. Instead of exploiting the natural relationships of the domain, sometimes the theoretical arguments are used to structure the contents of the domain (e.g., the elaboration theory (Reigeluth & Stein, 1983) suggests the structuring of the subject matter based on the elaboration sequences). As Romiszowski (1981) noted, transforming elements of a subject matter into performance objectives is not always an easy job. One solution to this problem is to use a taxonomy of educational objectives. One such taxonomy, quite popularly used in cognitive domains, was developed by Bloom (1956).

The type of course influences the relationship between the objectives. "This relationship may be a strict sequential dependence, one objective being impossible to achieve until another has been learned or it may be a thematic relationship of a looser character (for example some objectives cohere because they deal with the same general topic)" (Romiszowski, 1981, p. 281). The objectives for an educational activity form a structure that is more popularly called the curriculum structure. Some of the curriculum structures based upon the type of course are linear, spiral, core, pyramidal, project centered and inquiry centered (Romiszowski, 1981).

The question of sequencing the curriculum for an instructional system is handled more explicitly in the design phase. There are many influencing factors that determine the sequence decisions for the course being planned. Some of these are: the type of instructional course, the general philosophical or theoretical viewpoint regarding the nature of learning and instructional processes. For example, Anderson et al.'s (1990) tutoring systems use a model tracing methodology, which helps in the topic selection and sequencing decisions. This methodology is based on the ACT* learning theory.

Romiszowski (1981) listed a set of general purpose methods which are also used to sequence instruction for a course. These are explained briefly as follows: (1) From simple to complex: here, subject matters entities that are simple to learn should be taught first. (2) From known to unknown: "implying that learning should be so planned as always to commence from a concept or procedure that the learner has already mastered and expanding his activities by carefully building on his base" (p. 291). (3) From particular to general: "implying that general principles should be introduced by means of examples first." (p. 291). (4) From concrete to abstract: in one sense it is the same as the rule of particular to general but "also being taken in the sense implied by the viewpoints of Piaget, Bruner and their followers ..., concerning the *learning cycle* of concrete experiences followed by analysis followed by generalization in abstract terms and then back to more concrete experiences" (p. 291).

2.5.3 Elaboration Theory of Instruction: The "Zoom Lens" Metaphor. Besides general heuristics, based on experience, for the design of the instructional system, researchers have also developed formal theories that can be used to develop an instructional system. In this section I will discuss the elaboration theory of instruction (Reigeluth & Stein, 1983). The purpose of this section is two-fold. First, it will give the reader a flavor of the theoretical work in the field of instructional design. Second, and more important, the elaboration theory has an interesting parallel with the model of tutoring I am proposing.

A theory, in the instructional design field, does not necessarily covers all of the aspects of the instructional design process (Reigeluth, 1983b; Landa, 1983). Reigeluth & Merrill (1979) have proposed a framework for instructional design (see Pirolli & Greeno (1988) for another example of such a framework). Based on this framework, the elaboration theory is a macro level theory of instruction. At the macro level, a theory deals with representation, sequencing, and selection of a number of topics (objectives, goals, or issues). Using Halff's (1988) terminology (see section 2.5.1) the elaboration theory deals with the curriculum rather than the instructional issues. An example of a theory which deals at the micro level (i.e., concerned with the instructional issues) is Merrill's (1983) component display theory. A comprehensive coverage of the elaboration theory can be found in (Reigeluth & Stein, 1983; Reigeluth et al., 1980). Here I will discuss, very briefly, a few of its key themes.

The elaboration theory is comprised of three models of instruction and a system for prescribing these models in accordance with the goals and purposes of a course. These three models are based on the major type of knowledge used in the course. These models are: the conceptually organized model, the procedurally organized model, and the theoretically organized model.

According to Reigeluth & Stein (1983) "studying a subject matter *through* the elaboration model is similar in many respects to studying a picture through a zoom lens on a movie camera" (p. 340). A person starts with a wide-angle view that allows him/her to see the complete picture and its parts but without details. Then the person using the lens zooms in to see more details of the parts of the picture. In this metaphor, Reigeluth & Stein (1983) assumes that "instead of being continuous, the zoom operates in steps or discrete levels" (p. 340). After studying a part of the picture the person can zoom out again to the wide angle view to see other parts of the picture and analyze the context of the inspected parts with the whole picture. The person can continue the zooming in and zooming out operation at several levels and parts of the picture to analyze the picture at sufficient depth.

> In a similar way, the Elaboration theory of instruction starts the instruction with a special kind of overview of the simplest and most fundamental ideas within the subject matter; it adds a certain amount of complexity or detail to one part or aspect of the overview; it reviews the overview and shows the relationships between the most recent ideas and the ideas presented earlier; and it continues this pattern of elaboration followed by summary and synthesis until the desired level of complexity has been reached on all desired parts or aspects of the subject matter (Reigeluth & Stein, 1983, p. 341).

The key of the elaboration theory's simple-to-complex sequence is that it helps to ensure that the student is always aware of the context and importance of the different ideas that are being taught. This sequence, as Reigeluth & Stein (1983) states, is an extension of Ausubel's (1968) subsumptive sequencing, Bruner's (1966) spiral curriculum, and Norman's (1973) web-learning. Other characteristics of the elaboration theory's simple-to-complex sequence, as Reigeluth & Stein (1983) claims, are that it helps to build stable cognitive structures and "provides meaningful application-level learning from the very first lesson" (p. 337).

2.5.4 <u>Implications of ISD for ITS Research</u>. Relative to ISD, ITS research is a new field of study. Both of these fields have many similar interests. As a result, one can expect that research in one of these fields will also contribute to the other. Several researchers have already started to attempt to combine these fields into a unified framework (e.g., see Pirolli & Greeno, 1988). In this section I will be mainly concerned with a one sided view, the implications of ISD for ITS.

Halff (1988) noted that most instructional research is tangentially relevant to ITS, "either because it addresses other forms of instruction or is simply not sufficiently oriented to design to be of direct help" (p. 97). Galdes (1990) has also shown similar concerns. But ISD, which is a branch of instructional research, is a mixed blessing for ITS (Halff, 1988).

There are two major advantages of ISD research for ITS. First, ISD offers a framework for the comprehensive treatment of instructional design. ITS lags badly in this area, maybe because it is still in an exploratory phase of research (Galdes, 1990). As Halff (1988) noted “the chances of finding an intelligent tutor that meets the needs of a randomly chosen application are quite small indeed. By contrast, ISD offers a top-down approach that covers a large area of the instructional waterfront” (p. 98). Second, ISD proposes a comprehensive decomposition of the design process. The ITS design process overlaps greatly with ISD but it is not sufficiently comprehensive. In other words, many ITS skip the system viewpoint (see section 1.5.2). Maybe this is one of the reasons that the majority of ITSs are still not a part of real educational settings.

There are also two major reasons why ISD research is not of much use for ITS (at least directly). First, although ISD provides prescriptions for a wide variety of instructional settings, it does not much emphasize tutorial situations (Halff, 1988; Galdes, 1990). “Tutoring, after all, is an expensive and uncommon instructional method, and for this reason alone may have failed to capture the attention of the ISD community” (Halff, 1988, p. 99). It is probably evident from Section 2.5.2 that almost all ISD theories and models prescribe methods for instructional situations (e.g., lecturing) that are less dynamic than tutoring. In these instructional situations both curriculum and instruction can be developed prior to delivery. “Tutoring systems afford no such luxury because a tutor, human or machine, is bound to tailor the selection, sequencing, and methods of delivering instruction to meet the ever-changing needs of individual students” (Halff, 1988, p. 80). Second, most ISD theories and models are meant to be used by the instructional designers. These lack the specificity necessary for formalization and programming on a computer (Breuker, 1988; Halff, 1988; Galdes, 1990).

All in all, it is appropriate to encourage efforts to shorten the gap between these two fields so that each one can benefit from the other. The following sections will show many direct or indirect influences of ISD on the development of ITSs.

2.5.5 <u>View of Expertise: A Curriculum or A Model?</u> In the previous section we have seen that ISD concentrates on the issues of objectives (goals or aims). The two most popularly used methods of collecting and structuring objectives for a course are task analysis and subject-matter (or topic) analysis. These methods can also be called knowledge structuring methods (Park et al., 1987). The task analysis method is more natural for courses that concentrate on tasks (cognitive or psychomotor). Subject matter analysis is more natural for courses that are a primary source of instruction for a subject matter area. Most of the courses in current educational systems fall in this category. The distinction I have made here is not strictly followed by instructional designers. For an example of the task analysis approach to topic analysis see Romiszowski (1981).

In the ITS field knowledge structuring techniques are more popularly called knowledge representation methods (Park et al., 1987). Most of the research has concentrated on the problem-solving domains (Wenger, 1987; Galdes, 1990). Here more emphasis is being placed on developing cognitive models of expertise for problem-solving tasks. Reigeluth & Stein (1983) called this an information-processing approach to task analysis. These models are different in two respects from the knowledge structures resulting from the task analysis methods of ISD. First, these models are runnable to yield a simulation of the expert behavior. Second, these models emphasize the order in which tasks must be performed as opposed to the order in which they must be learned (Reigeluth & Stein, 1983).

Only a few ITS built to date have emphasized basing their architectures purely around subject-matter organization. These ITS use methods of developing knowledge structures that are analogous to the methods of subject-matter analysis that are popular in ISD. According to Wenger (1987) these ITSs emphasize a curriculum view of domain expertise as opposed to a model view of expertise. For the sake of clarity I will call these two opposing types of ITS curriculum-based ITS, e.g., BIP (Barr et al., 1976), WUSOR

(Goldstein, 1982), MHO (Lesgold et al., 1989) and model-based ITS, e.g., LISP Tutor (Anderson et al., 1990), QUEST (White & Frederiksen, 1990).

I have already described some model-based ITS in Section 2.4. In the next few sections I will describe some of the seminal ITS that emphasize the curriculum view of ITS. This line of research is currently proposing an integral view of these two opposing perspectives of ITS (Wenger, 1987).

2.5.6 BIP: An Early Curriculum-Based ITS. BIP (Basic Instructional Program) is one of the pioneering ITS that introduced the notion of curriculum-based ITS. The domain of this ITS is introductory programming in BASIC (Barr et al., 1976). BIP-I represents its knowledge in a network, called the curriculum information network. This network has three layers called techniques, skills, and tasks. Techniques are at the top of network and represent the issues of expertise in the BASIC programming language (e.g., output single value). These techniques are composed of lower-level knowledge units called skills (e.g., print numeric variable). The last layer contains tasks that exercise the skills of the network (e.g., Task Horse - write a program that prints the string "HORSE"). Techniques in the first layer are ordered using prerequisite relations.

BIP-I uses a problem-selection method based on a fairly straight forward optimization process. This method first constructs a set of skills to be exercised. Then another set is constructed to contain the skills that are sufficiently mastered. Finally, a task has been searched that exercises the greatest number of skills in the required set. Here the emphasis has been on finding a task that is not beyond the student's reach.

BIP-II (Wescourt et al., 1977) augments the curriculum information network in an important way. Here the skill level is greatly expanded and a number of domain independent relationships are introduced between the nodes of this level. The generic relationships representing the pedagogy information used are: analogical relations, class-object relations, functional dependencies and relative difficulty (Wescourt et al., 1977).

The nodes of the skill layer in BIP-II have also been changed to the primitive elements of the domain (e.g., control structure, if-then constructs).

One of the major criticism of BIP (and for a curriculum-based ITS in general) is that, unlike a model-based ITS, it does not have an operational model of expertise. As a result BIP's diagnostic capabilities and its usefulness as an active programming tutor are limited. Once a program has been completed, "BIP simply tests it on a set of input/output pairs without any analysis of the algorithm" (Wenger, 1987, p. 111).

2.5.7 WUSOR: Superimposing an Instructional Curriculum Network on an Operational Model of Expertise. WUSOR (Goldstein, 1982) provides a coaching environment for the computer game WUMPUS. This project concentrates more on the knowledge representation issues, with an attempt to include domain independent relationships in its domain knowledge in such a way that they help in pedagogical decision making. WUSOR is the first attempt to combine the curriculum-based and model-based features of ITS in a single architecture. Three versions of this system was developed, namely WUSOR-I, WUSOR-II, and WUSOR-III (Goldstein, 1982).

WUSOR-I is an expert-based coach. The domain knowledge in the expert is organized as production rules. This version does not have any tutorial strategies. As a result it always interrupts student when he/she is not making optimal move according to the expert's classification.

WUSOR-II used a detailed student model to improve the pedagogical effectiveness of the system. Research on this version formalized the "overlay theory of student modeling, which has become a standard paradigm in ITS" (Wenger, 1987, p. 137).

WUSOR-III introduced the notion of genetic graph (Goldstein, 1982). The nodes of this graph represent the elementary sub-skills (i.e., individual rules). These nodes are connected by evolutionary relations such as generalization/specialization, analogy, deviation/correction, simplification/refinement. "The word genetic refers to Piaget's notion of genetic epistemology because this representation attempts to capture the evolutionary nature of knowledge" (Wenger, 1987, p. 141).

The genetic graph provides two major features which help in the system's pedagogical decision making process. First, it facilitates the process of topic collection based upon the current context of the system. Second, it supports generation of multiple explanations based upon the evolutionary relations between its nodes. Goldstein (1982) argues that an ITS that represents expertise only using domain dependent relationships is not as powerful as a system employing a generic graph like representation; they fail to take advantage of the fact that the new knowledge of the learner evolves from old knowledge by processes like analogy, generalization, debugging and refinement. Goldstein (1982) noted that a genetic graph despite its many advantages does not solve the tutoring problem. Furthermore, creating a static graph, like a genetic graph is not easy for a complex domain. Although WUSOR-III is the first attempt to bring the curriculum-based and model-based themes together, its solution is not elegant in the sense that a more fundamental, and theoretically based formalism is needed.

2.5.8 Towards a Theory of Curriculum: An Attempt to Unite Curriculum-Based and Model-Based Themes of ITS. We now know that domain expertise alone is not enough to effectively perform pedagogical tasks (Lesgold et al., 1989; Anderson, 1988; Breuker 1988). Tutoring requires both domain and pedagogy expertise. Pedagogy expertise uses domain knowledge. Pure expert models of the domain usually lack the knowledge that guides the pedagogy expertise. Lesgold et al. (1989) define the curriculum knowledge as "the specification of the goal structure that guides the teaching of a body of expertise" (p. 342). Educational researchers, including ITS developers, often treat the domain expertise and curriculum (curriculum goals) as the same. Lesgold et al. (1989) further states that

> They (instructional system designers) assume that expertise can be split apart easily "at its joints" (to use Plato's phrase) and that curriculum is a natural hierarchy of goals and subgoals to teach the natural units of expertise. There

appear, however, to be many different plans for splitting apart expertise especially when it involves complex performances (p. 342).

As we have seen, WUSOR was the first system that made an attempt to superimpose an information curriculum network on an operational model of expertise. But again it equated each unit of domain expertise with a pedagogy unit in the information curriculum network.

Lesgold (1988) proposed a framework for knowledge representation in an instructional system. The characteristic of this framework is that it treats metacognitive skills, curriculum, and domain knowledge separately. The topmost layer of this framework contains metacognitive issues such as reading ability, verbal facility, speed of learning. Lesgold (1988) argues that, like good tutors, an instructional system should tailor the curriculum knowledge to the aptitudes of the student. Meta issues are directly related in this framework with the curriculum goals that constitute the second layer. He called this layer the curriculum layer or goal lattice layer. The contents of this layer comprised of hierarchy of goals. This structure is similar to the Gagne & Briggs (1979) notion of learning hierarchy. In an important way this layer differs from learning hierarchies in that it considers the notion of multiple views of the curriculum. According to Lesgold (1988), in some domains, it is possible to view the curriculum through different perspectives. For example, a basic resistor network course can be viewed through four different perspectives: circuit types, electric laws, electric concepts, and problem types. Based upon the student’s aptitude one can select an appropriate viewpoint for instruction. Notice the similarity of concept of the curriculum viewpoint with the notion of multiple mental models. The third layer is the knowledge layer. This layer contains the domain knowledge that the system intended to teach. “One way to think about that knowledge is that it is a model of expert capability in the domain. Such knowledge includes both procedures and concepts (i.e., both procedural and declarative knowledge)" (Lesgold, 1988, p. 121). Goals and subgoals in the curriculum layer point to the issues or chunks of knowledge in the knowledge layer.

This framework has contributed in many ways towards formalizing a conceptual model for an instructional system. First, it made explicit the usually hidden type of knowledge in ITS research - the knowledge of the curriculum (or goals). Second, curriculum and domain knowledge are elegantly organized into separate layers. Each type of knowledge involves different issues but these now are local to their respective layers. This framework emphasizes modularization - an important issue in ITS research. Third, it emphasizes the utility of a mostly forgotten aspect of human tutors - their metacognitive skills. Fourth, since this framework separates curriculum from domain knowledge, it naturally supports the integration of the curriculum-based and model-based themes of ITS.

Despite all of these advantages this framework has the following shortcomings. First, despite its effort to combine the two themes of ITS, it still emphasizes the curriculum-based aspect of ITS design. Second, the organization of the goal lattice in the curriculum layer supports systems that act as a primary source of instruction for their users. Third, the goal lattice supports declarative orientation of the subject matter. As a result the system using this framework supports learning of domain concepts and principles more than complex models of the domain. Fourth, Lesgold suggested only the overlay approach to student modeling with this framework. Research (e.g., see VanLehn, 1988) has shown that this is only one approach to student modeling.

Admittedly, Lesgold's framework has advanced the field of ITS considerably, but the true dream of weaving curriculum-based and model-based themes is not yet fully achieved.

2.5.9 <u>Selection and Sequencing Decisions: General Principles</u>. In the previous sections I have concentrated on the issues of knowledge/curriculum representation. These issues alone do not solve the problem of pedagogy (Goldstein, 1982). Prescriptions for the selection and sequencing of the material for tutoring are usually based upon some theory of tutoring (Collins & Stevens, 1991; Galdes, 1990). Each theory's prescriptions are based on its underlying philosophical standpoint. For example Collins & Stevens's (1991) theory is based on the discovery principle of learning. The two theories do not necessarily agree on the same philosophical standpoints, but there are some general principles that are usually accepted by the majority of the camps in the field of education. Halff (1988) described four such principles that help selection and sequencing decisions for active machine tutors. (1) Relatedness: concepts (or topics) that are closely related to current knowledge of the student are given priority. (2) Manageability: "every exercise should be solvable and every example should be comprehensible to students who have completed the previous part of the curriculum" (p. 85). (3) Structural transparency: the sequence of exercise and examples should be organized in such a way that they make explicit the structure of the knowledge being taught. This hopefully will help the student to induce the target knowledge. (4) Individualization: "exercises and examples should be chosen to fit the pattern of skill and weaknesses that characterizes the student at the time the exercise or example is chosen" (p. 87).

A number of (topic) sequencing principles have been used in different ITS (Wenger, 1987). For example WUSOR-II (Goldstein, 1982) and Anderson's tutors (Anderson et al., 1990) use the *simple to complex* principle, i.e., simple concepts (or topics) are given priority over the complex ones. BIP (Barr et al., 1976) and WUSOR-III (Goldstein, 1982) use the *prerequisite first* principle, i.e., the topic selection mechanism in these systems, using prerequisite and other genetic relations, give priority to the topics that are prerequisites for the current lesson. SCHOLAR (Carbonell, 1970b) uses the *important first* principle, i.e., here important topics are given the priority. WHY (Collins et al., 1975) uses the logical structure of the domain to drive its sequencing mechanism.

2.5.10 <u>Pedagogy as Problem Solving: An ITS Needs a Planning Mechanism</u>. Empirical studies of human tutors revealed several facts. For example, tutors are guided by an overall goal structure (Collins & Stevens, 1982) that has multiple levels (Woolf & McDonald, 1985). Some of these goals are related to global issues such as topic selection methods (Collins et al., 1975). Others address local issues such as responding to the student's last action (Collins et al., 1975; Woolf & McDonald, 1985). Each goal can be achieved by one or more tutoring strategies (Woolf & McDonald, 1985). Expert tutors possess a method of selecting among the possible strategies to accomplish a given goal (Leinhardt & Greeno, 1986). Each strategy determines a set of tactics that constitute the tutor's actions in a tutoring situation (Ohlsson, 1987).

The above facts form the basis of a view of tutoring. Ohlsson (1987) called this the problem solving view of tutoring. The major ingredients of this view, as is clear from the discussion above, are goals, strategies and tactics. Ohlsson (1987) claims that "the availability of a variety of local tactics, and of global strategies that organize the use of these tactics, is the ultimate determinant of a tutoring system's adaptability" (Wenger, 1987, p. 402). Notice that the type of problem solving Ohlsson is talking about is planning, also called instructional planning (Macmillan et al., 1988). He also conjectures that "a tutor needs to be able to generate a teaching plan on the basis of its representation of the student, its knowledge of the subject matter, and its current tutorial goal; furthermore, it should be able to revise its plan if it discovers that the plan does not fit the student" (Ohlsson, 1987, p. 232). He called this conjecture "The principle of teaching plan." According to this principle, the plan generation process uses strategies to generate plans for the goals of the tutoring system. The terminal ingredient of these plans are tactics that represent the tutor's actions (e.g., ask question, give summary). Strategies, in this view of tutoring, determine the methods for the classical problems of pedagogy (i.e., selection, sequencing, and presentation of the subject matter). One of the biggest advantage of the problem solving (more specifically planning) view of tutoring is that it provides a “design hypothesis” (Ohlsson, 1987) for ITSs. The following sections describe a number of ITSs that based their design on this hypothesis.

2.5.11 Pedagogical Styles: Plan-Based, Opportunistic or Embedded Contexts? Before I discuss various planning mechanisms used by ITS researchers, it is important to clarify the frequently talked about issue of pedagogy styles (Peachey & McCalla, 1986; Wenger, 1987; Derry et al., 1988; Murray, 1988b).

Teaching is goal oriented. Its attainment of these goals is based upon the pedagogical style used. There is a continuum of pedagogical styles available in the field of instruction. At one end of this continuum lies the plan-based approach that achieves these goals via an instructional planning activity. At the other end of this continuum lies the opportunistic method, which relies on the recognition of opportunities during tutorial interaction. At the center of this continuum lies a method that combines plan-based and opportunistic views. Based on empirical research, it is now recognized that this intermediate style is best in attaining teaching goals (Wenger, 1987; Derry et al., 1988; Woo et al., 1991).

In a plan-based context, “the tutor manipulates the sequences of experiences through which the student is expected to acquire the target expertise” (Wenger, 1987, p. 399). Here pedagogical goals predominate, whereas the student’s behavior is of less importance. Most of the instructional systems, designed using the ISD methodology, fall into this category of pedagogical style.

On the contrary, in the opportunistic context, the tutor “takes advantage of teaching opportunities that arise in the context of some activity or dialogue in which the student is engaged” (Wenger, 1987, p. 398). Most of the ITS build to date use this pedagogy style (Wenger, 1987; Peachey & McCalla, 1986; Derry et al., 1988). In these systems teaching opportunities are detected via diagnostic information and planning is locally focused on these opportunities. As Wenger (1987) noted, this pedagogy style is well suited to systems that emphasize problem-based guidance or coaching methods, "especially if the tutored activities complement other kinds of teaching such as formal instruction" (p. 399).

There are two possible ways of combining these two extreme pedagogy styles. The first one, which is not very popular, combines a global opportunistic strategy with local plan-based control. The second possibility, which is also more popular, combines a global plan-based style with local opportunistic control. One reason for the recent popularity of these mixed styles in ITS research is that this mix has been observed in empirical studies of human tutors (Collins & Stevens, 1982; Woolf & McDonald, 1985; Wenger, 1987; Galdes, 1990).

2.5.12 <u>Levels of Tutorial Planning Decisions</u>. It is common for instructional designers to divide the curriculum (composed of goals or objectives of the course under development) into a hierarchy. As mentioned in Section 2.5.2, common levels of this hierarchy are the course, unit, lesson, and exercise. A human tutor using these levels needs to perform decision making at all of these levels.

Murray (1988b) distinguishes three levels of instructional planning for machine tutors. (1) Curriculum planning: machine tutors at this level perform decision making for an extended sequence of lessons. (2) Lesson planning: at this level a tutor performs decision making for a single lesson. This decision making includes determining the subject matter to present and its order of presentation. (3) Discourse planning: it deals with "planning communicative actions between the tutor and the student within a lesson" (Murray, 1988b, p. 3).

Notice that this curriculum level planning is analogous to course and unit level decision making by instructional designers. Lesson level planning is more or less like lesson level decision making. Finally discourse level planning is more like the exercise level decision making. At the curriculum and lesson planning levels the machine tutor needs to decide about the selection and sequencing problems of instruction (see section 2.5.1), whereas at the discourse level, the machine tutor deals with the presentation of instructional material. Murray (1988b) noted that "these levels cannot in practice be so cleanly separated and frequently discourse planning and lesson planning are intertwined, as are lesson planning and curriculum planning" (p. 3).

2.5.13 <u>Discourse Management Network: A Generic Architecture for Discourse Planning</u>. MENO-TUTOR was developed by Woolf & McDonald (1985) as a coherent framework for representing and organizing elements of a discourse strategy. It attempts to capture the discourse strategies observed in human tutors who strive to be sensitive to their listeners (Halff, 1988). MENO-TUTOR is based on a "discourse management network" (DMN) - a kind of augmented transition network. The nodes or states of this network correspond to tutorial actions. These states of the DMN are hierarchically organized into three layers - pedagogic, strategic, and tactical (Woolf, 1984; Woolf & McDonald, 1985; Woolf, 1988b). In all, the DMN has 40 tutoring states. There are two types of arcs used in the DMN. The first type defines the sequences of states normally traversed by the tutor. From a pedagogical viewpoint, these transition corresponds to default tutorial decisions. The second type of arcs in the DMN represent meta rules that can move the focus to any state in the network when their conditions are satisfied. The DMN allows MENO-TUTOR to support both plan-based and opportunistic tutoring. Plan-based tutoring, in a more or less compiled form, is achieved by default transition between the nodes of the DMN, whereas opportunistic tutoring corresponds to the transition between tutoring states based on meta rules firing.

According to Murray (1988b), the DMN is well-suited for supporting opportunistic tutoring strategies, but it must be coupled with other control mechanisms to support lesson and curriculum planning. Although it is true that the plan-based capabilities of DMN are limited, it offers a generic control mechanism, which along with a layered curriculum representation can provide a powerful instructional planning mechanism at all levels (curriculum, lesson, and discourse).

2.5.14 <u>Blackboard Architecture: A Flexible Control Mechanism for ITS</u>. The blackboard architecture provides the second generic planning mechanism for ITS. There are three main features of the blackboard architecture: (1) Hierarchically structured global database or blackboard: this is the place where solutions of the problem being solved are stored. (2) Independent knowledge sources: these are rules that get activated when changes are posted on the black board. These rules "contribute to an evolving solution by adding or modifying solution elements on the blackboard. Knowledge sources communicates only by adding or changing the contents of the blackboard" (Murray, 1988b, p. 10). (3) Agenda control: this is a list of possible actions that can be performed on the elements of black board. "A scheduler selects the next action to execute from the agenda" (Murray, 1988b, p. 10).

Briefly, the blackboard control mechanism works as follows: knowledge sources are activated by the changes in the contents of the blackboard. This causes new actions to be added to the agenda. The scheduler selects the next action to be performed. This next action may cause change in the elements of black board and then this cycle repeats.

BB-IP (Murray, 1988a) and IDE-INTERPRETER (Russell, 1988) use a blackboard architecture to implement their instructional planners (Murray, 1988b). Murray (1988b) argues that, unlike the DMN, the blackboard architecture supports all three levels of instructional planning (curriculum, lesson, and discourse). Furthermore, the blackboard architecture more easily supports multiple tutorial strategies and the development of customized, globally coherent curriculum and lesson plans. An additional advantage of this architecture is that it facilitates the separation of knowledge about planning from both domain knowledge and tutorial strategies (Murray, 1988b).

2.6 <u>KADS: A Generic Knowledge Based System Development Methodology</u>

Intelligent tutoring systems are knowledge-based systems (Clancey, 1987b); hence they can be developed by a generic knowledge-based system (KBS) development methodology. KADS (Knowledge Analysis and Design Structure) is an important step forward in systematic KBS development methodology (Breuker, 1990; Wielinga & Breuker, 1990; Wielinga et al., 1987). This methodology takes the view that KBS development is essentially a modeling activity that yields a series of models at different levels of abstraction. This modeling process starts at the linguistic level, where the raw data of KBS development (e.g., knowledge from text books, interviews, think aloud protocols) is identified. As a model of this knowledge is abstracted, models at the epistemological level of analysis are constructed. These models are sufficiently removed from the real world to be free from any issue of implementation and act as an intermediate representation between the linguistic and the implementation levels (Breuker, 1990). Fundamental to the KADS methodology are the epistemological structures of the model of expertise (Wielinga & Breuker, 1990). Within the KADS methodology, expertise is described in levels of different types of knowledge (Wielinga et al., 1987). According to this methodology the knowledge of an expert is modeled using four levels. The first level contains the static knowledge of the domain. The knowledge at this level does not constrain the potential inferences that can be made. The second level is the inference level and describes what inferences can be made on the basis of the knowledge at the static level. The third level is the task level. It determines how and when certain inferences will be made. The final level is the strategic level and enables the system to acquire the behavioral flexibility shown by a human expert.

I have used some aspects of this methodology in the design and development of the domain and pedagogy experts of CIRCSIM-Tutor (v.3).

CHAPTER III

BACKGROUND OF THE CIRCSIM-Tutor PROJECT: A HISTORICAL TRACE

3.1 <u>Computer-Based Medical Systems for Teaching the Baroreceptor Reflex</u>

CIRCSIM-Tutor is an ITS that assists first year medical students in mastering the behavior of the baroreceptor (BR) reflex, the part of the cardiovascular (CV) system that is responsible for maintaining a more or less constant blood pressure. In this chapter we will trace the history of our system. Briefly this system has roots in two Computer-Aided Instruction (CAI) systems, namely HEARTSIM (Rovick & Brenner, 1983) and CIRCSIM (Rovick & Michael, 1986), developed at Rush Medical College. A prototype and two versions of CIRCSIM-Tutor have already been developed and work is continuing to develop the third version. The research reported in this thesis is a part of this effort. In this chapter we will trace the influence of these early systems/versions on the most recent version of CIRCSIM-Tutor. Figure 3.1 shows a chronology of events for the research reported in this chapter. In the following sections we will first describe these versions briefly. Common concepts and characteristics shared by these versions are explained later in this chapter.

3.2 <u>HEARTSIM: A Quantitative Model of CV System with Didactic Feedback</u>

HEARTSIM was the first of a series of computer-based medical systems for teaching the baroreceptor reflex developed at Rush Medical College. During the development of HEARTSIM a number of novel concepts was pioneered that are still in use in the most recent computer-based tutoring systems developed at Rush/IIT.

In HEARTSIM an instructional component was superimposed on a well known quantitative model of cardiovascular system - MacMan (Dickenson et al., 1973). MacMan here was translated and reformatted to run on PLATO (Rovick & Brenner, 1983). HEARTSIM is a CAI system. Its instructional component is designed to tailor its feedback according to the student’s responses but still its responses are stereotypical compared to what a human tutor can provide in a specific case. MacMan, in this system, simply calculates the responses for CV problems. It is not involved in deciding on the feedback for the student. The student modeling capability of this system is based on the designer's knowledge of the subject domain and extensive teaching experience (Rovick & Brenner, 1983; Rovick & Michael, 1992). HEARTSIM was designed to be user friendly. It provides feedback in textual, numerical, and graphical form. This system was also designed to be as a "free-standing unit" (Rovick & Brenner, 1983, p. 236) so that students could use it independently or in a group format. HEARTSIM was used quite successfully as a part of a physiology course at Rush Medical College for several years. A detailed discussion of this system can be found in (Rovick & Brenner, 1983). An expanded discussion of various concepts pioneered by this system is provided in the following sections.

3.3 CIRCSIM: A HEARTSIM on PC With an Expanded Didactic Capability

CIRCSIM is the most extensively used and tested system for teaching the baroreceptor reflex ever developed at Rush Medical College. CIRCSIM in a way is a PC version of HEARTSIM. One of the major reasons for its development was to widen the audience that can access it (Rovick & Michael, 1986). Instructional and student modeling capabilities in CIRCSIM have been relatively extended. Here the system explicitly attempts to convey an algorithm to the student to solve CV problems. Also instead of basing feedback on single errors, CIRCSIM groups errors to model student. CIRCSIM's teaching effectiveness has been explicitly evaluated and it has been found more effective than traditional methods of instruction (e.g., reading text books). A detailed discussion of this evaluation can be found in (Rovick & Michael, 1992). CIRCSIM is still in use at Rush Medical College as part of a physiology course for first year medical students.

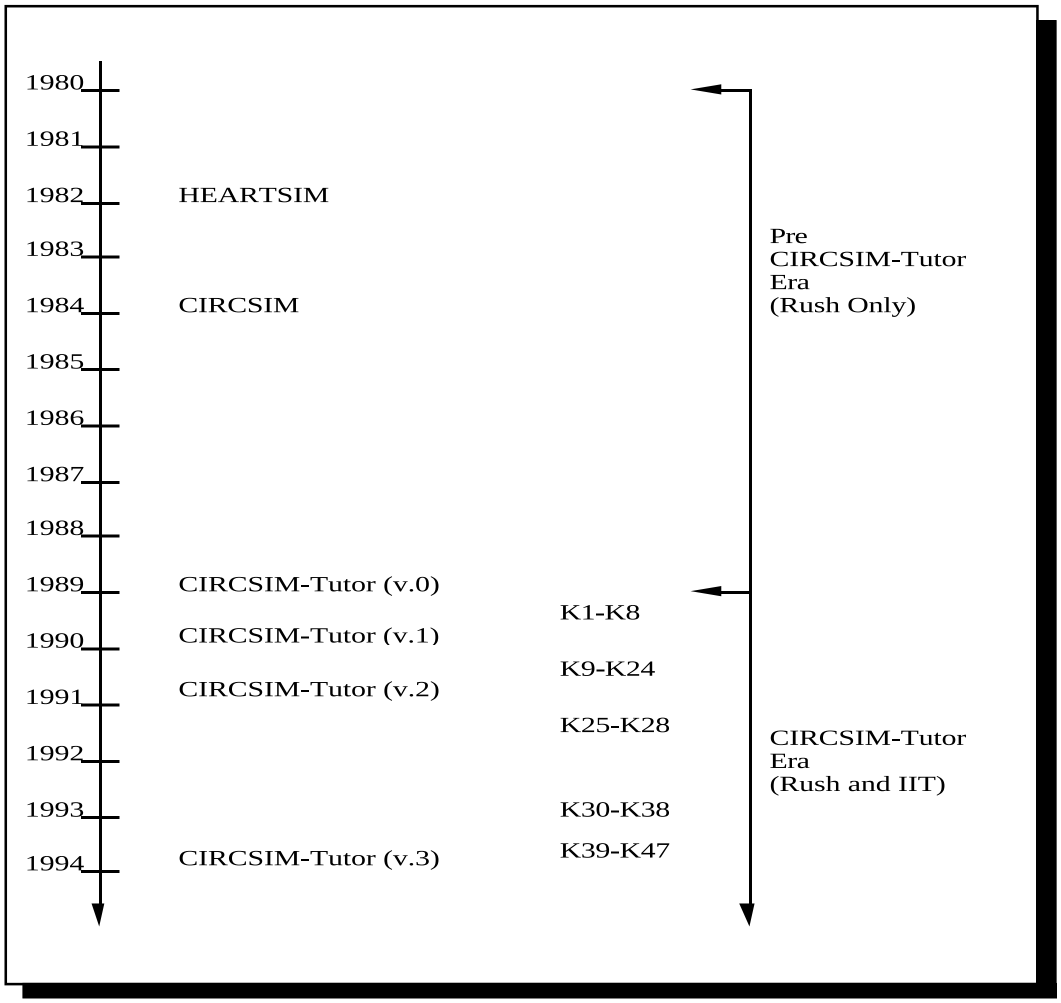


Figure 3.1 Chronology of Events of the Research in Computer-Based Medical Teaching Systems Developed at Rush/IIT

3.4 <u>CIRCSIM-Tutor (v.0): Beginning of an ITS Era</u>

CIRCSIM-Tutor (v.0) is the prototype system (Kim et al., 1989). This ITS was the first result of a joint venture between IIT and Rush Medical College to develop ITS systems. CIRCSIM-Tutor (v.0) used Prolog on a DOS machine. This system inherited many concepts from CIRCSIM but unlike CIRCSIM it makes explicit the domain model, instructional and student modeling issues that were implicit in the CIRCSIM design. CIRCSIM-Tutor (v.0) did not have any natural language capability, also it did not use MacMan, instead a very simple qualitative and causal model of CV system was used.

This system developed an architecture that was very limited in its capability but it provided a basis for further research and development.

3.5 CIRCSIM-Tutor (v.1): Towards an Effective Machine Tutor

The work on this version of CIRCSIM-Tutor started after the development of the prototype version. This version did not came up as an integrated system instead only a few modules of it were developed using the Inter Lisp on Xerox lisp machine. These modules were mostly concerned with processing the natural language aspects of CIRCSIM-Tutor. In fact it is this version that initiated research on natural language interface for CIRCSIM-Tutor. References (Lee et al., 1990; Lee et al., 1991; Lee & Evens, 1992; Zhang et al., 1990) describe the research conducted during the development of this version of CIRCSIM-Tutor.

3.6 CIRCSIM-Tutor (v.2): First Fruit of this Research Movement

CIRCSIM-Tutor (v.2) continued the march started by the CIRCSIM-Tutor (v.0) towards an effective tutor. This is a Macintosh based Lisp system. It inherited most of its concepts from CIRCSIM-Tutor (v.0). The natural language capability here was relatively extended compared to what was available in CIRCSIM-Tutor (v.1). Like CIRCSIM-Tutor (v.0) it also uses a qualitative and causal domain model. It expanded the conceptual model of CIRCSIM-Tutor (v.0) in many respects. A detailed account of these issues is provided in the following sections. Unlike earlier versions, this system has been experimentally used by a limited number of first year medical students. So far no systematic evaluation of CIRCSIM-Tutor (v.2) has been performed but the students who used this system responded favorably to it.

3.7 Characteristics, Capabilities and Research Issues of Computer-Based Medical Systems for Teaching the BR Reflex Developed Before and During the CIRCSIM-Tutor Era

In this section we will describe in detail the characteristics, capabilities and research issues of this sequence of CAI/ITS systems. We will discuss each issue independently and compare the way it has been handled in CAI/ITS systems described above. See Figure 3.2 for some interesting facts about these systems. To increase readability of this chapter we will use the following convention. By CAI systems we will mean all the pre-CIRCSIM-Tutor systems (i.e., HEARTSIM and CIRCSIM). By ITS systems we will mean CIRCSIM-Tutor (v.0), CIRCSIM-Tutor (v.1), and CIRCSIM-Tutor (v.2) and by all systems we will mean all CAI and ITS systems built at Rush/IIT.

Computer-Based Medical Systems For BR Reflex	Type of System	Type of Domain Model	Number of CV parameters used in the PT	Number of CV phases Used in the PT	Number of CV procedu-res	Does the System Offer a Guided Procedure?	No. of CV parame-ters Used in the CM	Type of Student Modeling Approach Used	Didactic Feedback Starts As Soon As:	Was This System Used in Real Ed. Setting?	Hardware Platform Used	Program-ming Language Used
HEARTSIM	CAI	Quantita-tive (used only to display graphically the descriptions of the system responses)	8 (HR, SV, CO, CC, Ra, Pa, Patr, Pc)	3 (DR, RR, SS)	6	Yes	No CM is Used	Implicit Generic Student Model	All Three Columns of PT are Filled By the Student	Yes	PLATO	Tutor
CIRCSIM	CAI	Quantita-tive (used only to display graphically the descriptions of the system responses)	7 (CC, RAP, SV, HR, CO, RA, MAP)	3 (DR, RR, SS)	8	Yes	No CM is Used	Implicit Generic Student Model	All Three Columns of PT are Filled By the Student	Yes	DOS Machine	BASIC
CIRCSIM-Tutor (v.0)	ITS	Qualita-tive	7 (CC, RAP, SV, HR, CO, TPR, MAP)	3 (DR, RR, SS)	4	No	13	Overlay + Bug Library	Each Column of PT is Filled By the Student	No	DOS Machine	PROLOG
CIRCSIM-Tutor (v.2)	ITS	Qualita-tive	7 (CC, RAP, SV, HR, CO, TPR, MAP)	3 (DR, RR, SS)	4	No	16	Overlay	Each Column of PT is Filled By the Student	Only on Experim-ental Basis	Macintosh	Common Lisp

Figure 3.2 Teaching Systems For BR Reflex: Facts

3.7.1 <u>Domain: Its Nature and the Way it has been Modeled</u>. The knowledge domain used in all systems is cardiovascular physiology, specifically the baroreceptor reflex, that part of the cardiovascular system responsible for maintaining a more or less constant blood pressure (Berne & Levy, 1993). This is a complex domain. The cardiovascular system consists of the heart, a network of blood vessels distributed throughout the body, and a fluid, the blood, that is circulated through the system. The role of the cardiovascular system is to provide every cell in the body with a “source” of those nutrients required for its function (O_2, glucose, etc.) and to provide a “sink” for those products of metabolism which must be eliminated (CO_2, H^+, heat, etc.). The baroreceptor reflex is a negative feedback system. The mechanism by which this reflex regulates blood pressure is as follows. The value of the parameter being regulated, blood pressure (Mean Arterial Pressure - MAP), is sensed, and information about its value is sent to the Central Nervous System (CNS), specifically to cardiovascular centers in the medulla. There it is compared against the set-point (or desired) pressure which is represented in the CNS in some as yet unknown manner. The direction of the “error” - whether the pressure present is higher or lower than the set-point pressure determines the output of the cardiovascular controller, which, working through sympathetic and parasympathetic centers, control the function of the cardiovascular effectors (the heart and the blood vessels). Cardiac output and resistance are thus altered so as to restore MAP toward the set-point level.

The reflex involves a multitude of anatomical components all over the body, but functionally one can think about the reflex in terms of the causal interaction between a limited number of parameters (e.g., cardiac output, mean arterial pressure). Nevertheless, the number of parameters involved, and the complexity of causal interactions that occur, make it difficult for students to master this system.

MacMan (Dickinson et al., 1973) is a mathematical model of the baroreceptor reflex. HEARTSIM uses MacMan only to display the descriptions of the CV responses in graphical and tabular form. In translating HEARTSIM into CIRCSIM AAR & JAM realized that they were making no essential use of the mathematical model that was a part of HEARTSIM. That is to say, the most effective teaching was being generated from the stored correct predictions for each procedure, not from the quantitative outputs generated by the model. Thus, in CIRCSIM not only the correct predictions but also the limited data needed to display the results of each procedure, were stored. There is no mathematical model in CIRCSIM, although all the results displayed by CIRCSIM come from MacMan.

Designers of CIRCSIM-Tutor (v.0) adopted a different approach to model domain knowledge. This approach was based on the AI-tradition of knowledge representation. This time a qualitative causal model of the CV system was developed and used as a domain knowledge base for the system. An almost similar version of this model was also adopted for CIRCSIM-Tutor (v.2). One obvious advantage of this model was that the system's articulation capability was greatly improved. Now it is possible to develop qualitative and causal reasoning behind a change in any part of the model as a result of changes in other part. In a way the system now uses the same reasoning style while "thinking" and "talking." Here we briefly describe the qualitative and causal model used in CIRCSIM-Tutor (v.0) and CIRCSIM-Tutor (v.2).

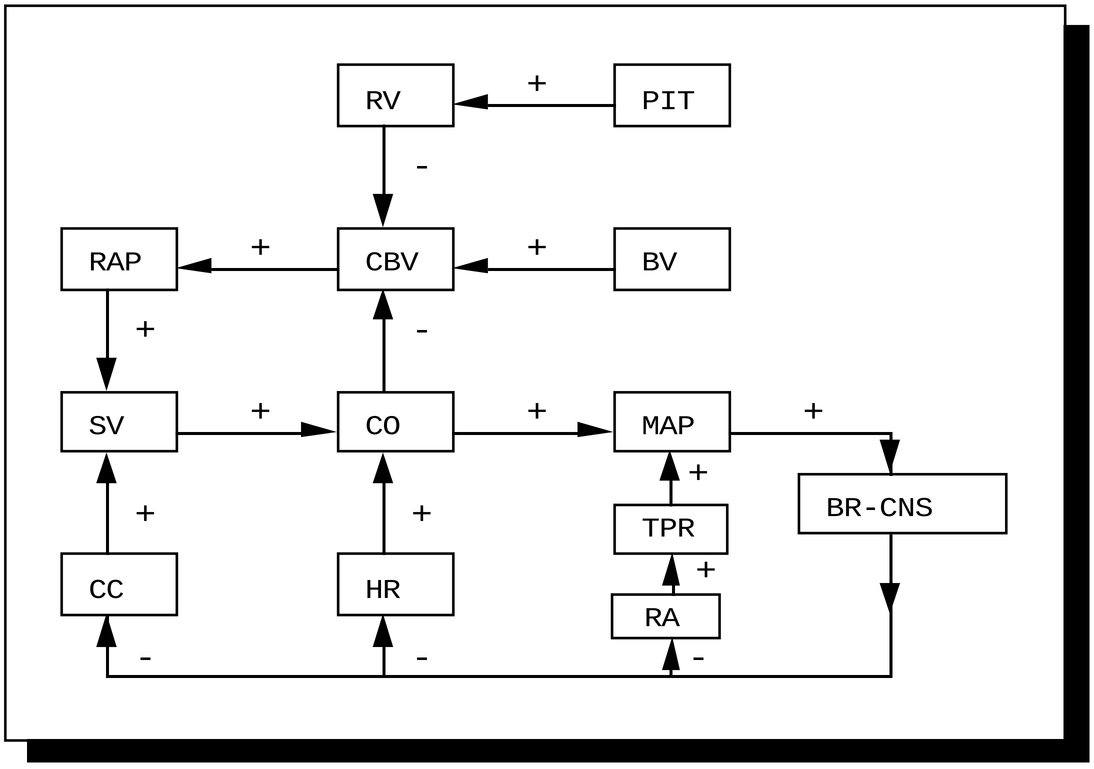


Figure 3.3 Concept Map Used in CIRCSIM-Tutor (v.0)

A graphical representation of the domain model of CIRCSIM-Tutor (v.0) is shown in Figure 3.3. We call this graphical representation the *concept map*. In our context, it represents the key physiological concepts (e.g., physiological parameters, anatomical components involved in a physiological process) and relationships between the concepts (e.g., Heart Rate is directly related to Cardiac Output). The notion of concept map used here is similar to the notion defined by Novak & Gowin (1984). The links used in Novick & Gowin style concept maps are almost always hierarchical or taxonomic. They are rarely causal. Our concept maps only use causal links. The physiological relationships between the concepts in the concept map of Figure 3.3 are represented by directional arrows. These arrows emphasize the unidirectional influence of one concept on the other. “+” and “-” signs on the arrow represent the directly or inversely proportional nature of the causal relationship. The concept map only represents the static nature of the knowledge in the domain model. In order to use this model to simulate the behavior of the baroreceptor reflex one needs rules that control the dynamic behavior of the CV system. These rules are physiology principles that are involved in the functioning of the CV system. In order to fully appreciate the power of this simple qualitative and causal model of CV system to predict the qualitative changes for the key physiology concepts, let us describe very broadly the functioning of this model. This will also show a usage of rules that allow to mimic the dynamic behavior of CV system.

Suppose a patient loses one liter of blood. This model under the given conditions will respond in three different phases. The immediate response of the system to the perturbation is called the Direct Response (DR) and it always results in a change in MAP (a physiology variable in the concept map of Figure 3.3). Due to the given condition Blood Volume (BV) decreases (see Figure 3.3). BV is directly related to Central Blood Volume (CBV) hence CBV will also decrease. CBV is directly related to RAP hence RAP will decrease. If we continue to follow the arrows of Figure 3.3 in a similar manner then following predictions will result: SV -, CO -, MAP -. When we reach MAP, one of the domain rules will disable the propagation of causal influence in this phase of CV system. Here CC, HR, and TPR are neural variables and are unchanged. The remaining concepts of Figure 3.3 are also unchanged because they do not fall in the causal influence propagation path that started from BV and ended at MAP. The change in MAP in DR gives rise to a Reflex Response (RR) that is organized to restore MAP to its normal level. This mean RA +, TPR +, HR +, CC +, CO +, MAP +, CBV -, RAP -, SV -, BV 0, RV 0, PIT 0. Eventually, a new Steady State (SS) is achieved which is the "sum" of the DR and RR responses. This means that the final value of the parameters in the concept map (compared to their values in DR) is: BV -, CBV -, RAP -, SV -, CO -, MAP -, RA +, TPR +, CC +, PIT 0 and RV 0.

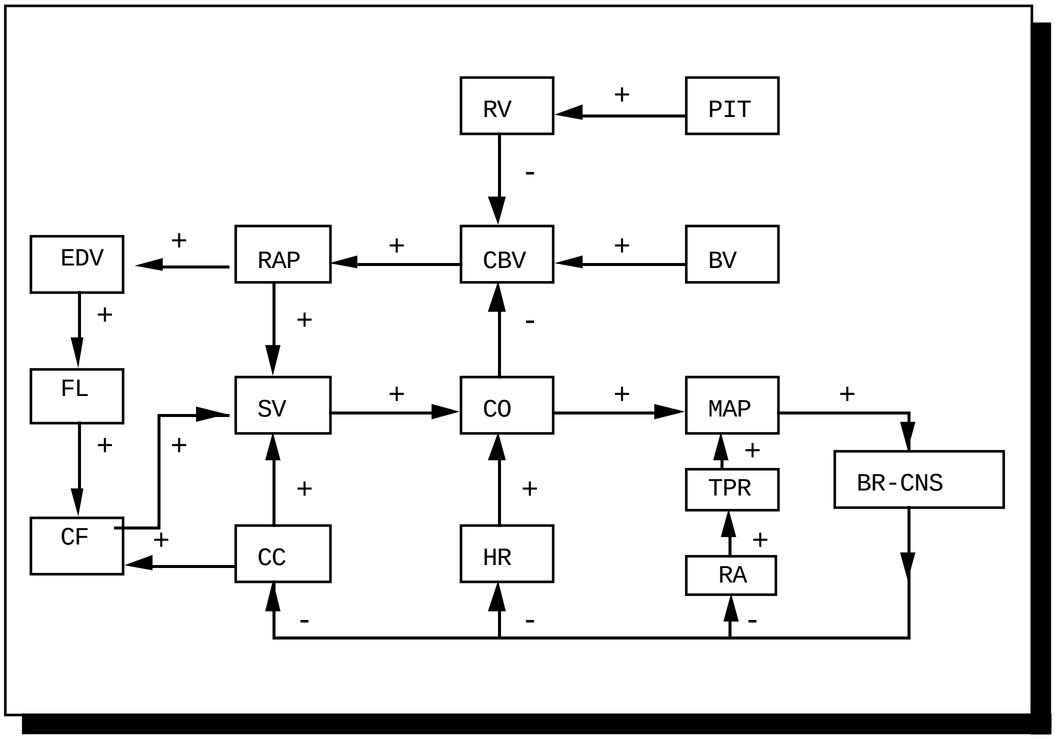


Figure 3.4 Concept Map Used in CIRCSIM-Tutor (v.2)

During the development of CIRCSIM-Tutor (v.0) it was assumed that this model of the CV system will be enough for the tutoring task. But soon it became clear that this model should be expanded to do sophisticated reasoning in the domain. One simple method of introducing sophistication in the model is to introduce more concepts and physiological relationships between them. This was the approach taken by the designers of CIRCSIM-Tutor (v.2). Here the concept map used is shown in Figure 3.4. Notice that only some of the relationships of Figure 3.3 have been expanded with additional paths to generate alternative explanations.

The natural language capability of CIRCSIM-Tutor (v.2) is much improved than CIRCSIM-Tutor (v.0). This also put heavy demands on the domain model in order to generate sophisticated explanations (Zhang et al., 1990). Unfortunately the instructional component of CIRCSIM-Tutor (v.2) did not use the full power of the domain model, instead it stuck to the CIRCSIM-Tutor (v.0)'s concept map. The developers of CIRCSIM-Tutor (v.2) soon learned following lessons regarding the domain model of the system. (1) The current model of the domain needs to be expanded greatly. (2) This expansion should be at different levels so that it can help the tutoring process. (3) Besides the causal phenomena captured by the model, there is a need of modeling other processes (e.g., hydraulic-transfer events that takes place between the organs (Zhang et al., 1990)) in the CV system.

These lessons have been neatly taken care in the design of CIRCSIM-Tutor (v.3). In fact a formal model has been created that takes care of the missing bridge between the domain and tutoring model for CIRCSIM-Tutor. In the following chapters we will describe in detail these developments.

3.7.2 <u>Student Population Using the System and their Learning Context</u>.

Physiology is a compulsory basic science course for the first year medical students. HEARTSIM was originally designed to be an integral part of a physiology course for first year medical students. Considering the goals of this system (see section 3.7.3) a specific time frame was created for its use by the students. It is offered only when the students have gone through a sequence of lectures, one laboratory and a number of small group problem solving sessions (workshops). It is assumed that at this time the students are prepared to carry out the kind of task HEARTSIM asks them to do. CIRCSIM and all versions of CIRCSIM-Tutor are also designed for the same student population and learning context as HEARTSIM.

3.7.3 Teaching Goals. Two main educational goals of all the above systems are that: (1) the students, using the system, acquire a qualitative, causal model of the cardiovascular system, and (2) they learn a problem-solving method that enables them to solve any problem in the domain. It is assumed that the students using the system will have acquired the necessary knowledge through attendance at lectures, reading the textbook, and participating in other scheduled activities in the physiology course. While the system will provide students with missing information that is required to successfully solve problems, this system is not designed to be, and it is not used as, the source of primary learning of the basic physiology knowledge. From experience it is observed that students using the system possess the necessary knowledge to solve CV problem but this knowledge is not in an integrated form. These systems are intended to assist students to integrate their knowledge about the CV system into a mental model of the BR reflex with which they can successfully predict and explain the responses of the CV system to disturbances.

3.7.4 Teaching Environment: The Protocol. In this section we will describe the teaching environment of all systems described above. Here we will also describe the protocol that students use to interact with the system. We will start with HEARTSIM because it is this system that introduced many novel concepts. Most of these concepts are then inherited by other systems (i.e., CIRCSIM and CIRCSIM-Tutor).

HEARTSIM forces the student to develop a qualitative reasoning method to solve CV problems. It basically offers a problem-solving environment in which the student is required to solve CV problems. These problems are designed in such a way that their solutions require the student to use their mental model of the CV system. HEARTSIM starts by presenting a list of available procedures. These are stimuli to the CV system for which the program provides didactic feedback (Rovick & Brenner, 1983). Once a procedure is selected the system instruct the student to predict the response of CV system for a set of parameters by filling the entries in a table called *predictions table*. A detailed description of this table is provided in section 3.7.5. Entries in this table are made by touching the monitor screen. One touch enters an up arrow, a second enters down arrow and the third enters zero for increase, decrease and no change, respectively (Rovick & Brenner, 1983). Once all entries in the table have been made by the student, the system graphically illustrates the response and provides a data table. It then evaluates the student's responses and provides didactic feedback. A detailed description of types of error detected by the system is provided in section 3.7.7. The feedback provided by the system is in textual, numerical and graphical form. At the end of a procedure the student may select another (or the same) procedure from the available procedure list. HEARTSIM provides a guided procedure to the student who is using the system the first time. This procedure allows the student to step through the protocol of the system under the guidance of the program.

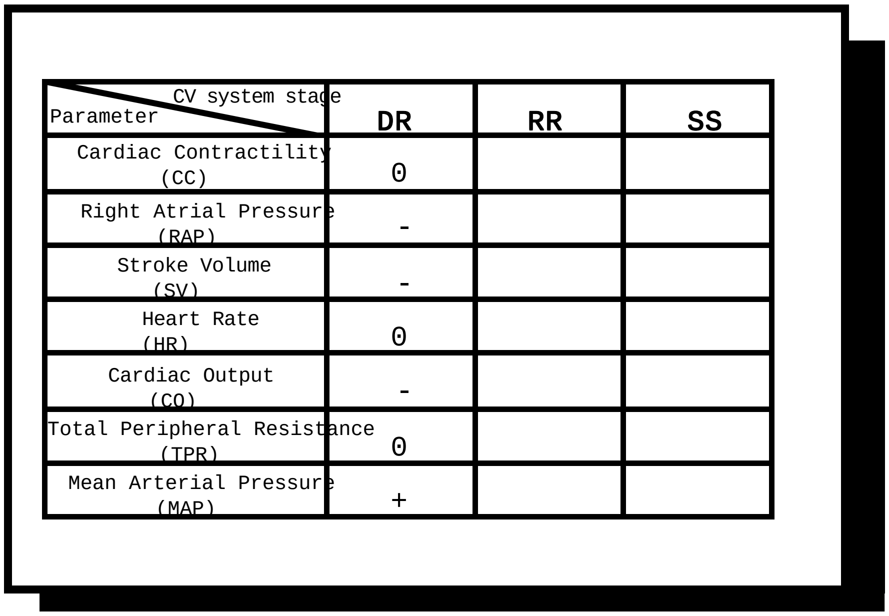
CIRCSIM inherited the same teaching environment as HEARTSIM with the exception that it is a PC based system. In CIRCSIM the process of filling in the prediction table is done by the student using the keyboard. Here the error evaluation and feedback processes are comparatively more advanced than those in HEARTSIM. These processes are described in more detail in section 3.7.7 and 3.7.8 respectively.

CIRCSIM-Tutor (v.0) brought a paradigmatic shift in the line of research done at Rush. This system made explicit many of the decisions (knowledge types) that were implicit in HEARTSIM and CIRCSIM. But it still retained the tutoring environment of CIRCSIM to a great extent. Since CIRCSIM-Tutor (v.0) was a prototype many ideas were tried in this version. One environment that could be definitely attributed to CIRCSIM-Tutor (v.0) was not radically different from the CIRCSIM environment.

During the prediction phase the system displayed a concept map of the CV system. As soon as the student made an acceptable prediction for a variable, the system highlights that variable in the concept map. Also the system frequently interrupts the prediction phase of the student to provide hints for the incorrect prediction. CIRCSIM-Tutor (v.0) is not a completely developed system as were HEARTSIM and CIRCSIM but the major contribution of this system is that it provided a feasibility study for an ITS in the domain of baroreceptor reflex.

Developers of CIRCSIM-Tutor (v.0) soon realized that the successful experience with CAI systems (i.e., HEARTSIM and CIRCSIM) is good but not enough to develop a successful ITS system because unlike CAI system, ITSs are tutoring systems. In order to fully appreciate the full blown complexity of a tutoring system one needs an empirical study of tutoring that could provide a basis for the development of an ITS. This observation is not uncommon in the ITS field (see Galdes, 1990). Immediately after the development of CIRCSIM-Tutor (v.0) a series of empirical studies were performed by Dr. Allen Rovick and Dr. Joel Michael at Rush Medical College (see Figure 3.1). A brief detail of these studies is provided in section 3.7.6. CIRCSIM-Tutor (v.2) was the first system whose design also used the analysis of these empirical studies (Woo et al., 1991). These empirical studies were themselves very much influenced by the CIRCSIM experience. The tutoring environment of CIRCSIM-Tutor (v.2) inherited many features from CIRCSIM but one of the major change this time was that the student is tutored as soon as a column of predictions table is completed by him/her. In other words the tutoring process is initiated as soon as the student has predicted the behavior of CV system for one of its phases (either DR, RR, or SS). Also, like CIRCSIM-Tutor (v.0), this system interrupts the student’s prediction process by hints so as to keep the student on the right track (see section 3.7.8). Here there is no guided procedure available for the student.

3.7.5 <u>Prediction Table: A Multi-Purpose Tool</u>. Like many other novel concepts, the idea of the prediction table was first invented during the development of HEARTSIM. This concept was later found so useful that all the systems inherited it. The usefulness and generality of this table is such that it has been used in some other teaching systems (such as GASP, ABASE (Li et al., 1992) - see Rovick & Michael, 1992 for more details).



CV system stage / Parameter	DR	RR	SS
Cardiac Contractility (CC)	0		
Right Atrial Pressure (RAP)	-		
Stroke Volume (SV)	-		
Heart Rate (HR)	0		
Cardiac Output (CO)	-		
Total Peripheral Resistance (TPR)	0		
Mean Arterial Pressure (MAP)	+		

Figure 3.5 The Prediction Table

The prediction table (see Figure 3.5) is a two dimensional matrix whose left hand column lists the physiology variables for which the predictions in a problem are sought and whose top row lists the time frames (DR, RR, and SS). Each entry in a cell in the predictions table can have a value of: increase (marked as “+” or an up arrow or “i (I)), decrease (marked as “-” or a down arrow or “d (D)) or no-change (marked as “0” or “u (U)). This tool has multiple purposes. It is used by the system to collect the student’s predictions for the desired physiology variables in a problem, assess these responses and the underlying student knowledge (Rovick & Michael, 1992). It is used by the student as a record keeping tool and hence expands his/her memory by visual means. The names of the physiology variables listed in the first column remind the student all the time of the core domain concepts for which predictions are sought. See Figure 3.2 for a complete list of physiology parameters used in different systems. Predicting the qualitative behavior of the CV system a stage at a time (i.e., DR, RR, or SS) is a strategic decision of the system and the prediction table portrays this visually by representing these three stages as three separate columns. A detailed description of the use of the prediction table in assessing student's knowledge can be found in (Rovick & Michael, 1992).

3.7.6 Human Tutoring Experiment: Keyboard-To-Keyboard Sessions. Two major reasons for performing empirical study of tutoring for CIRCSIM-Tutor were to understand the nature of the tutoring language and the tutoring processes used in our tutoring context. A series of such studies started immediately after the development of CIRCSIM-Tutor (v.0) (see Figure 3.1). To date 48 keyboard-to-keyboard sessions have been conducted over a period of 5 years. This section explains in some detail the nature and method used to conduct these tutoring sessions.

The tutoring sessions that were conducted all involved the student making predictions about the response of the baroreceptor reflex to a perturbation with the tutorial dialogue that aimed at correcting student errors and assisting the student to explain the responses he or she was describing. The goal of the tutor, then, was to assist the students to make correct predictions and explain them.

Tutoring sessions always occurred just shortly before the students were scheduled to use CIRCSIM in a computer laboratory setting. The students were thus nominally prepared to carry out the kind of problem solving they were to asked to do.

The tutors in these sessions were Allen Rovick and Joel Michael, both of whom are professors in the Department of Physiology and both of whom are teachers in the physiology course being taken by the student subjects. Both tutors are middle-aged males. The students were first year medical students at Rush Medical College, and they were paid volunteers who understood that their participation in the experiment would have no bearing on their grade in the course, although they were told that they would learn something of relevance to them (the advertisement used to recruit participants is headed “EARN WHILE YOU LEARN!”). Students participating in the tutoring sessions were selected only on the basis of their availability, and no information about their performance in the physiology course was available when they were recruited to participate. The student participants were male and female, with a range of ages from 21 to 37 years (mean of 25 years). See Figure 3.6 for some more facts about these tutoring sessions.

Two obvious methods for capturing a tutoring session are audio-taping and video-taping with subsequent transcription of the dialogue. The fidelity of transcription, what non-lexical elements get coded, will, of course, dependent on the nature of the analysis to be pursued.

Keyboard -to- Keyboard Sessions	TUTOR(S)	PERIOD	APPROX. DURATION FOR EACH SESSION (HOURS)	PROTOCOL (VERSION #)
K1-K8	AAR	11/10/89 - 11/17/89	1	1
K9-K24	AAR & JAM	4/23/90 - 5/3/90	1	2
K25-K28	AAR & JAM	4/23/91 - 4/25/91	1.5	2
K30-K38	AAR & JAM	11/10/92 - 11/13/92	2	3
K39-K48	AAR & JAM	4/27/93 - 5/1/93	1.5	3

Figure 3.6 Facts About the Keyboard-to-Keyboard Sessions

Although the first study employed audio-taping, and subsequently some sessions were video-taped, the approach that was adopted directly captures the tutorial dialogue. This approach employs two linked computers with the student and the tutor communicating by typing at the keyboard and reading the comments of the other on the computer screen.

This approach was adopted for two quite different reasons. Pragmatically, it was found that even simple, direct transcription of tapes (whether audio or video) was extremely time consuming. Furthermore, any attempt at a higher fidelity of transcription would require personnel training and would be even more time consuming.

Equally important, however, it was realized that the communications channel available to, CIRCSIM-Tutor would be a very narrow one, compromised of only text entered at the keyboard by the student and text generated by the tutor and displayed on the computer screen. Thus, all of the non-verbal clues that are normally present in a face-to-face tutoring session, such thing as tone of voice, pauses, facial expressions, etc., would be unavailable to CIRCSIM-Tutor.

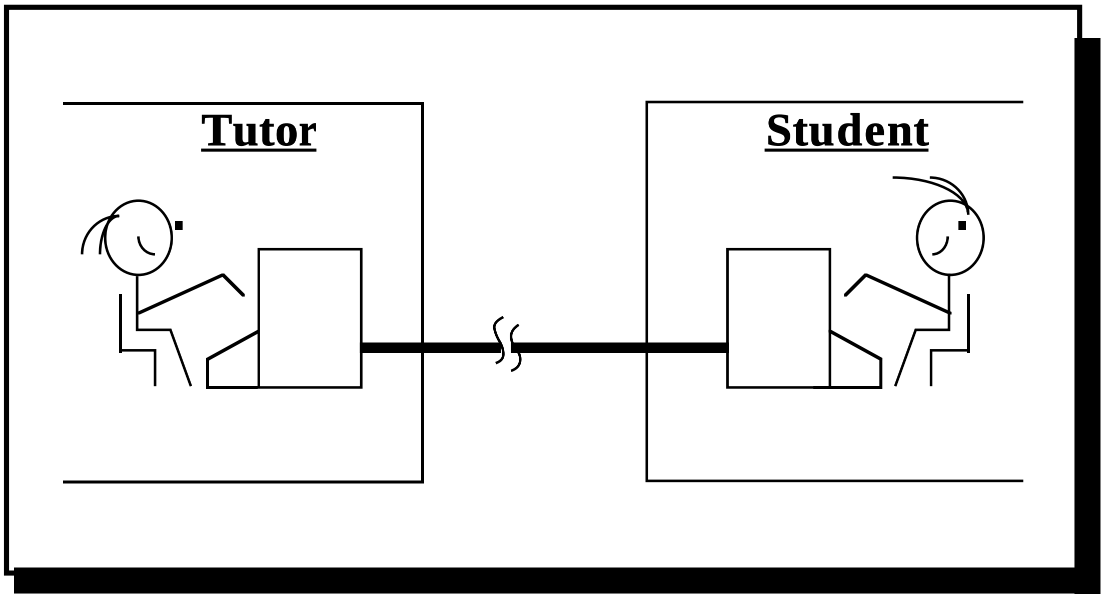


Figure 3.7 Student and Tutor in a Keyboard-To-Keyboard Session

Thus, for reasons of economy and in order to gather data about tutoring that would most nearly simulate the situation to be encountered in CIRCSIM-Tutor, the experimenters decided to develop a computer-based system with which the tutor and the student could comfortably communicate with one another. The experimental setup for the resulting keyboard-to-keyboard method of conducting human tutoring experiment is described below.

In the keyboard-to-keyboard method the tutor and the student were seated in two separate rooms (see Figure 3.7). A special software package, called the Computer Dialogue System - CDS (Li et al., 1992), has been developed which allow the tutor and the student to communicate with each other, over a telephone line or a direct cable connection, by typing at the computer keyboards. During the course of communication, the two users take turns typing. Each character typed by one user is displayed on both screens almost simultaneously. CDS also provides the users with the ability to mimic some verbal characteristics of natural dialogues, such as interruption of one participant by the other, without losing the message (Li et al., 1992). Finally, the dialogue information on the date and participants and the timing information are well formatted and saved on the hard disk of one of the computers. A more detailed description of this system can be found in (Li et al., 1992).

Before the transcript of a keyboard-to-keyboard session is analyzed the dialogue captured by the CDS is processed through a numbering program which using a numbering scheme tags each sentence or fragment with an identification number. The current numbering scheme has the following format (Li et al., 1992):

session # - who - turn # - sentence #

For example, the number k47-st-057-01 indicates that the sentence comes from keyboard session 47, that the student is typing, that this is turn 57, and that it is the first sentence in that turn. A sample segment from a keyboard-to-keyboard session is shown in Figure 3.8. Details of the tutoring environments used in these sessions is given in section 5.5.2.

K30-tu-34-2:	Let's talk about your predictions.
K30-tu-34-3:	Firs, what does DR mean?
K30-st-35-1:	It's the immediate response to gravity
K30-tu-36-1:	Yes.
K30-tu-36-2:	It's what happens before any reflex changes can occur.
K30-tu-36-3:	Do you want t o change any of your predictions?
K30-st-37-1:	Half the answers that I gave were direct responses and the other were reflexes.
K30-st-37-2:	Do you want me to list the parameters .
K30-tu-38-1:	Yes.
K30-tu-38-2:	Which of the parameters would not be affected in the DR period?
K30-st-39-1:	Cardiac contractility
K30-tu-40-1:	Good.

Figure 3.8 A Sample Segment From a Keyboard-To-Keyboard Session

3.7.7 <u>Cognitive Diagnosis: A Way the System Uses Multiple Student Inputs to Model the Student's Knowledge State</u>. In this section we will briefly describe the student modeling process used in all CAI and ITS teaching/tutoring systems developed for BR reflex. Here once again we will start with HEARTSIM because it pioneered many ideas including for student modeling. Here we will not dwell on the issue of the differences between CAI and ITS methodology for student modeling but briefly CAI systems carry out no dynamic student modeling. On the other hand, ITS systems have the potential to build a model of each student being tutored from the student's responses (VanLehn, 1988).

As mentioned before as soon as a procedure is selected, HEARTSIM asks the student to predict the behavior of the CV system, i.e., to predict 24 values for the eight variables in the prediction table (see Figure 3.1). These multiple student inputs provides particularly rich information about the student's cognitive state, and can thus more readily provide information with which to build student model or determine the pattern of response selection. HEARTSIM does not have an explicit student model but it exists implicitly in its decision making process. Basically there are three different ways in which HEARTSIM classifies student input. First, HEARTSIM checks for the logical errors that mismatch transient and steady state changes. Next it checks for the errors that mis-state the relationship between two or more variables in a physiological relationship and finally it detects errors which mispredict the effects of the experimental procedure (Rovick & Brenner, 1983). The first two classes do not specifically relate to any particular CV procedure. Hence they are reviewed immediately after the prediction table has been completed. The third class relates to the specific procedure that is about to be carried out. These errors are reviewed after the computer has simulated the effects of the procedure (Rovick & Brenner, 1983). In other words each error of the student in the prediction table triggers some pattern of error in the system that in turn triggers the appearance of the text intended to remediate the assumed source of the particular error. HEARTSIM does not engage the student in a dialogue hence it does not have a means of confirming the source of an error in the student's prediction. Here the responses generated by the system are based on the extensive experience of expert tutors in the domain.

CIRCSIM inherited the same student modeling capability from HEARTSIM (Michael et al., 1992). The only difference, here, is that the ordering of error patterns has been guided by the solution algorithm that is portrayed explicitly by CIRCSIM. Also it tries to group error patterns for feedback as much as possible.

CIRCSIM-Tutor (v.0) is a prototype for the intended smart tutor for the BR reflex. It conceptually created concepts for overlay and bug library models for student modeling (Kim, 1989) but these were not implemented as a fully working system.

CIRCSIM-Tutor (v.2) incorporated an overlay model. The overlay modeling approach assumes that the student's knowledge is a subset of the expert knowledge (VanLehn, 1988). The overlay model describes whether or not a student has knowledge of a particular subject material. It does not say what kind of incorrect knowledge or what misconceptions have caused the student's current incorrect responses (Clancey, 1987b).

Unlike HEARTSIM and CIRCSIM, CIRCSIM-Tutor (v.2) requires the student to predict only a single column of the prediction table at a time. Once the students complete their predictions the system tutors for the incorrect responses before asking them to continue predicting for the remaining columns (Woo, 1991). Each error in the prediction table points to an error pattern that usually is a physiology relationship. Due to the overlay modeling approach, the system assumes that the sensitized relationship is missing from the student's knowledge base. As a result a lesson is planned to provide this missing piece of knowledge.

3.7.8 Conceptual Model of Teaching/tutoring: What, When, and How to Convey Knowledge. A conceptual model of teaching or tutoring in its broader sense contains both the pre-session and the in-session behaviors observed in a human tutoring environment (see Chapter V for more details). The in-session behavior deals with deciding about what, when, and how to teach the subject material. In this section we only discuss the in-session behavior for all systems described above.

The conceptual model of teaching for HEARTSIM and CIRCSIM are relatively simple compared to CIRCSIM-Tutor because in these CAI systems no interactive dialogue is initiated by the system. As a result these systems adhere to their initial diagnosis of the student which they make immediately after predictions have been made by the student. Topics discussed by the system here are wholly based on the set of errors present in the prediction table. These systems have a set of well defined error/topic ordering rules. According to these rules the system first discusses logical inconsistencies, such as SS predictions that do not agree with the combined DR and RR predictions for a specific variable. Next the system discuses predictions that violates essential physiological relationships (e.g., MAP = CO x TPR). Although the system prompts for these error patterns and provide a chance for the student to correct the actual predictions triggering these error patterns, no remedial feedback is provided if the student is unable to correct these errors. The third type of errors are procedure specific and are discussed after the simulation is run.

HEARTSIM and CIRCSIM differ conceptually while making ordering decisions for this third type of errors. In HEARTSIM no explicit attempt is made to expositorily convey an algorithm to solve a CV problem. CIRCSIM does this by ordering this third category of error patterns ("bugs") according to a solution algorithm with the hope that the student will "get it" and use it to solve other CV problems. Each presentation block that conveys the feedback for each error pattern is well crafted with the assumption that it, most likely, will remediate the source of this error pattern. The feedback, in general, may:

(1) describe or discuss the underlying physiology,
(2) summarize the changes observed in the simulation, or
(3) ... review lesson on the baroreceptor reflex ...
(Rovick & Brenner, 1983).

In addition to this, in CIRCSIM, a grand summary is provided at the end of each procedure that describes the behavior of the physiological mechanisms that are involved with the regulation of blood pressure under the given condition.

CIRCSIM-Tutor (v.0) using ITS tradition distinguishes explicitly between domain knowledge and tutoring knowledge (Kim, 1989). It, like CIRCSIM, tries to make explicit the problem solving algorithm that is a part of its domain knowledge. This is accomplished in CIRCSIM-Tutor (v.0) in two steps. During the prediction phase, in which the student predicts the qualitative value of physiology variables, system interrupts if the student's prediction violates the rules of the general algorithm. Here the system provides generic heuristics that are aimed at bringing the student at a right track. In the second step the system tutors the wrong predictions in a sequence that conforms to the general algorithm. Here topic selection is based on the patterns of error present in the predictions table and the dialogue that goes on between the system and the student. Instead of pre-stored presentation blocks for feedback, here system plans an instructional interaction. The default tutoring method used here uses the discovery method but the hinting process of this system is very primitive.

CIRCSIM-Tutor (v.2) inherited many ideas from CIRCSIM-Tutor (v.0). But unlike CIRCSIM-Tutor (v.0) it bases its planning decisions only on the overlay model. Here topics are ordered according to their importance rather than on the sequence of the general CV problem solving algorithm. Here the system emphasizes discovery method rather than conveying the goals of the system in an expository style. Unlike the conceptual model of CIRCSIM-Tutor (v.0), CIRCSIM-Tutor (v.2) uses the ideas extracted from the empirical studies performed at Rush (see section 3.7.6) (Woo, 1991).

CHAPTER IV

AN ITS DEVELOPMENT FRAMEWORK: KNOWLEDGE ACQUISITION METHODOLOGIES

The field of Intelligent Tutoring System (ITS) needs a systematic and comprehensive development methodology. This conclusion nowadays appears quite frequently in the ITS literature (Halff, 1988; Clancey, 1992; Breuker, 1990; Khuwaja et al., 1994a). Although this field has existed for more than two decades, most of the early research effort concentrated on basic research issues involved in the process of tutoring rather than on the development of tutoring systems that could exist as instructional units in a real educational environment.

None of the early versions of CIRCSIM-Tutor used any explicit ITS development framework. Being the designer and developer of several of CIRCSIM-Tutor (v.3)'s major components, I am strongly affected by this deficiency in the ITS field. This chapter describes an ITS development framework that I have developed to guide my research. Although much further research is needed to make this framework complete and comprehensive and although this framework is not the major theme of my research, it has indeed been helpful. I view knowledge acquisition as a modeling process. This chapter explains this underlying philosophy and describes different knowledge acquisition methodologies used to develop a model of tutoring - a central theme of this thesis.

4.1 <u>A Systematic ITS Development Framework: Influences and Structure</u>

An ITS is a Knowledge-Based System (KBS). Recently a number of systematic KBS development methodologies have become popular. KADS (see Section 2.6) is one of the forerunners among these methodologies. This methodology is very general and therefore applicable for a wide variety of knowledge-based systems. It takes the view that KBS development is essentially a modeling activity that yields a series of models at different levels of abstraction. Since this methodology does not consider the educational aspects of machine tutors, using this alone would not satisfy all requirements for developing an ITS.

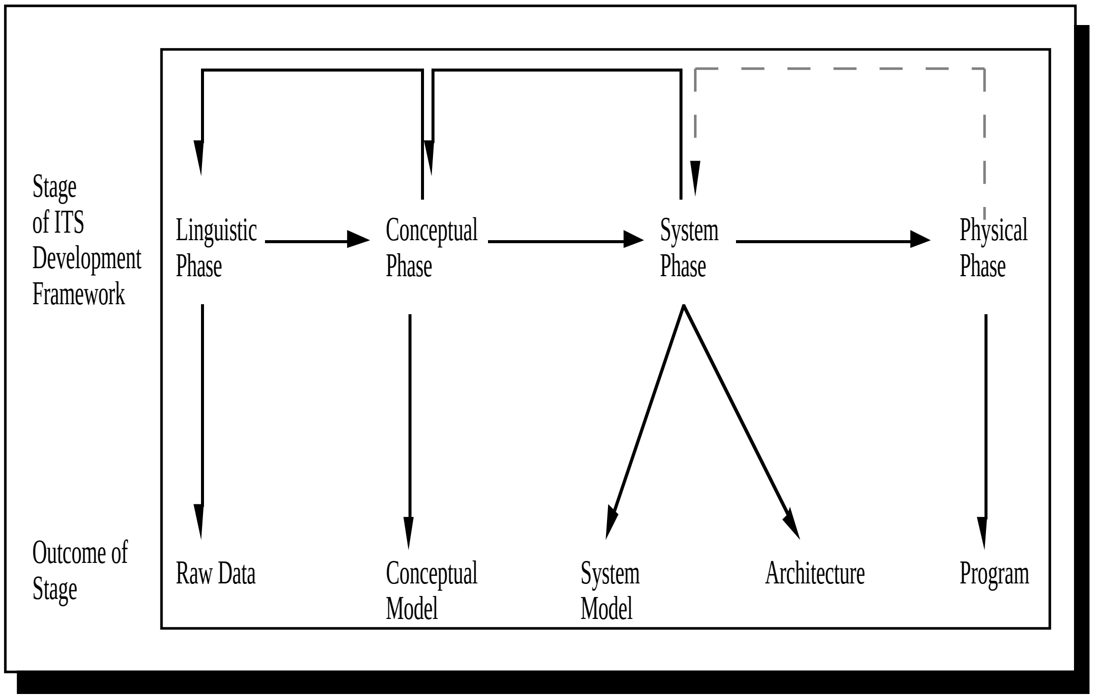


Figure 4.1 Stages in the ITS Development Framework

The field of Instructional System Design (ISD) has a long history as a discipline that is concerned with understanding and improving one aspect of education: the process of instruction (Reigeluth, 1983b). Section 2.5.4 has listed a set of avenues in which ISD can help the field of ITS. One of the major failures of the field of ITS is that it does not have a systematic design process that is tailored to develop instructional systems that can be a part of the real educational environment. ISD can help ITSs to achieve this goal (Halff, 1988).

This section will describe an ITS development methodology that combines the attractive features of a systematic KBS development methodology with some design prescriptions from the field of ISD. The KBS development methodology used here is KADS (Wielinga & Breuker, 1990). This framework is not complete and further research is needed so that it can be used to develop a complete, coherent and widely applicable methodology for ITS development. But in its current form it is sufficiently structured so that I have used it as a research guide.

In the first chapter I viewed an ITS from three perspectives (or viewpoints). These views are: the conceptual view, the system view, and the physical view. In this framework these three views form the major stages (or phases) of ITS development. These stages are arranged in this framework as shown in Figure 4.1. Like KADS this framework views the ITS development as a modeling activity that yields a series of models at different stages of its development.

ITS development here starts at the linguistic phase. Here raw data for ITS development (e.g., knowledge from text books, interviews, think-aloud protocols) is identified.

In the second phase, the raw data from the first phase is analyzed to develop conceptual model(s) of ITS. These models are sufficiently abstracted to be free from any issue of implementation. Here the KADS's model of expertise (Wielinga & Breuker, 1990) can be used to organize knowledge of the system. This model views expertise as consisting of knowledge at four different levels - domain, inference, task, and strategic. See Section 2.6 for a brief description of each level. A detailed description of this model of expertise can be found in (Wielinga & Breuker, 1990). In this second phase the major emphasis is on the conceptual issues underlying the tutoring behavior. If the method used here is to observe the behavior of expert human tutors (see Section 2.2.3) then in this phase the conceptual models of different behaviors (or roles) of the human tutor are developed. This phase is at the highest level of abstraction. Here no consideration is given to realizing these models in a machine form.

It is the next phase that causes my framework to deviate from KADS and at the same time makes it more specific to ITS development. This phase is also at the same abstraction level as the conceptual phase. In this phase, an ITS is viewed as an instructional system. In other words, in this phase, the focus is on “system” issues rather than on the “expertise” driving the system. Here a design process used in the field of ISD could be used to shape the development of an ITS. Most of the ITSs developed have avoided this phase. One consequence of this is: most ITSs cannot take their place in real educational settings. The system phase has two subphases. The first subphase emphasizes the instructional system point-of-view, whereas the second subphase emphasizes the software system point-of-view. From the instructional system point-of-view the designer views the ITS as an instructional system. The outcome of the first subphase is a set of system model(s) of the ITS. These models are more “complete” than conceptual models. The conceptual phase, in other words, provides a theory of tutoring that is first realized in these system model(s) of the ITS. The second subphase brings the ITS development one step closer to its physical realization as a computer program. Here the developer views the ITS as a software system. In this subphase software engineering principles shape the system model(s) into a coherent architecture. The key issue here is to keep this architecture independent of the implementation formalism.

The next step is to code the architecture of the ITS, using some implementation formalism (e.g., a generic programming language, an expert system shell), as a software program.

The sequence of these phases defines the normal system realization mode. Some of these phases influence decisions made earlier. For example, the system model might force the developer to change or perform further investigations at the conceptual level. This in turn might require further empirical studies at the linguistic level. As a result, it is not uncommon that iterative development takes place between these phases. It will be advantageous to avoid, as much as possible, the iterations between the physical phase and the system phase. A careful use of software engineering principles for the design of the architecture of the system can hopefully reduce the risk of this expensive iterative loop.

Figure 4.1 shows that the conceptual model influences the data gathered at the linguistic phase and the only thing that influences the system model at first is the conceptual model in the conceptual phase. This is a typical data-driven design scenario. In a practical system design this might not be the case. Sometimes many constraints for the ITS have already been decided on, for example, the tutoring style of the system (e.g., Socratic dialogue), the type of tutoring environment (e.g., based on problem solving), the type of the student population (e.g., college students), etc. All these constraints sketch a very broad picture for the conceptual model, which in turn provides constraints for the empirical study required at the linguistic level. Data collected from this study further helps to develop a conceptual model and, if needed, further constrains the activities at the linguistic level. This iterative approach to system development causes alternation between the system and conceptual phases. A theory of the development of the system model already has been worked out even before the conceptual model is developed. This system model can constrain the activities at the conceptual phase. In other words this ITS development framework can support data driven development, theory driven development, or a mix of these two styles of system development.

4.2 Role of the Multiple Expert Metaphor in the ITS Development Framework

ITSs are complex systems. It is always helpful to use tools that break complex behavior into parts in order to manage complexity. The multiple expert metaphor is one such tool that at a very high level of abstraction, breaks complex tutoring behavior into four, commonly accepted, parts - domain expert, student expert (or student modeler), pedagogy expert, and communication expert (Breuker, 1990; Self, 1988; Wenger, 1987). According to this metaphor, in an ITS these experts communicate with each other and coordinate their activities to create effective tutoring behavior. This metaphor is so popular that it has served as a design model for many ITS projects during all phases of their development. Unfortunately, the separation between these experts is preserved even in the physical phase. Breuker (1990) has criticized overuse of this metaphor for ITS development.

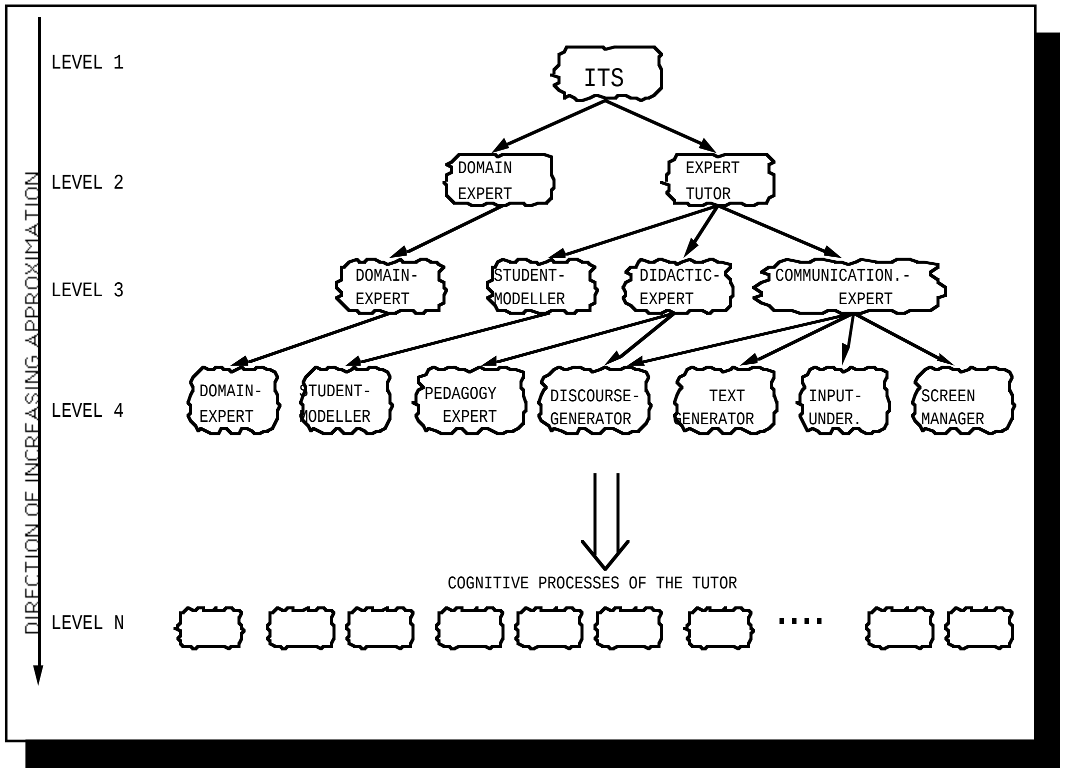


Figure 4.2 Increasing Complexity of the Conceptual Model of ITS.

In the conceptual phase one of the major activities performed is to break the functionality of the target tutoring behavior into appropriate components (Khuwaja et al., 1994a). A detailed analysis of tutoring suggests that there are different levels at which this functionality of the tutoring behavior can be broken down. Figure 4.2 shows some possible levels. For example, the second level states that the tutoring behavior of an ITS can be described as a function of both domain and tutoring expertise. The successive levels of Figure 4.2 define an increasingly close approximation to the behavior of a machine tutor. The actual level used by the developer depends upon the issues under consideration. The third level defines the multiple expert metaphor. It is clear from

Figure 4.2 that the multiple expert metaphor is only one possible level in the conceptual space of ITS. It is up to the designer of the ITS to decide how much functional decomposition is desirable at a high level of the conceptual model. Many ITSs developed to date have used the multiple expert metaphor as their last level of conceptual decomposition. Although this ITS development framework does not restrict the use of the multiple expert metaphor at the conceptual level, it certainly does not encourage the developer to carry this metaphor over to the physical phase.

It is not uncommon for complex tutoring projects to be developed by a team of developers. This is the case with CIRCSIM-Tutor. In such situations it is common to divide the responsibility of developing different conceptual components of an ITS among the team members. The ITS development framework, described above, is still applicable in such situations. Here a developer focuses only on a part of the complex tutoring behavior. In such situations he/she has to still go through all the phases of this framework. Individual component models developed by each developer need to be integrated at some point to build a complex and coherent design. It is advisable to integrate all types of models (conceptual, system, architecture). Doing this might trigger some aspects of the ITS that are previously ignored/overlooked. It is often true in such situations that the whole is more than sum of its parts!

While developing the architecture of an ITS this framework encourages the developer(s) to use software engineering principles to transform the system model into appropriate modules (architectural components). These modules are relatively low level components and define part of the functionality of an expert in the multiple expert metaphor.

4.3 A Usage of ITS Development Framework For CIRCSIM-Tutor (v.3)

No explicit system development methodology was used for the development of earlier versions of CIRCSIM-Tutor and no clear cut separation was made between the conceptual and physical design of the system. But seen from the viewpoint of this development framework several decisions about different phases (models) had already been crystallized by the systems developed before and after the CIRCSIM-Tutor era (see Section 3.1). During the linguistic phase, for example, a considerable number of empirical studies have been performed (see Section 3.7.6 for a description of the methodology). Research on CIRCSIM-Tutor (v.2), to some extent, has crystallized its conceptual model based on the analysis of these empirical studies. At the system level, for example, many decisions regarding tutoring environment, tutoring protocol, student population, and learning context have been made by the pre and post CIRCSIM-Tutor era systems. See Chapter III for a detailed discussion of these early systems/versions. CIRCSIM-Tutor (v.3) has inherited many characteristics from earlier systems/versions. For this recent version decisions have already been made about the high level functional analysis of its conceptual model. Level Four of Figure 4.2 portrays this decision. As mentioned earlier, I have used this developmental framework to guide my research and development on the domain and the pedagogy experts for CIRCSIM-Tutor (v.3).

At the linguistic level I have used various knowledge acquisition techniques to collect the raw form of expertise (i.e., empirical data) that, in the next phase, has been analyzed to develop a model of tutoring - a central theme of this research. This model only accounts for behavior associated with the domain and pedagogy expert of the expert human tutor. The development of this conceptual model is not achieved by one pass from the linguistic to the conceptual phase, instead several iterations have been performed between these two phases. Section 4.4 gives a detailed account of the various knowledge acquisition techniques used to collect the raw form of expertise, the empirical data. Chapters V and VI describe this conceptual model in detail.

The development of the system model is primarily inspired by the theoretical work done by Lesgold (1988). This model has forced me to rethink/evaluate the assumptions made at the conceptual phase. Chapter VII describes the system model for CIRCSIM-Tutor (v.3). Although it is difficult to ignore the influence of the conceptual and the system model on the architecture of an ITS, I have made an effort to lay down a foundation for an architecture such that only minimum effect is required to make a major change in the conceptual and system model of the system. Chapter VIII describes this architecture. Before I began the decision had already been made to use Common Lisp as the implementation language for CIRCSIM-Tutor. For the purposes of my research I chose to use CLOS, the Common Lisp Object System, as the implementation formalism.

4.4 Knowledge Acquisition: A Modeling Activity

Throughout the period of research reported in this thesis, I played the role of knowledge engineer. I used several techniques to collect data to develop a model of tutoring. The following sections describe these techniques in some detail. But first, I describe a recent point of view that is becoming popular in the knowledge engineering community. This view is based on the belief that knowledge acquisition is a modeling activity. Although I have not done knowledge acquisition explicitly using this perspective, methodologically our approach agrees with this theoretical standpoint.

Usually the knowledge engineering methods used in the first generation of knowledge based systems are based on epistemological presuppositions that are fundamentally at odds with much current research in cognitive science (Ford et al., 1993). For example, one very popular analogy in the knowledge acquisition literature is the "mining analogy."

> This analogy suggests that our eliciting knowledge from experts involves "mining those jewels of knowledge out of their heads one by one" (Feigenbaum & McCorduck, 1983). The underlying assumptions are that there exists some "gold standard" of knowledge and that the expert has captured a discrete (presumably large) part of the "reality" governing observed events in the domain (Ford et al., 1993, p. 10).

An associated concept with this analogy is that of "knowledge acquisition bottleneck." According to this concept the problem of developing a knowledge base is to squeeze a large amount of already formed concepts and relations through a narrow communication channel (Clancey, 1993).

These ideas have been challenged by a new wave in the knowledge acquisition field for the second generation of KBSs. This movement brings ideas from constructivist epistemology and social cognition to create new methodologies for knowledge acquisition. Ford & Bradshaw (1993b) is a good reference for this new view of knowledge acquisition. According to this view knowledge acquisition is a modeling process, not merely an exercise in “expertise transfer” or “knowledge extraction.” In other words

> Knowledge acquisition is ... a cooperative enterprise, in which the knowledge engineer and expert collaborate in constructing an explicit model of problem solving in a specific domain. This external model is largely based on the expert’s internal mental “model” of the domain. Thus the product emerging from the knowledge-acquisition process is a model of a model (Ford & Bradshaw, 1993b, p. 2).

Clancey (1993) describes this new perspective: “knowledge acquisition is a process of developing qualitative models of systems in the world - physical, social, technological - often for the first time, not extracting facts and rules that are already written down and filed away in an expert’s mind” (p. 33). In Ford & Bradshaw’s (1993b) words, Clancey (1993) reminds us that from this new knowledge acquisition methodology

> ... the resulting computer models may prove useful for some set of tasks at hand, we must never forget that they are not once-and-for-all versions of the world. Rather, they are specific static artifacts bounded by their context, confounded by our individual interpretations, and severely limited (in comparison with humans) in their flexibility and creative potential (p. 2).

4.5 <u>Knowledge Acquisition in the CIRCSIM-Tutor Context: A Collaborative Process Between Knowledge Engineer and Expert</u>

Ever since the first version of CIRCSIM-Tutor, the knowledge engineering expertise has been provided by the researchers at Illinois Institute of Technology under the supervision of Professor Martha Evens. In this project Dr. Allen Rovick and Dr. Joel Michael act as domain experts. The roles our experts play in this project are different from the roles domain experts play in other projects. It is imperative for the purposes of this chapter and this thesis as a whole to specify these roles more explicitly.

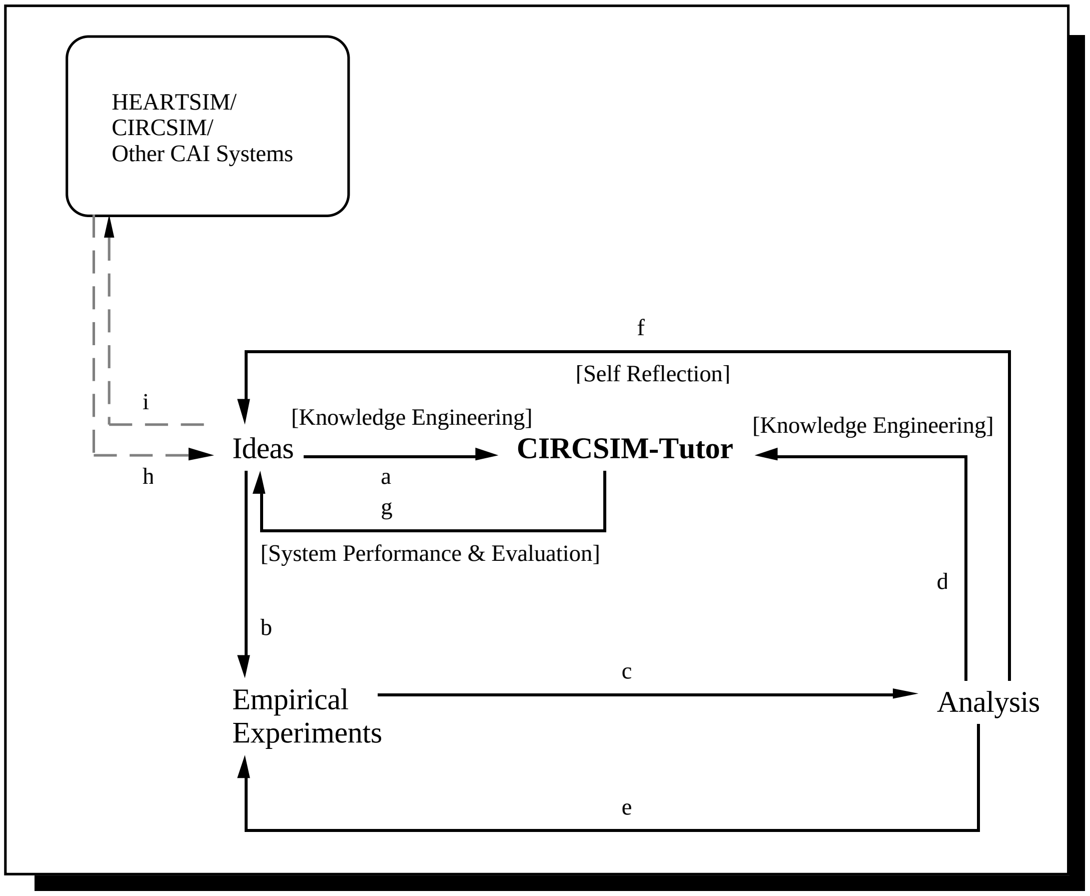


Figure 4.3 Influences on Expertise

These two domain experts are Professors of Physiology at Rush Medical College. Besides this they are also researchers on automated tutoring systems. They have developed several CAI systems in different medical domains (e.g., cardiovascular (Rovick & Michael, 1986), respiratory (Rovick & Michael, 1991), and acid/base regulation (Li et al., 1991)). They also have extensive tutoring experience. CIRCSIM-Tutor is the first intelligent tutoring system project with which they have been involved. They not only provide the domain expertise but also the tutoring expertise for this project (see Chapter V for a detailed account of these roles). Their extensive research experience has provided a unique form of expertise for the CIRCSIM-Tutor project. Here we briefly describe the influence for their unique expertise in the context of the CIRCSIM-Tutor project. One major reason for this analysis is to show that under our research framework the knowledge acquisition process cannot simply be a matter of “expertise transfer” or “knowledge extraction” but is instead an active process of model construction.

Figure 4.3 shows various activities that influence the expertise of our experts. Here arcs are labeled only for explanation purposes. This influence diagram is explained as follows. Here “ideas” represent the ideas of our experts regarding the functioning of CIRCSIM-Tutor. These ideas were originally influenced (see arc “h”) by their experience with HEARTSIM and CIRCSIM. Their experience with other CAI systems (e.g., GASP, ABASE) also has provide a unique source of ideas for CIRCSIM-Tutor. These ideas through the process of knowledge engineering (see arc “a”) helped to develop conceptual model for CIRCSIM-Tutor. These ideas also provide a basis for their empirical study of tutoring (see arc “b”). Empirical experiments (see Section 3.7.6) have provided a rich source of information for CIRCSIM-Tutor. This empirical data is analyzed (see arc “c”) for various purposes (for example, to understand the process of tutoring (Woo, 1991; Khuwaja et al., 1994b; Khuwaja et al., in preparation (a)), domain knowledge (Khuwaja et al., 1992; Khuwaja et al., in preparation (b)), student modeling (Shim et al., 1991; Hume et al., 1993), and language generation and understanding (Evens et al., 1993; Sanders et al., 1992; Seu et al., 1991)). This analysis in turn helped to develop the conceptual model for CIRCSIM-Tutor (see arc “d”). An analysis of these empirical studies has also provided a basis to perform further tutoring experiments (see arc “e”). This analysis of the behavior of our experts in empirical studies also provides a self reflecting experience (see arc “f”) for our experts/tutors that also shapes their behavior in future experiments and their ideas about the functioning of CIRCSIM-Tutor. The performance and evaluation studies of various versions of CIRCSIM-Tutor also provide a rich source of ideas (see arc “g”) for the current and the future versions of

CIRCSIM-Tutor. This whole experience with the CIRCSIM-Tutor project provides new insights to our experts (see arc “i”) that help them develop new CAI systems in various other domains of medical sciences.

It is clear from Figure 4.4 that the role our experts (AAR and JAM) play in the CIRCSIM-Tutor project is more dynamic than traditional experts in ITS or KBS projects. Here these experts collaborate with knowledge engineer(s) to perform various activities and help to construct a model of CIRCSIM-Tutor. I have also coordinated with these experts to develop a model of tutoring for CIRCSIM-Tutor (v.3).

4.6 Methods Used to Get Data about Domain and Pedagogical Expertise

This section describes my methods of getting expert information to develop a model of tutoring. Our experts are colleagues; they have worked together to develop automated tutors for more than ten years; they have a remarkable ability to resolve conflicts in their ideas and knowledge by discussion. The methods described below to collect raw form of expertise have been used at different phases of the ITS development framework described earlier.

4.6.1 Interview. Most information that is used to develop different models has been obtained by interviewing our experts. Interviewing is a skill that requires more than asking the right questions in the right way. It also implies adequate preparation, recording, and documentation.

Generally there are two types of interviews - focused and structured (Wielinga et al., 1987). I have used both types. The focused interview is a “normal” interview. It is the most widely employed technique for data collection (Wielinga et al., 1987). Here the interviewer asks questions on topics of conversation he has prepared in advance, and the interviewee provides the answer. The high level structure of this type of interview consists of three parts: an introduction explaining the purpose and structure of the interview, a series of questions focusing on a sequence of topics, and a closing summary possibly reviewing some of the topics discussed earlier in the interview. In this type of interview most valid answers are obtained by specific questions rather than general ones, preferably following a chronological sequence.

The second type of interview is the structured interview. In this type detailed insight is sought about some aspects of the domain. While a focused interview is comparable to normal conversation, a structured interview is much more like an interrogation. The major difference between these two types of interviews is that in the structured interview, the dialogue consists mainly of a large variety of questions put by the knowledge engineer to probe a few topics in depth, rather than a number of topics.

4.6.2 <u>Keyboard-To-Keyboard Tutoring: A Form of Live Human Tutoring</u>. This method captures the actual interaction used by the tutor while tutoring the student in the domain. Interviewing techniques by definition cannot be used to capture this type of data. Actually here the expert mimics as much as possible the function of the prospective system (Wielinga et al., 1987). I have not participated in collecting this type of data. Rather I was supplied with transcribed versions of tutoring sessions. See Section 3.7.6 for more details on this type of data collection.

T2A-28: I first look at the change in MAP in DR and I say , that change is going to inversely affect the neural variables that have the capacity of changing.
T2A-29: I am assuming we administered sufficient beta blockers so that we can not through the sympathetics affect either CC or HR.
T2A-30: So MAP is down.
T2A-31: TPR is not affected by the drug.
T2A-32: So TPR will be up.
T2A-33: Now I am just moving up from TPR to next neural variable, HR.
T2A-34: HR can not increase because of increase sympathetic activity but it can increase because of decrease parasympathetic activity.
T2A-35: So HR is up.
T2A-36: OK.
T2A-37: CC is the next neural variable.
T2A-38: CC can not increase because of increase sympathetic activity and that the dominating influence on CC.
T2A-39: So I need to put a zero there for CC.

Figure 4.4 A Sample Segment from a Think-Aloud Session

4.6.3 Concurrent Verbalization: A Method of Capturing Expert's Problem-Solving Behavior. In concurrent verbalization, the expert thinks aloud, while solving a problem. This method is particularly suited to elicit information about the control aspects (the task structure and the strategy) of the reasoning process. I have conducted a set of think-aloud sessions using this method to capture the problem-solving behavior of our domain experts. This section describes this experiment. The subject of this experiment is Dr. Allen Rovick - one of the experts participating in the CIRCSIM-Tutor project. Here I have acted as the experimenter and tape-recorded the verbalization of the subject. It was the experimenter who selected the domain problems from a large set of possible ones. Each problem presented the subject with a defined perturbation to the CV system (something that will cause a change in blood pressure) and required the subject to make qualitative predictions (increase/decrease/no change) about the responses of seven CV parameters to that perturbation. In all, six problems were solved by the subject. Besides verbalization, during problem-solving, the subject also used the predictions table (see Section 3.7.5) as a tool for recording his solutions for a problem. The tape recorded concurrent verbalization (Ericsson & Simon, 1993) from this experiment was then transcribed by hand. In this transcription process a numbering scheme was used to tag each sentence with an identification number. This numbering scheme has the following format.

session # (problem letter) - sentence #

For example, the number T2B-64 indicates that the sentence comes from think-aloud session 2, that it is the second problem of this session, and that it is the 64th sentence in this problem. Figure 4.4 shows a sample segment from a think-aloud session.

4.6.4 Retrospective Verbalization. In this method a subject verbalizes after the task is completed. I have used a form of this method in which our experts (AAR and JAM) thought aloud while reading a transcript that was captured in a recently conducted keyboard-to-keyboard session. I have tape recorded these sessions. This type of data is useful in capturing the reasoning behind various actions performed by the tutor in a tutoring session. A much better way to capture this type of data is to allow the tutor to think-aloud during a keyboard-to-keyboard session but since this has not been done (at least not very successfully) in our case, the retrospective verbalization is an alternative to it.

4.6.5 Group Sessions. I have participated in frequently held group sessions in which general and high level issues in the development of CIRCSIM-Tutor (v.3) were discussed. In these sessions several researchers at IIT under the supervision of Professor Martha Evens meet our domain and tutoring experts (AAR and JAM). Generally these sessions start with a very high level agenda but it is up to the participants to direct the flow of thoughts. Each researcher at IIT has been assigned a part of the research work needed to develop CIRCSIM-Tutor. These group sessions create an opportunity for all the members to get-together and familiarize themselves with the issues of other components of CIRCSIM-Tutor and view their work as part of this complete tutoring system.

Besides participating in these large group sessions, I have also arranged small group meetings. These were much more focused in their agenda. Four or fewer people participated in these meetings. Two of the participants, here, were our experts (AAR and JAM). These meetings are very effective in discussing common issues between two components of CIRCSIM-Tutor developed by separate researchers. In all of these large or small group sessions AAR and JAM not only play the role of domain experts but also act as researchers in automated tutors.

4.6.6 Review. Reviews are particularly relevant for repairing gaps in data. They are a way of assessing collected data. This method can also be used for assessing interpretations of data. Here it may take the form of discussing, for instance, the conceptual model with the expert. In an extreme form, the evaluation of a prototype by the expert can be viewed as a review technique as well.

I have used this method all through the knowledge acquisition phase of my research. During the development of the conceptual model of the domain expert (see Section 6.8) I have found the graphical representation of the concept map to be a very effective method of representing and discussing issues with our domain experts. In a sense it acted as a mediating representation in the development of the conceptual model of the domain expert (see Ford et al., 1993). Unlike some representations that are directly executable in the completed system (e.g., production rules embodied in an implementation-specific syntax), a mediating representation can not be directly executed, but is useful because it serves as a medium of communication between expert and knowledge engineer. In general, I have found graphical representations of knowledge as more effective than textual forms for the review process.

4.6.7 Study of Previous Research Work Performed in the Pre and Post CIRCSIM-Tutor Era. I have also studied the literature on other tutoring projects at Rush (e.g., HEARTSIM, CIRCSIM, earlier versions of CIRCSIM-Tutor). This study has also provided me with a wealth of knowledge and inspiration for the work on CIRCSIM-Tutor (v.3). Many design decisions from these early systems, after careful consideration, have been ported to CIRCSIM-Tutor (v.3). I have also played with most of these systems (with the exception of HEARTSIM) to get insight into their functional behavior. This has provided me with a source of many new ideas for CIRCSIM-Tutor (v.3).

4.7 Task Structure: A Way of Representing Problem-Solving Knowledge

In this thesis I will use a representation that captures a fixed strategy to perform a function in the domain. This representation is expressed using structured English. The ingredients for task structures are goal statements and control statements (Breuker, 1990). Goal statements are specified as action terms with or without objects, e.g., "get (prediction)." Goal statements may consist of structures of (sub)goal statements. Indentation is used to make the sub-goal structure explicit. Control statements are Pascal type statements, e.g., "Begin." It is possible to avoid many control statements by proper indentation. An example of a portion of a task structure is shown in Figure 4.5. This task structure can be used to solve the DR phase of a CV problem.

```
Solve (DR)
    Identify & Predict (Primary Variable)
        Identify & Predict (Procedural Variable)
        Propagate (Procedural Variable, Primary Variable)
    Identify & Predict (Regulated Variable)
        Propagate (Primary Variable, Regulated Variable)
    Predict (Rest of the prediction table variables)
        Propagate (Variable X, Variable Y)
```

Figure 4.5 A Task Structure to Solve the DR Phase of a CV Problem

CHAPTER V

TUTORING QUALITATIVE REASONING FOR THE FUNCTIONING OF THE BARORECEPTOR REFLEX: A COGNITIVE MODEL

5.1 Introduction

The purpose of this chapter is two-fold: to paint a very broad sketch of a model of tutoring - a major theme of this research - and to set the stage for the remaining chapters of this thesis. This model of tutoring is intended for CIRCSIM-Tutor (v.3). The multiple expert metaphor (see Section 2.2.1) provides one way of classifying the functional complexity of a model of tutoring. For the purposes of the research described in this thesis I have limited my model of tutoring to consider only the functionality of the domain and the didactic expert of this metaphor (see level three of Figure 4.2). The functional aspect of the didactic expert with which I am concerned deals only with the pedagogy decision making (see level four of Figure 4.2). From now on when I refer to my model of tutoring I will mean a model with only the above mentioned functionality.

Unlike the approaches taken by the earlier versions of CIRCSIM-Tutor, I have used an ITS development framework (see Chapter IV) to develop this model. Considering the methodology underlying this framework, the development of this model of tutoring has gone through four major phases - the linguistic, conceptual, system, and physical phases (see Section 4.1). Three major models (conceptual, system, and architecture) result out of this activity. These models are described in detail in Chapters VI, VII, and VIII, respectively. See Chapter IV for a description of the activities I carried out at the linguistic level. The sketch of my model, described in this chapter, is independent of the phases of the ITS development framework used in my research.

The major theme of this model of tutoring is that, in a problem-solving environment, it facilitates the student to integrate his/her knowledge into a coherent qualitative causal model of the domain and solve problems in the domain. The key feature of this model is that it uses multiple models of the domain in the process of facilitating knowledge integration.

5.2 Orientation and Limitations of the Model of Tutoring Used in the Earlier Versions of CIRCSIM-Tutor

HEARTSIM and CIRCSIM (see Chapter III) have greatly influenced the behavior of our tutors (AAR and JAM) in empirical studies of tutoring (see Section 4.5). CIRCSIM-Tutor (v.0) was developed before these experiments were conducted (see Figure 3.1). As a result the model of tutoring used in this system was wholly based on the previous experience of our tutors with their CAI systems and their vision about the functioning of CIRCSIM-Tutor. The developers of CIRCSIM-Tutor (v.0) did not pay much attention to the theoretical orientation of the model of tutoring used in this system. Also, in accordance with the tradition of the first generation of knowledge based systems (Ford & Bradshaw, 1993a), the development of CIRCSIM-Tutor (v.0) emphasized the product rather than its process. But as Ford & Bradshaw (1993a, p. 1) noted, "the most important product of a specific knowledge-acquisition project is not the knowledge-based system, but rather the insight gained in the process of articulating, structuring, and critically evaluating a model of some domain."

The capability of the domain model used in an ITS determines, in turn, the pedagogical capability of the system (Anderson, 1988). The domain model used in CIRCSIM-Tutor (v.2) is quite simple, but sufficient to solve a CV problem. During the development of this system it was assumed that this model is sufficient to enable the system to perform sophisticated tutoring. Soon it became obvious that this is not the case.

Instead of putting emphasis on the misconceptions of the student, as done in HEARTSIM and CIRCSIM, CIRCSIM-Tutor (v.2) puts more emphasis on the overlay modeling approach. This approach considers the student as possessing a subset of the knowledge of the tutor. The major goal of the tutor here is to find out the missing chunks of knowledge that are responsible for the student's sub-optimal behavior and develop lessons so that the student “fills-in” the gaps in his/her knowledge. The model of CIRCSIM-Tutor (v.2) used this theoretical orientation as its basis for tutoring. Not much emphasis was paid to the interaction between various experts in the multiple expert metaphor. For example, what are the implications of the domain expert for the behavior of the pedagogy expert. Also the pedagogy expertise here does not consider behaviors that are related to developing the tutoring environment and the tutoring protocol for the system (see Chapter III). CIRCSIM-Tutor (v.2), as mentioned earlier in Chapter III, inherited many characteristics from its earlier systems/versions. During the development of this system, a considerable amount of empirical data was also available (see Figure 3.1). But unfortunately its model is not heavily influenced by the analysis of these sessions. This system’s domain model is not radically different from CIRCSIM-Tutor (v.0). The pedagogical model in CIRCSIM-Tutor (v.2) is comparatively enhanced but again no theoretical foundation was established to view the functionality of the domain and the pedagogy expert in the light of each other. This version of CIRCSIM-Tutor rigidly followed the overlay philosophy to view the knowledge state of the student. As mentioned in Section 5.5.11, this is not the way our tutors perform in a human tutoring experiment. Again the tutoring protocol and the tutoring environment of this version of CIRCSIM-Tutor were not considered as part of the model of tutoring. See Chapter III for some more detail on these two versions of CIRCSIM-Tutor.

As a whole, one can conclude that the earlier versions of CIRCSIM-Tutor failed to consider some important aspects of the extensive data available from the human tutoring experiments conducted at Rush Medical College. Also, no comprehensive analysis of the complex tutoring behavior of our tutors was performed to synthesize it into a coherent model in light of its underlying theoretical orientation.

5.3 Scope of the Model of Tutoring For CIRCSIM-Tutor (v.3)

The model of tutoring developed for CIRCSIM-Tutor (v.3) is an attempt to overcome some of the shortcomings of the earlier versions of CIRCSIM-Tutor. Here more emphasis is placed on analyzing the behavior of our tutors in the keyboard-to-keyboard sessions. Also a systematic classification of behavior is performed in order to understand the nature of the underlying theoretical orientation of this model. Despite all this emphasis on observing the behavior of our tutors in the keyboard-to-keyboard sessions, no attempt has been made to develop a psychologically faithful model (or true simulation) of our tutors.

This research is based on the assumption that the tutoring performed by effective human tutors provides the best scenario on which the development of an effective model of tutoring for a machine tutor can be based (see Section 2.2.3). See Section 4.5 for a detailed account of the influences that have shaped this model. This section describes some of the major factors involved in this model. These factors define the scope of usage (or generality) of this model in a way. No explicit theoretical position has been considered by our experts while performing empirical experiments. The behavior of our tutors in these experiments is wholly based on their extensive experience as tutors/teachers, domain experts, course designers, and researchers on automated tutoring.

One of the fundamental assumptions behind this model of tutoring is that an effective tutor requires expertise in both the domain and in the process of tutoring. A successive breakdown of the roles of the tutor is shown in Figure 4.2. I have used these levels at different stages of this research to analyze the behavior of the tutor in our situation. As mentioned earlier, this model only considers the behaviors of the domain expert and the pedagogy expert (see Section 5.1). Traditionally these behaviors have been analyzed in isolation of each other. Here I have made an attempt to analyze these two roles both in isolation and in light of each other’s underlying theoretical orientation. I hope that this approach will bridge the gap between the various experts of the multiple expert metaphor (see Section 2.2.2). The domain expert provides the domain intelligence to the system so that an expert performance in the domain can be achieved. But it is the pedagogy expert that uses this intelligence to achieve the goals of the tutor. In our context human tutors perform a considerable amount of activity before a tutoring session starts. This activity mainly concentrates on creating the tutoring environment (and hence the tutoring protocol) in which the tutor and the student communicate. I classify this as *pre-session* activity of the tutor. The *in-session* activity is the activity performed by the tutor while a tutoring session is underway. This, in our case, is recorded in a keyboard-to-keyboard transcript (see Section 3.6.6). The model of tutoring I am describing considers both pre-session and in-session activities of the tutor. These activities are part of the behavior carried out by the pedagogy expert. During tutoring, the pedagogy expert performs three major decisions: What to teach, When to teach, and How to teach. These activities are performed in order to achieve the goals of the system.

Next I will describe various variables that influence my model of tutoring. These variables limit the generality of this model and are inherited from the earlier versions of CIRCSIM-Tutor (see Section 3.6). Here I review these in the context of CIRCSIM-Tutor (v.3)

5.3.1 Style and Method of Tutoring. Like CIRCSIM-Tutor (v.2), CIRCSIM-Tutor (v.3) uses a Socratic style (Wenger, 1987) to communicate with the student. In this style the tutor frequently asks questions and responds to the student's queries. In this system the default method used to tutor is based on discovery, i.e., here the tutor tries to help the student learn by discovering knowledge of the subject domain by him/herself. If this is not successful then the tutor switches mode and teaches the material in an expository fashion. The discovery method naturally suits the Socratic style because in the Socratic style the tutor lead the student by constantly asking questions so that he/she comes closer to discovering the targeted knowledge of the domain. Involving the student in this active inquiry process will make his/her knowledge extensible, we hope (Wenger, 1987). In CIRCSIM-Tutor (v.3) it is the pedagogy expert that determines both the style and method of tutoring.

5.3.2 **Tutoring Domain.** Just as in earlier versions of CIRCSIM-Tutor, the knowledge domain of CIRCSIM-Tutor (v.3) is cardiovascular physiology, specifically the baroreceptor reflex, which maintains a more or less constant blood pressure using a negative feedback mechanism. While tutoring in this domain the system forces the student to concentrate only on the causal nature of the working of the baroreceptor reflex. See Section 3.6.1 for details about the nature of this domain. In my model of tutoring the domain expertise is provided by the domain expert.

5.3.3 **Learning Context.** CIRCSIM-Tutor (v.3), like earlier versions, is intended to be used by first year medical students. This system will be an integral part of a physiology course at Rush Medical College. It is assumed that the student using the system will have acquired the necessary knowledge through attendance at lectures, reading the textbook, and participating in other scheduled activities in the physiology course. See Section 3.6.2 for more detail about this learning context.

5.3.4 **Teaching Goals.** The teaching goals of CIRCSIM-Tutor (v.3) emphasize qualitative reasoning to perform problem-solving in the domain. Very broadly the goals of this system are described as follows. The students, using the system, acquire a qualitative causal model of the cardiovascular system, and they learn a problem-solving method that enables them to solve any problem in the domain. Again it is the pedagogy expert that is responsible for trying to achieve the goals of the system as much as possible while interacting with the student.

5.3.5 **Nature of the Tutoring Task.** The learning exercise in CIRCSIM-Tutor (v.3) is based on problem-solving, i.e., the system provides an opportunity to the student to develop problem-solving skill in the domain using qualitative reasoning. The nature of the task that the student is required to perform in the domain can be classified as "prediction" (Clancey, 1985; Rovick & Michael, 1992). In this task the student predicts the qualitative responses for CV parameters involved in the baroreceptor reflex. Within the system it is the domain expert that performs this task to solve each problem correctly.

This solution (and the associated domain knowledge) is then communicated to the student by the pedagogy expert. Clancey (1985) has classified the task that the pedagogy expert performs as “instruction.” The way the pedagogy expert communicates with the student depends upon many factors, for example, the style of tutoring, the method of tutoring, the kind of tutoring. In the keyboard-to-keyboard session our tutors concentrate on remediating underlying misconceptions of the student who has made errors while predicting the responses for parameters of the CV system (see Section 5.5.11 for more details). This type of tutoring has also been selected for CIRCSIM-Tutor (v.3). Although earlier versions of CIRCSIM-Tutor also have this goal, a different approach has been adopted for CIRCSIM-Tutor (v.3).

5.3.6 Learning Environment. CIRCSIM-Tutor (v.3) promotes learning by doing. This is our basic philosophy of education. This system offers to the student an structured environment in which he/she can use his/her knowledge of physiology to solve problems. The protocol adopted by this system to interact with the student is based on an extensive empirical study done by our tutors at Rush Medical College. An analysis of the tutoring environment and protocol used in this empirical study has been described in detail in Section 5.5.2. CIRCSIM-Tutor (v.3) promotes the learning by doing philosophy more than the earlier versions. This is achieved by adopting a protocol that neatly integrates the uninterrupted activity required for learning by doing with the Socratic style that has been designed to remediate misconceptions using the discovery method of learning.

5.4 Evaluation of Keyboard-To-Keyboard Experiments: How Effective Our Tutors Are?

The true effectiveness of my model can only be tested once CIRCSIM-Tutor (v.3) is ready to be used by the medical students. But before we reach that stage, one of my prime concerns was to find out how effective our human tutors are in keyboard-to-keyboard experiments. No such evaluation study had been performed before although Rovick & Michael (1992) had performed an evaluation study of CIRCSIM. There are three main reasons for my concern. (1) As mentioned before, my model mimics as much as possible the behavior of our tutors in these experiments, (2) the tutoring environment created by the keyboard-to-keyboard experiments is designed by our tutors to closely simulate the situation to be encountered in CIRCSIM-Tutor, and (3) a positive result of an evaluation study would give me confidence in the data on which my model is based. This would also act as a major motivational factor for me to continue my effort in building this model.

In the spring of 1993, convinced by my arguments, AAR and JAM planned an experiment that included pre tests and post tests to evaluate their teaching effectiveness. In this section I briefly state the results of this experiment. Appendix A contains the protocol used for the experimental and the control groups as well as the results of this experiment compiled by Dr. Allen Rovick.

In this evaluation study two groups of first year medical students at Rush Medical College were formed. The first group, called the control group, was given reading material from a physiology book. Once the subjects in this group finished reading this material for an specified period of time, a CV problem (P) was given to solve. The other group, called the experimental group, was tutored by AAR and JAM using the keyboard-to-keyboard setting. Pre and post tests were administered for both of the groups. These tests were designed to measure the student's knowledge of a set of important CV relationships and their ability to solve a CV problem. Two measures were used to judge the student's ability to solve problems: the number of incorrect predictions and the number of relationship errors in the solution of the problem. Appendix A contain tables showing the results of this experiment for both groups. These results indicate that the experimental group learned a substantial amount of problem-solving.

From this evaluation it can be concluded that the data used to build my model of tutoring is obtained from quite effective human tutoring behavior. This does not automatically make my model effective but at least it gives me the confidence to continue my effort to base this model on the effective tutoring behavior of our tutors.

5.5 Nature of Expertise in Keyboard-To-Keyboard Sessions: A Cognitive Model

This section describes the model of tutoring that I have developed for CIRCSIM-Tutor (v.3). Here I will sketch a very broad picture of this model by classifying its major components. A detailed description of this cognitive model in the form of conceptual model, system model, and architecture is given in Chapter VI, VII, VIII, respectively.

Figure 5.1 schematically shows the two major experts whose behavior is considered for this model of tutoring. Here my hypothesis is that it is the domain expert that provides domain intelligence to the pedagogy expert, which in turn provides the tutoring expertise and communicates with the student. Because of the nature of the activity of our tutors in the keyboard-to-keyboard sessions, from the transcripts of these sessions one can only observe the composite behaviors of these two experts. It would be extremely advantageous if we could capture the behavior of one of these experts in isolation and then take it as a reference to analyze the behavior of the second expert from the keyboard-to-keyboard session. Fortunately in our research framework our tutors play multiple roles (see section 4.5). The two roles that I am concerned with require them to act as expert in the domain and in the process of tutoring as well. So here it is possible to capture the behavior of the domain expert (see Figure 5.1) by letting our tutors verbalize while solving the kind of problems they give to the student in a keyboard-to-keyboard session. This is exactly what I have done to capture the problem-solving method of our tutors. As mentioned in Section 4.6.3, this method of knowledge acquisition is particularly suited to eliciting information about the control aspects of the reasoning process. Section 5.5.1 describes in detail our tutor's method of solving a CV problem. Chapter VI describes in detail other types of domain knowledge (e.g., structural, causal) used by our tutors. These types, in fact, form different models of the domain that they use while tutoring the student.

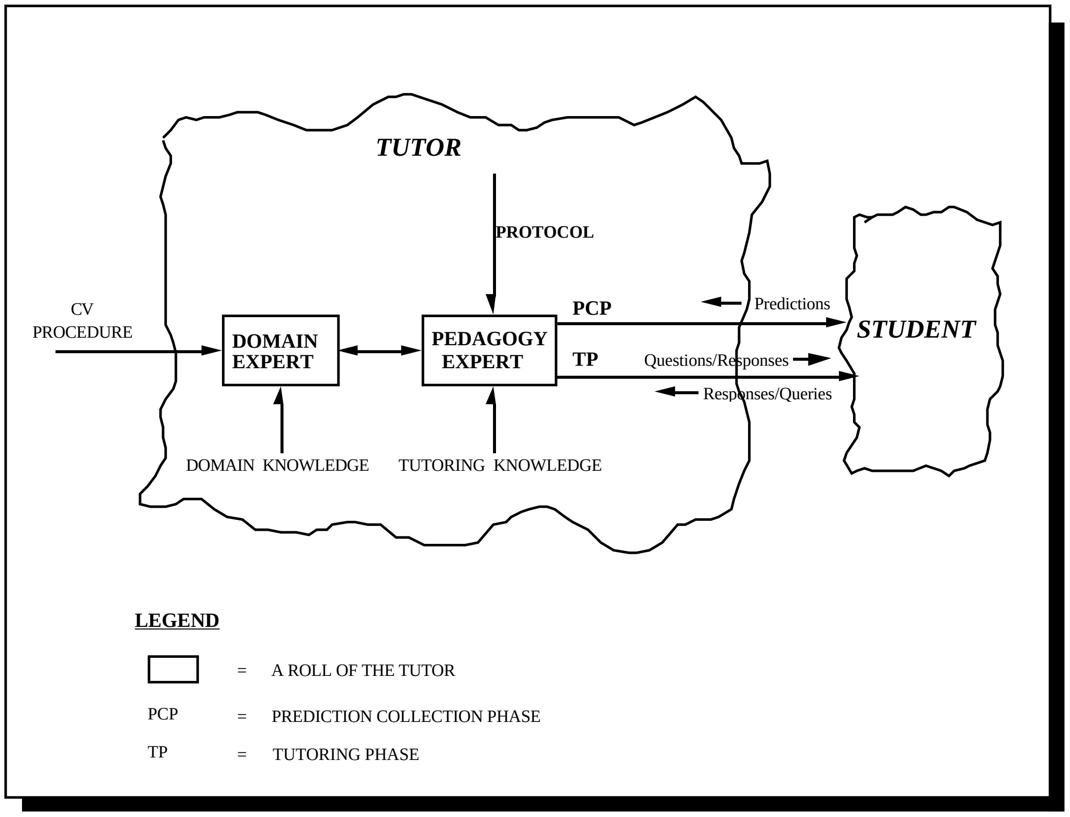


Figure 5.1 Schematic View of a Cognitive Model of Tutoring Used in Keyboard-To-Keyboard Sessions

Once a model for the domain expert was completed I turned to modeling the behavior of the pedagogy expert. Here two issues are worth noting. (1) I have not only modeled the behavior of these two experts in isolation but also tried to analyze the interaction and influences of these experts on each other, and (2) I have broadly classified the behavior of the pedagogy expert as pre-session and in-session. Pre-session behaviors are frequently ignored by the developers of ITS. But in my view these are essential in developing the tutoring environment in which the tutor and the student interact. I think before a tutoring session starts a human tutor makes many decisions that are in many ways similar to the decisions made by the instructional system designers. Hence in order to make a serious effort to bring an ITS in a real educational environment, it is imperative that we also pay attention to the activities of the tutor that he/she performs before entering into a tutoring session. In the following section I will extensively analyze the pre-session behaviors of our tutors. In order to capture a high level behavior of our tutors during a keyboard-to-keyboard session, I will analyze, in the following section, a keyboard-to-keyboard transcript.

5.5.1 Problem-Solving Behavior of the Domain Expert. Section 4.6.3 describes the method used to capture and transcribe a set of think-aloud sessions. The major purpose of these sessions was to capture the problem-solving behavior of the domain expert role of our tutors. Such behavior was already captured by the developers of earlier versions of CIRCSIM-Tutor (Kim, 1989; Zhang et al., 1990). The dominant method used by these developers was interviewing our tutors. Concurrent verbalization, compared to interviewing techniques, is regarded as a much better method to capture this type of domain knowledge. I have also used the transcripts obtained by these sessions to analyze other components of my model of tutoring (see Chapter VI). A task structure representing the problem-solving method extracted from these think-aloud sessions is shown in Figure 5.2. The subject (AAR) in these sessions applied this method to various knowledge structures to solve CV problems. Chapter VI gives a detailed account of an approximation of these knowledge structures used by our tutors.

The task structure of Figure 5.2 partitions the problem-solving behavior of the subject into three stages - DR, RR, and SS (see Section 3.6.1). In each stage the three most common operations performed by the subject are: identify, predict, and propagate. The first operation identifies a physiology variable according to the criterion specified as its argument. For example, the operation "identify(primary variable)" means identify the physiology variable that is first affected by the current perturbation and is listed in the prediction table. Other possible arguments of this operation and their definitions are as follows. Procedural variable - this is the variable in the concept map that is first affected by the perturbation. Regulated variable - this is the variable that the baroreceptor reflex system monitors and holds constant (i.e., MAP). Control variables - these are the variables that are under neural control.

```
Solve (cv problem)

    Solve (DR)
        Identify & Predict (Primary Variable)
            Identify & Predict (Procedural Variable)
            Propagate (Procedural Variable, Primary Variable)
        Identify & Predict (Regulated Variable)
            Propagate (Primary Variable, Regulated Variable)
        Predict (Rest of the prediction table variables)
            Propagate (Variable X, Variable Y)

    Solve (RR)
        Identify & Predict (Controlled Variables)
            Propagate (Regulated Variable (DR), Controlled Variables)
        Identify & Predict (Regulated Variable)
            Propagate (Controlled Variables, Regulated Variable)
        Predict (Rest of the prediction table variables)
            Propagate (Variable X, Variable Y)

    Solve (SS)
        Solve Causally  |
            Identify & Predict (Regulated & Controlled Variables)
                {Identify & Predict (Regulated Variable)
                Identify & Predict (Controlled Variables)} |
                {Identify & Predict (Controlled Variables)
                Identify & Predict (Regulated Variable)}
            Predict (Rest of the prediction table variables)
                Propagate (Variable X, Variable Y)
        Solve Algebraically
            Algebraic Addition (Variable X (DR), Variable X (RR))
```

Figure 5.2 The Task Structure Used by the Domain Expert

The second most common operation used in the task structure of Figure 5.2 is “predict.” In this operation the subject predicts a qualitative value (increase, decrease, or no change) for a physiology variable specified as its argument. The third most common operation used is “propagate.” This is a very complex operation compared to the other two operations used in this task structure. While using this operation the subject mentally propagates the causal influence from one physiology variable to another. In other words, this operation allows the subject to predict the value of the next causally relevant physiology variable in a stage of a CV problem.

In one of the problems that the subject solved, the arterial resistance (RA) was reduced to 50% of the normal. In this case the subject first started to solve DR (see Figure 5.2). Here the first goal created by the subject was to identify and predict the qualitative value of the primary variable. In order to do that, the subject first identified and predicted the procedural variable that in the current case was explicitly given as part of the problem description (i.e., RA decrease or RA -). Next the subject propagated this influence in the CV system and predicted TPR, which is the primary variable in the given case (refer Figure 3.2). Next the subject created a goal of predicting the qualitative value of the regulated variable. In order to do that the subject propagated the causal influence from TPR to MAP. This operation yields MAP -. The next operation in Figure 5.2 says to predict the rest of the prediction table variables in a causal sequence. Starting from MAP the subject next predicted the following variables: SV +, CO +, RAP -. Since the non-primary neurally controlled variables do not change in the DR phase, the propagate operation assigns 0 to these remaining variables of the predictions table. Next the subject solved the RR phase in a similar manner.

The SS phase of Figure 5.2 is more interesting. Here the subject had a choice of using either the causal or the algebraic method to predict CV variables (in Figure 5.2 "|" represents an OR). In the causal method the subject had a choice of starting from the controlled variables (CC, HR, and TPR) or the regulated variable (MAP) and moving backwards to predict the rest of the prediction table variables. On the contrary in the algebraic method the subject used a truth table to predict SS values for the prediction table variables. This truth table is shown in Figure 5.3. This table shows how the DR and the RR values for a CV parameter determine the SS value for that parameter. As is clear from this table, most of the time, the SS value is the same as the DR value except when the DR value is zero. In that case the SS value is the same as the RR value.

DR	RR	SS
+	+	+
0	+	+
+	0	+
+	-	+ OR -
-	+	+ OR -
-	-	-
0	-	-
-	0	-
0	0	0

Figure 5.3 Truth Table to Solve the SS Phase of a CV Problem

5.5.2 <u>Tutoring Protocol: Pre-Session Behavior of the Tutor</u>. Tutoring a problem-solving task effectively in virtually any structured domain requires the tutor to create a problem-solving *environment* so that he/she can communicate effectively in the domain. The creation of a problem-solving environment requires the development of a set of rules that govern the interaction of the tutor and the student in the environment. This set of rules does not constrain the behavior of the tutor or the student in each step in the problem-solving process but rather emphasizes higher level constraints of the problem-solving task in the domain and a generic way of proceeding in it. We call this set of rules the *tutoring protocol.* In other words, the tutoring protocol is developed by the tutor to allow him/her to exercise control over the tutoring environment. The tutoring protocol is a very high level plan of the tutor, which is developed *before* the actual interactive communication with the student starts. But it is flexible and carefully thought out to ensure that optimal knowledge communication takes place during the tutoring session.

In the following sections we will use keyboard-to-keyboard transcripts to analyze tutoring protocols used by our tutors. Three different tutoring protocols have been used in 47 tutoring sessions (see Figure 3.6). Here we will classify different behaviors of our tutors in developing these protocols. A comparison of the characteristics of these protocols has yielded many insights that have also clarified the theoretical orientation of the approach of our tutors.

5.5.3 Problem-Solving Environment of Our Keyboard-To-Keyboard Sessions. When tutoring is carried out, whether face-to-face or employing a keyboard-to-keyboard communications channel, there are rules that define and constrain the interaction between the tutor and the student. These rules are usually implicitly understood by both parties to the interaction; some rules are generic to any tutoring interaction, while other rules are specific to the particular tutoring that is occurring. For example, the rules that govern the conduct of both parties in a session where a student in academic difficulty hires a tutor are not entirely the same as the rules that govern a session that is conducted as part of an experimental study. In the same way, differences in the knowledge domain being tutored or in the kind of problems being solved will result in different tutoring rules.

The rules being considered here govern such aspects of the tutoring as how the communications medium is to be used (making entries at the keyboard, turn taking, how to interrupt, etc.), how the problem is defined for the student, what kind of problem solving behavior is expected of the student, what constitutes success in problem solving, how much time is available for the tutoring session, etc. We will describe these constraints here very broadly, in the order in which they have been used by the tutor. In the next section we will classify these constraints/rules in a way that makes explicit various activities our tutor performed in a keyboard-to-keyboard session.

In our situation it is clear that the tutoring session is controlled by the tutor/experimenter. The student is given a notebook containing seven pages of information/instructions, and the use of the keyboard system is explained. The interaction then begins with the tutor obtaining the student's name and social security number. The student is then asked to read the pages in the notebook in sequence and ask questions, if any, about the material on each page. Page 1 describes the general procedure for the tutoring session that will be conducted. The second page provides the student with a brief description of the physiological system to be thought about. The problem, the perturbation that causes an alteration in blood pressure, is then defined on page 3. The fourth page defines the first of the periods (DR) about which predictions must be made, and the fifth page explains how the student is indicate those predictions using the keyboard. At this point, the tutoring session actually begins with the student making predictions about the DR phase. When the tutoring about the DR phase is completed the student is asked to read page 6, which defines the second (RR) phase of the response and then makes predictions and is tutored about this aspect of the response. Finally, the seventh and last page defines the last phase of the response, SS. The preceding description of the tutoring session applies to all 47 that have been conducted.

5.5.4 A Classification of Rules of a Tutoring Protocol. In this section we will classify rules used in the keyboard-to-keyboard session. Here the terms rules and instruction are used interchangeably.

5.5.4.1 Rules for the Student. These rules are meant for the students to follow and are broadly classified into the following categories.

I) Rules for the Use of the Communication Medium. Communication in a keyboard-to-keyboard session is achieved via two PC's connected via telephone lines and controlled by CDS (Li et al., 1992). The student should be told the operations of CDS for effective communication with the tutor. A set of rules accomplishing this goal has been developed by our tutors. Some of these rules are displayed on the screen of the student's side PC all the time (i.e., integrated in the design of the CDS screen, for example, in the "INSTRUCTION" window on the screens of CDS) and others are compiled as an instruction document for the student to read before the actual tutoring experiment begins (examples of these instructions are: "When you have finished reading this material, type OK and <RETURN> if you understand it. Or, if you have a question about the material, type your question. Then press <RETURN>." "If you think that the value will go up enter I, if down enter D, if unchanged enter 0"). Since the tutor is monitoring the proper use of these instructions by the student, sometimes the tutor reminds the student about these rules during the tutoring experiment (e.g., at K1-tu-25-1 the tutor instructs the student: "Remember to finish each entry with an xxx. Are you finished?").

II) Rules for Acting in a Tutoring Experiment. This set of rules determines the nature and constraints of the tutoring exercise and his/her responsibilities to learn as much as possible from it. These sets of rules in a keyboard-to-keyboard session are given to the student in a form of a formal document to read. If necessary, the tutor also reminds the student of these rules during the experiment. A possible classification of these rules follows.

A) Rules Which Convey General Description of the Exercise. These instructions inform the student about the nature of the exercise (e.g., "You are going to be given a problem to solve."; "The problem consists of a description of a patient ....") and tasks he/she ought to perform (e.g., "You will be asked to predict the effect of the patient's problem on seven of his cardiovascular variables"; "Please look over the list of variables.").

B) Problem Description. This part of the tutoring protocol describes the current problem at hand to the student (e.g., "Ms BV entered the hospital for elective surgery. The night before the surgery was to take place, she was accidentally rapidly transfused with 1 liter of compatible blood.”).

C) Rules Which Determine the Way the Problem Solution Should Be Approached. One of the functions of these rules is to describe the tools that the student should use to approach the solution, e.g., “you have been given a worksheet (the prediction table). The worksheet has a list of the cardiovascular variables you will be making predictions about.” Another function of these rules is to outline a general approach for the solution of the exercise at hand, e.g., “I will be asking you first to indicate which variable you want to predict.” “I would like you to first predict the DIRECT RESPONSE (DR) of the seven variables to the transfusion.” “Now that you have completed your predictions of the DIRECT RESPONSE to the transfusion, please predict the REFLEX RESPONSE that will occur.” “Finally, predict how the value of each variable will have changed when the patient’s cardiovascular system comes to a new STEADY STATE, i.e., show the change from the period before the transfusion to the STEADY STATE.” “Make your predictions in an order that is consistent with the causal relationship between the variables.” “The STEADY STATE change for each variable is the SUM of its DIRECT and REFLEX changes.” “You can keep track of your predictions by entering them in the prediction table.”). These rules also make explicit the assumptions/constraints of the current exercise (e.g., “In arriving at your predictions, assume that the right ventricle, the pulmonary circulation and the left atrium act as a single passive structure and automatically pass the effect of a change in the right atrial pressure (RAP) over to the left ventricle (LV). A change in RAP would thus directly affect LV function.” “By DR I mean what happens to the seven variables in the short period after the transfusion but before reflex changes can occur.” “... then only the direct physical consequences of the transfusion would be seen.” “The RR should show how the baroceptor reflex changes the value of the variables from the values produced by the DIRECT RESPONSE.”

5.5.4.2 <u>Rules for the Tutor</u>. This set of rules allows the tutor to achieve control over the tutoring experiment and presents a general way of handling the tutoring process. Some of the rules are only meant for the tutor and others for how, what and when to give instructions to the student. We have broadly classified these rules into the following categories.

I) <u>Rules that Collect Demographic Information</u>. The tutor uses these instructions to obtain the student's personal information (e.g., "Please type your name and social security number." "Have you used the teaching program CIRCSIM or HEARTSIM (Plato)?").

II) <u>Time Constraints</u>. Time constraints force the tutor to finish a tutoring experiment in a certain time period. Because of time constraints the tutor needs to continuously monitor the progress and pace of the experiment and adjust accordingly (e.g., "Arrange for 1.5 hour session ...").

III) <u>Rules for Using the Communication Channel at Hand</u>. The tutor needs to know how to communicate with the student. For example, in the keyboard-to-keyboard sessions the tutor must be aware how to use CDS. Along with this knowledge the tutor also needs to know what instructions, regarding the use of communication channel, the student should be reminded of and when (e.g., "If you make a mistake or want to change an entry, you can erase it by backspacing." "When you have finished please press <RETURN>."). Instructions in this category achieve these goals.

IV) <u>Rules that Determine When and in What Order the Student Should Be Exposed to "Student's Set of Instructions"</u>. In our tutoring experiment the tutor is in control of the experiment. One of the responsibilities of our tutors, while participating in the experiment, is to control the flow of instructions to the student. This includes the student's set of instructions, which are compiled in a document form and are in the student's possession, and on-line instruction from the tutor. These rules provide guidelines for the tutor to accomplish these goals (e.g., "Have student read pages 1 through 4, one at a time." "Wait for OK or question after each page." "Tutoring session begins after student OKs page 4.").

V) <u>General Instructions for How to Approach Tutoring</u>. These instructions form the heart of a tutoring protocol. But these are kept very general in order not to force the tutor to give up his natural style and approach for tutoring. These instructions also sketch a very high level plan for the session. The actual interaction during tutoring is very student dependent and opportunistic (from the tutor's point of view). Because of the general nature of these instructions, tutors in some of our 47 tutoring sessions have violated some aspects of these instructions. But, to the best of our knowledge, these violations have not created serious flaws in our experiments. Examples of these instructions are: "Primary variable must be correctly identified." "... student then reads SS definition and after any question tutoring (in SS) begins." "After the correct prediction for primary variable allow student to predict for remaining variables and then start tutoring."

5.5.5 <u>Protocols of the Keyboard-To-Keyboard Sessions</u>. Our tutors have conducted 47 keyboard-to-keyboard sessions during the last four years (see Figure 3.1). Over this span of time three different tutoring protocols, which we call Protocol 1, Protocol 2, and Protocol 3, have been used. These protocols are alike in many ways, but their differences have noticeably different effects on the tutoring process.

In Section 5.5.4 we have described a classification of the rules that guide the tutor and the student behavior in a tutoring session. All three protocols are essentially identical to each other. Only some rules for the use of the CDS have been changed in the most recent protocol (i.e., in Protocol 3), because a new version of CDS was used in this tutoring experiment. The section that varies most from one session to another was the "general instruction for how to approach tutoring" (see Section 5.5.4). From now on we will discuss the three protocols from the point of view of these general instructions.

In this section we will use an analysis of the keyboard-to-keyboard sessions to describe the details of three protocols and their effect on the process of tutoring. One of the tutoring actions that we will closely observe is the frequency of tutorial intervention in each protocol. Here I have also used other knowledge acquisition techniques (e.g., interviewing tutors) to add details to this analysis.

A detailed investigation of these three protocols requires a framework on which the analysis can be based. Here we assume that the problem-solving behavior of the tutor functioning as a domain expert will affect the development of the tutoring protocol. In order to investigate this hypothesis I have conducted a series of think-aloud sessions to develop a description of the problem-solving behavior of a domain expert. Section 5.5.1 describes this behavior in detail. The resulting description of the problem-solving behavior of the domain expert is used, in the following sections, as a central component of the framework for the analysis of tutoring protocol.

Another aspect of our analytical framework arises out of the fact that the tutoring interaction that we are studying always arises from a prediction made by the student about one of the cardiovascular variables listed in the prediction table. The tutoring session is always divided into two phases: the prediction(s) collection phases (PCP) and the tutoring phases (TP) (see Figure 5.1). Variations in the ordering and arrangement of these two phases are the major differences between the three tutoring protocols; an important consequence of these differences is the timing of the availability of information about the student’s cognitive state when the tutoring interaction begins.

5.5.6 <u>Three Tutoring Protocols: Common Characteristics</u>. In this section we will describe the common characteristics of the three tutoring protocols that are used in our keyboard-to-keyboard tutoring sessions. In the following sections we will describe each protocol, independently, in detail.

All three of our tutoring protocols (Protocol 1, Protocol 2, and Protocol 3) break the tutoring process into three stages (i.e., DR, RR, and SS) just as in the task structure of

Figure 5.2. In each stage, two major (high level) operations are performed by the tutor. The first of these operations is "collect-student's-prediction." Here the tutor collects the student's prediction for a physiology variable(s) in the predictions table. The second operation is "tutor." These operations make up a part of the prediction collection and tutoring phases. The chief difference between the protocols lies in the relationship between these two phases.

All tutoring protocols for DR require that the primary variable be predicted first and correctly, and the students are tutored until they have this correct. In DR the two groups of variables, neural and "physical" are tutored differently. Tutoring in RR is driven by the two stereotypical response patterns that are possible, compensating for increased or decreased MAP-DR. All three protocols explicitly hint about an algebraic method of predicting in SS and most of the time, when the student used this algebraic method the tutor reinforced it. Only in a few sessions, was tutoring about the SS phase approached by the student and the tutor in a causal way.

5.5.7 **Tutoring Protocol 1**. This protocol was used in the first set (K1-K8; see Figure 3.6) of transcripts and no formal specification was developed for it, at the time it was used. An analysis presented in this section is solely based upon analysis of keyboard-to-keyboard sessions and interviewing and debriefing tutors.

The task structure for this protocol is shown in Figure 5.4. The prediction collection and tutoring phases overlap greatly in this protocol. As a consequence the tutor provides *immediate* feedback for each student's prediction/response. If the student's prediction is correct then the tutor gives a positive acknowledgment and if needed provides additional (relevant) knowledge at that point. If the student's prediction is wrong the tutor tries to remedy the problem that caused the wrong prediction at that time before letting the student predict the remaining variables. The most obvious global plan in each stage (DR, RR, and SS) is to follow operations in the respective stages of the task structure of Figure 5.2. But the tutor seems not to be forcing this plan, hence, the sequence in which variables are predicted and discussed is determined by the student solving the problem. Since the tutor only gradually obtains the student's predictions, there are at least two ways a tutor could behave in such an environment. He either forces his problem-solving method by ignoring all of the student's out-of-sequence predictions or explores the student's response at each point in problem-solving. Our tutors seem to use this latter strategy and push their problem-solving sequence only when the student is unable to progress in the process of problem-solving.

```
Tutor (cv problem)

    Tutor (DR)
        Collect & Tutor (Primary Variable)
            Collect-student's-prediction (Primary Variable)
            Tutor (Primary Variable)
        Collect & Tutor (Rest of the prediction table variables)
            Collect-student's-prediction (Variable X)
            Tutor (Variable X)

    Tutor (RR)
        Collect & Tutor (Prediction table variables)
            Collect-student's-prediction (Variable X)
            Tutor (Variable X)

    Tutor (SS)
        Collect & Tutor (Prediction table variables)
            Collect-student's-prediction (Variable X)
            Tutor (Variable X)
```

Figure 5.4 The Task Structure of Tutoring Protocol 1

5.5.8 **<u>Tutoring Protocol 2</u>.** This is the second tutoring protocol used in our tutoring experiments (K9-K28; see Figure 3.6). This time a formal specification was developed for this protocol. The task structure for this protocol is shown in Figure 5.5. In comparison with the first protocol, the PCP and the TP phases overlap less.

The prediction collection phase is quite complex compared to the first protocol. Besides collecting predictions for the prediction table variables the tutor monitors and provides help (via a hinting process) for *sequence* errors. A sequence error occurs when a prediction is made at the wrong point compared to the order that the domain expert (using his problem-solving method) uses, i.e., when the student predicted out of a logical causal sequence. But this protocol does not allow the tutor to remedy the actual cause of an error, just to provide generic hints so that the student predicts variables in the same sequence as in the task structure of Figure 5.2. The hinting process does not depend upon the problem or the type of student. It reminds the student that a sequence violation has taken place and provides a general heuristic for dealing with this situation, “In order to predict a parameter you have to have predicted its determinants.”

```
Solve (cv problem)

    Solve (DR)
        Collect & Tutor (Primary Variable)
            Collect-student’s-prediction (Primary Variable)
            Tutor (Primary Variable)
        Collect (Rest of the prediction table variables)
            Collect-student’s-prediction (Variable X)
                If “sequence” violation
                Then give a hint (but do not tutor)
        Tutor (Rest of the prediction table variables)

    Solve (RR)
        Collect (Prediction table variables)
            Collect-student’s-prediction (Variable X)
                If “sequence” violation
                Then give a generic hint (but do not tutor)
        Tutor (Prediction table variables)

    Solve (SS)
        Collect-student’s-prediction (Prediction table variables)
        Tutor (Prediction table variables)
```

Figure 5.5 The Task Structure of the Tutoring Protocol 2

When tutoring begins, a complete column of predictions (either in DR, RR, or SS) is available to the tutor. Thus the tutor is much better informed about the student’s knowledge of the domain, and is in a position to remedy the fundamental problems of the student using patterns of errors (instead of individual errors). In the RR stage this protocol follows exactly the same pattern as in DR. The SS stage of this protocol is the same as the RR stage except no sequence checking is done because the tutors have found no definitive order for handling the SS column, as shown in Figure 5.2.

5.5.9 Tutoring Protocol 3. This is the third protocol used in our human tutoring experiments (K20-K48; see Figure 3.6). This protocol was carefully thought out and formalized by our tutors before using it in a tutoring situation. The high level task structure for this protocol is shown in Figure 5.6. The amount of overlap between the PCP and the TP phases is even smaller than in Protocol 2. The issue that is responsible for this small overlap is primary variable tutoring.

The prediction collection phase, in all stages, is relatively simple compared to Protocol 2. In Protocol 3 the tutor does not provide any help in this phase and allows the student to predict variables in any order. During the DR stage as soon as the primary variable issue is raised the tutor is just a silent viewer. Until the DR column is complete, his job is to provide neutral commands like: "What variable do you want to predict next?" to obtain the next prediction table variable and "OK, how will it change?" to get the prediction.

Before tutoring, the tutor has the complete student's solution for a stage, much more information than in the first protocol. Also the tutor never interrupts the student during the PCP (unlike the second protocol), hence the record of the sequence of predictions and their values represents a true record of the student's performance. This (natural) information helps the tutor to form an initial model of the student. The tutoring phase is again, as in the second protocol, error pattern driven and lets the tutor, right away, start to develop a hypothesis for the underlying cause of the student's problem. The sequence of remedying the student's prediction error is not based upon the sequence of predictions from the task structure of Figure 5.2 but rather the tutor attacks fundamental student problems. The importance of the problems determines the sequence in which errors are ordered and tutored.

```
Solve (cv problem)

    Solve (DR)
        Collect & Tutor (Primary Variable)
            Collect-student's-prediction (Primary Variable)
            Tutor (Primary Variable)
        Collect-student's-prediction (Rest of the prediction table variables)
        Tutor (Rest of the prediction table variables)

    Solve (RR)
        Collect-student's-prediction (Prediction table variables)
        Tutor (Prediction table variables)

    Solve (SS)
        Collect-student's-prediction (Prediction table variables)
        Tutor (Prediction table variables)
```

Figure 5.6 The Task Structure of Tutoring Protocol 3

5.5.10 <u>A Comparison of Three Tutoring Protocols: Conclusions</u>. The evolution of the tutoring protocols was driven by the tutors' desire to obtain as much information as possible about the cognitive state of the student before the tutoring phase begins. This form the basis for developing a better student model; and we assume that the better the student model the better the tutoring and the more the student benefits. In other words the changes in the protocol are the result of our tutor's intuitive sense that "more (information) is better" before tutoring starts.

Various tutoring domains are classified as factual, causal, problem-solving (requiring procedural knowledge), etc. Our domain is both a causal and a problem-solving one. The students participating in our tutoring experiments needed to use their *causal* understanding of the functioning of baroceptor reflex to *solve* problems put by the tutor. The Socratic method is the default tutoring method used by our tutors. Many researchers (e.g., Galdes, 1990) argue that the Socratic method is not well-suited to problem-solving domains. In a problem-solving domain the student needs a chance to exercise his knowledge. The constant questioning style of the Socratic method does not provide the student with the chance to solve a problem. Our tutors also agree with these arguments.

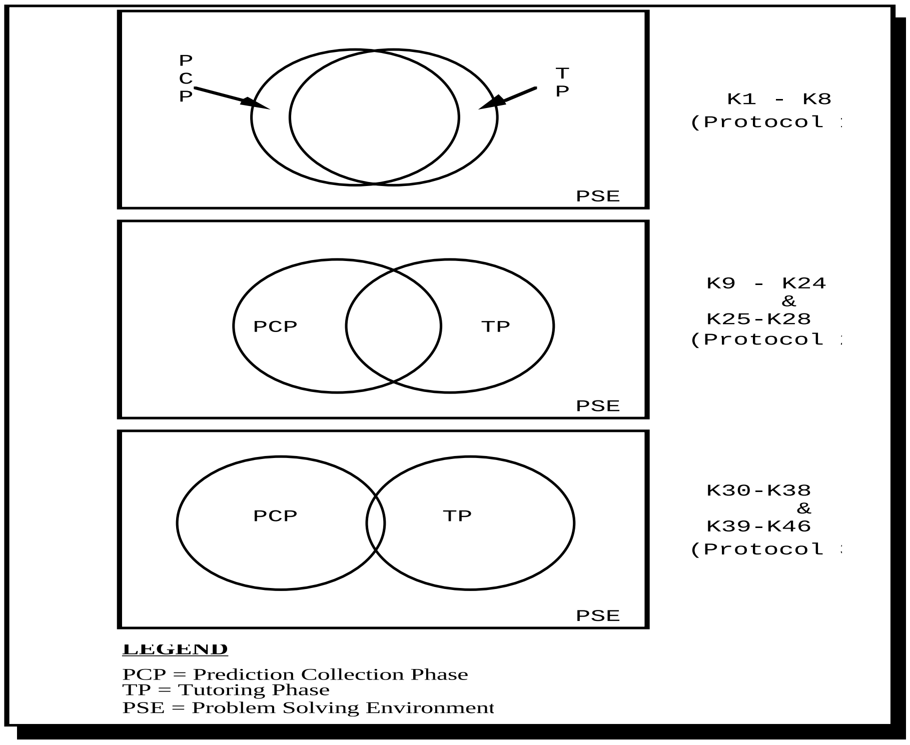


Figure 5.7 Interaction Between the Prediction Collection and Tutoring Phases

During the first set of experiments, in which the first tutoring protocol was used, there was a great overlap between PCP and TP (see Figure 5.7). Here the tutor behaved like a Socratic tutor, responding to every student prediction/response. The tutor provided *immediate* feedback without giving the student a chance to use his/her mental model to solve a complete phase of the problem. In this protocol the tutor has to use a *guess-ahead* method to visualize the problems of the student and tailor feedback accordingly. By guess-ahead we mean the process of guessing about the student's knowledge state with insufficient information. In this process it is not always possible to identify the student's deficiencies because the complete solution of the problem is not yet available to the tutor.

Although the alternative to this approach, in which the tutor has the student's complete solution does not always lead to the correct diagnosis of the student's problem, at least the tutor has much more information about the student's performance. Our tutors feel better able to make an informed diagnosis when they have collected several predictions from the student.

In the second set of experiments, in which the second protocol was used, the tutor reduced the amount of feedback *during* problem-solving. The behavior of the tutor here is more like a coach (see Section 2.1). He watches the student's sequence of predictions and interrupts only if a violation takes place. But still our tutors find it extremely difficult to interrupt and provide the student with a vague (and general) heuristic without giving specific physiology knowledge to put the student on the right track. The tutoring process starts only when a column (for DR, RR, and SS) is completely filled with predictions by the student and the tutor continued Socratic dialogue to remedy the misconceptions. With a full column of predictions the tutor can make a very fine grained diagnosis (Michael et al., 1992) of the student's problem and tailor the tutoring accordingly. This diagnosis is of course enriched by probing the student's knowledge in the tutoring phase. Due to the nature of this protocol the tutor is more informed about the knowledge state of the student than in the first protocol. This organization was much better than it was in the first protocol but still our tutors felt that the students should be given *full* chance to use their mental model of the CV system to solve a problem.

As a result, the third version of the tutoring protocol was devised to meet this requirement. There is one major way in which this protocol differs from the second protocol: No help is provided in the PCP (Figure 5.7). We believe that this is more advantageous to the student because here the student is forced to rely on his thinking (more exactly his mental model of the CV system) to solve the problem without interruption and the tutor gets a chance to understand the student's thought processes and tailor the tutoring process accordingly. This last version, to our understanding, is the best compromise to enable a successful use of the Socratic method in a problem-solving domain. This protocol provides the student with full freedom to use his/her knowledge to practice problem-solving under the watchful eyes of the tutor who in the second phase (i.e., in the TP) of this protocol tries to remedy misconceptions.

We have already decided to use this protocol for CIRCSIM-Tutor (v.3). This comprehensive analysis of the pre-session behavior of our tutors has increased our understanding of the tutoring protocol and strengthened our confidence that Protocol 3 can help achieve the goals of our system.

The tutoring protocol determines the pre-session behavior of the pedagogy expert (see Figure 5.1). Comparing the task structure of Figures 5.2, 5.4, 5.5, and 5.6, it is clear that the tutoring protocol also reflects the tutor's domain expertise (notice the division of all these task structures into three phases - DR, RR, and SS). One consequence of this influence is that the pedagogy expert can adjust the learning environment so that it makes explicit the underlying structure of the domain knowledge to the student.

The advantage of this analysis of the pre-session behavior of our tutoring is two-fold. First, it directly points to the prescriptions that are essential in the development of a machine tutor. Second, it makes explicit the pedagogical ingredients that are essential for the student's learning. We believe that an ITS should have an explicit "notion" of the tutoring protocol it uses. The tutoring protocol is a high level plan of the tutor. Hence, in our view, an ITS should have an explicit planning mechanism to handle the reasoning required for the tutoring protocol. Handling the tutoring protocol explicitly in an ITS not only enhances the flexibility of the system but also provides an experimentation tool so that a better tutoring environment (and hence tutoring protocol) could be devised for a tutoring domain. Chapter VII discusses a planning mechanism, used in CIRCSIM-Tutor (v.3), that handles Protocol 3 explicitly.

5.5.11 <u>In-Session Behavior of the Pedagogy Expert</u>. In this section we will concentrate on the behavior of our tutors observed during a tutoring session. From now on we will only consider the in-session behaviors resulting as they use Protocol 3 because it is this protocol that has been selected for CIRCSIM-Tutor (v.3). A detailed description of the model of pedagogy expert is included in Chapter VI. Here we will only describe a few important aspects. This we will achieve by analyzing a keyboard-to-keyboard transcript.

K37-tu-42-1:	Ok, let's see what you have predicted.
K37-tu-42-2:	First, though, can you tell me what DR means.
K37-st-43-1:	The changes that occurs in the students body immediately following centrifugation before and responses can be made by the body.
K37-tu-44-1:	Good definition.
K37-tu-44-2:	Can you tell me how the output or values of cc, hr, and t pr are determined?
K37-st-45-1:	Do you mean how they are calculated?
K37-tu-46-1:	No, what physiological inputs determine their outputs?
K37-st-47-1:	TPR is determined by the size of the lumen of the vessels (vasoconstriction or vasodilatation).
K37-st-47-2:	Hr is determine{INTERRUPTED STUDENT INPUT}
K37-ti-48-1:	Lets talk about TPR before dealing with the others.
K37-ti-48-2:	You are right, resistance is determined by vessel size.
K37-ti-48-3:	But, what inputs to the vessels determine their size?
K37-st-49-1:	Sympathetic stimulation
K37-tu-50-1:	Your right.
K37-tu-50-2:	Now, what causes sympathetic activity to the blood vessels to change?
K37-st-51-1:	Baroreceptors
K37-tu-52-1:	Right, reflexes, in this case the baroreceptor reflex.
K37-tu-52-2:	But, let's go back to your definition of DR, and remember that you predicted that TPR would in crease.
K37-tu-52-3:	Do you want to rethink this?
K37-st-53-1:	The reason I initially said it would increase was because there is more blood in the peripheral vessels which increases the force the walls need t o exert to keep from blowing up.
K37-st-53-2:	I was not thinking of TPR in relation to the size of the vessels.
K37-tu-54-1:	Then what change to TPR do you think will occur in DR as a result of the centrifuge?
K37-st-55-1:	I see two sides: first, DR is before the baroreceptors do anything so it would not change.

Figure 5.8 A Section of a Keyboard-To-Keyboard Transcript

K37-st-55-2: Second, if you consider that some vessels have elastic properties, the vessels would increase in size to accommodate the additional blood.
K37-tu-56-1: Let me briefly deal with your second thought.
K37-tu-56-2: Most of the blood that is displaced to the periphery is in the veins.
K37-tu-56-3: While it is true that they will get bigger, the veins contribute so little to TPR that we can ignore this effect.
K37-tu-56-4: So, your first line of thinking is correct -TPR in DR is unchanged.
K37-tu-56-5: Do you want to now think about hr and cc?
K37-st-57-1: Sure.
K37-tu-58-1: So, what change will occur to cc and hr in DR?
K37-st-59-1: Heart rate should stay the same before any baroreceptor response.
K37-st-59-2: Do you w ant me to go through what causes hr like we did with TPR?
K37-tu-60-1: No.
K37-tu-60-2: Tell me how cc will change in dr.
K37-st-61-1: I've never fully understood CC. but before baroreceptor intervention, it would not change, i think.
K37-tu-62-1: Good.
K37-tu-62-2: So, TPR, cc and hr are unchanged.
K37-tu-62-3: Let's look at your prediction for map -you said it would increase.
K37-tu-62-4: Do you want to stick with this prediction?
K37-st-63-1: No.
K37-st-63-2: Remembering that the peripheral blood would go to the veins, the MAP should decrease because there will be less blood in the arteries
K37-tu-64-1: Your prediction is now correct -MAP will decrease.
K37-tu-64-2: But let's think about why this occurs.
K37-tu-64-3: You predicted that rap, SV and CO would all decrease (and you are right).
K37-tu-64-4: So, with CO down and TPR unchanged what must map do?
K37-st-65-1: Map must decrease
K37-tu-66-1: Exactly.

Figure 5.8 A Section of a Keyboard-To-Keyboard Transcript (continued)

Figure 5.8 shows a section of a keyboard-to-keyboard transcript in which the tutor is attempting to teach the DR phase of a CV problem. This section of the transcript is unedited except to correct typographical and spelling errors. The description of the CV problem used here is as follows: “An astronaut was placed in a human centrifuge. The centrifuge was rotated to provide a force of 3 gees (3 times the force of gravity) acting from his head towards his feet."

CV Variable	**Student's Prediction**	**Tutor's (or correct) Prediction**
RAP	-	-
SV	-	-
CO	-	-
TPR	+	**0**
MAP	+	-
CC	-	**0**
HR	-	**0**

Figure 5.9 A List of Student's and Tutor's Predictions

Figure 5.8 shows a dialogue between the tutor and the student in the tutoring phase (see Figure 5.1). Figure 5.9 shows the student's predictions collected by the tutor in the prediction collection phase. This figure lists variables in the sequence in which these were predicted by the student. Here the third column lists the actual (correct) prediction for the given problem. Notice from Figure 5.9 that the student has made four errors (i.e., the predictions for TPR, MAP, CC, and HR are wrong). The section of the dialogue shown in Figure 5.8 can be divided into two major parts. The first part starts at K37-tu-42-1 and ends at K37-tu-62-2. The second one is from K37-tu-62-3 to K37-tu-66-1. The first part deals with tutoring that is triggered by the errors in the variables that our tutors call neural variables (TPR, HR, and CC). The second part deals with tutoring to correct errors in predicting MAP (the regulated variable of the BR reflex). Within the first part the tutor talks about neural variables in a particular sequence, which in the given case is TPR, HR, and then CC.

Many observations can be made from the given example of tutoring, some of these are given as follows. (1) Mostly tutoring is error driven. It is an error in the student's prediction that causes the tutor to organize a tutoring interaction with the student. (2) Before tutoring on a phase (DR, RR, or SS) of the CV system, the tutor combines errors into groups. We call these groups "error patterns." In the given example HR, TPR, and CC are all neural variables and hence are combined by the tutor and treated as a unit (see Section from K37-tu-42-1 to K37-tu-62-2 in Figure 5.8). (3) Errors/error patterns are arranged in order by the tutor before tutoring. Notice that in Figure 5.8 neural variables are tutored before MAP. (4) The major focus of the tutor in this session is to remediate underlying causes that have led the student to predict incorrectly. One major approach used by our tutors to accomplished this goal is to use a set of causes for each error pattern that he has developed through experience. We call these causes "student difficulties" (in ITS literature they have also been called misconceptions or bugs). Here the tutor additionally needs to select and order these causes in order to interact with the student. At K37-tu-42-2 (see Figure 5.8), the missing or misunderstood definition of DR is the tutor's hypothesis about what might has caused the incorrect student's prediction for the neural variables. The student's response at K37-st-43-1 has eliminated this possibility and as a result the tutor at K37-tu-44-2 has created the second hypothesis that deals with the misunderstood mechanism controlling these variables. During the dialogue the tutor also sometimes discovers misconceptions of the student that are not in the tutor's list (or bug library). Sometimes the tutor does not give high priority to some misconceptions but during the dialogue these turn out to be the major cause of the student's confusion. For example, at K37-st-55-2 the student has pointed to a misconception to which the tutor normally does not give much weight. But after detecting it as one of the major misconceptions of the student, he remediated it at K37-tu-56 before proceeding in the session. (5) During the remediation process for each student difficulty the tutor uses different models of the domain. This process is described in detail in Chapter VI.

From this partial list of observations one can conclude that the tutoring process is error driven and geared to remediate misconceptions of the student. Also the tutor makes three major decisions during a tutoring session: *What* to teach (this includes selection of errors, error patterns, and student difficulties); *When* to teach (this includes ordering and grouping of errors, error patterns, and student difficulties); and *How* to teach (this includes remediation techniques). The student modeling technique closely approximates the behavior of our tutors in a bug-library. Unlike the earlier versions of CIRCSIM-Tutor, CIRCSIM-Tutor (v.3) puts more emphasis on this technique to model the knowledge state of the student.

The cognitive model of tutoring described in this chapter is based on the behavior of our tutors in the keyboard-to-keyboard sessions. As a result, unlike the models used in earlier versions of CIRCSIM-Tutor, this model presents a much closer approximation of the behavior of our tutors. The tutoring effectiveness of the method of our tutors has now been formally evaluated. This evaluation gives us confidence in the effectiveness of this model of tutoring and confidence that it will perform well. In the next chapter we will also take a detailed look at the underlying theoretical orientation of our model of tutoring for CIRCSIM-Tutor (v.3).

CHAPTER VI

PEDAGOGY AND DOMAIN EXPERTS: A CONCEPTUAL VIEW

6.1 Introduction

In this chapter I will describe, in detail, conceptual views of the pedagogy expert and the domain expert of the model of tutoring (see Figure 5.1) for CIRCSIM-Tutor (v.3). A conceptual view of a model describes components and their interdependencies in terms of functions, rather than in terms of machine executable formalisms (Breuker, 1990). Here I will first describe a conceptual model of the pedagogy expert. This expert performs two functions that are responsible for the pre-session and in-session behavior of the tutor (see Section 5.5). Protocol 3 has been selected for CIRCSIM-Tutor (v.3). Chapter V describes, in detail, the characteristics and functionality of this protocol. In this chapter, we will concentrate only on the in-session behavior of the pedagogy expert. Next I will describe the domain expert that provides domain intelligence for the system. Here we will view the domain models constructed in the light of the pedagogical standpoint of my model of tutoring. Next I will describe the nature of integration between these two experts. And finally, in the last section of this chapter I will take a look at the underlying theoretical orientation of this model of tutoring.

For the domain expert I am concerned with representation and inferencing of the domain knowledge. For the pedagogy expert I am concerned with the representation and decision making process of tutoring knowledge. The representation of the tutoring knowledge in the form of curriculum is described, in detail, in the next chapter.

6.2 A Conceptual Model of the Pedagogy Expert

The in-session behavior of the pedagogy expert deals with the activities in the tutoring phase (see Section 5.5). According to Protocol 3, this phase starts once a column of the student's prediction is available to the tutor, i.e., when the student has completed problem-solving for a phase of the CV system. In the tutoring phase the pedagogy expert makes three major decisions: *What* to teach, *When* to teach, and *How* to teach. It is this last decision that causes the pedagogy expert to interact heavily with the domain expert.

In CIRCSIM once the student has completed predicting for a CV problem the system starts evaluating his/her responses. Conceptually, at first, this system finds out the incorrect predictions of the student by comparing it with the correct set of predictions for the given CV problem. There could be several reasons for the student's incorrect predictions. But in general terms, the student may be missing some pieces of information, or may have some misconceptions about the physiological mechanism(s) underlying the functioning of the CV system. Since CIRCSIM has no natural language capability, there is no way for this system to figure out the *exact* cause of a prediction error for the student. CIRCSIM resolves this problem by mapping prediction errors to error patterns (or bugs). An error pattern is not the underlying cause of the student's prediction error rather it is a concept that when a particular piece of knowledge is missing or in an incorrect form could yield an incorrect prediction. For example, in one of the CIRCSIM problems if the student makes an error in RAP in RR then the system maps this to the CO -> RAP relationship. This relationship is a piece of domain knowledge (see Figure 6.14 (a)). Here the assumption is that no matter what caused an error in RAP, the CO -> RAP relationship is not known by the student or is incorrectly used. Under this condition the CO -> RAP relationship is treated as an error pattern. In other words, in order to correctly predict the value for RAP, in the given problem, the student must use this causal relationship correctly. One way of doing this mapping is to check the domain knowledge needed to correctly predict a value for a CV variable. This means in order to predict correctly the value of MAP in DR in a problem, the student must use TPR -> MAP and CO -> MAP relationships correctly (see Figure 6.14 (a)). Error to error pattern mapping did not solve all problems of cognitive diagnosis but it at least provided a way of managing the complex pedagogy task in CIRCSIM, which has limited communication capabilities.

As has been mentioned before, the remediation process in CIRCSIM is not interactive. The system, after considering an error pattern for an incorrect prediction, provides feedback in a didactic fashion (in one shot) to the student. For example, for an incorrect prediction in RAP in RR the system provides the following feedback. “Remember that RAP is inversely related to CO so that if CO goes up then RAP must decrease and vice versa.” The description of the problem used in this case is “Hemorrhage: Remove 0.5 L (Blood Volume = 4.5 L).” In short, most of the feedback of CIRCSIM is organized around the problem-solving behavior of the system. I call this “providing-missing-steps-of-problem-solving” based teaching.

CIRCSIM has many influences on the behavior of our tutors in the keyboard-to-keyboard sessions (see Chapter III). It would not be incorrect to say that our tutors started these experiments with “providing-missing-steps-of-problem-solving” as a dominant model of teaching in their minds. In comparison with CIRCSIM, the keyboard-to-keyboard environment provides an opportunity to the tutor to consider at length the knowledge state of the student. At one extreme it is possible that the tutor explores the fundamental cause of each student error and then chooses the remedial feedback accordingly. At the other end, the tutor can adhere to the CIRCSIM model where only minimal information about the student is needed.

Like the student, the tutor also learns. Of course the nature and the content of learning for both parties are different. In this chapter we will not specifically talk about the tutor’s learning but it is worth mentioning that our research setup provides an excellent opportunity to investigate the tutor’s learning over time. The evolving model of tutoring for CIRCSIM-Tutor provides an excellent example of learning for our tutors. The next section describes the details of the conceptual model of the pedagogy expert.

6.3 Tutoring Cycle: A Process of Making the Student Active While Learning

The cognitive diagnosis of the knowledge state of the student in CIRCSIM-Tutor (v.3) is performed by the student modeler (see Figure 4.2). It is the pedagogy expert that uses information about the student generated by the student modeler. As soon as the prediction collection phase of Protocol 3 finishes, the student modeler generates an initial evaluation of the knowledge state of the student. Based on this information the pedagogy expert develops a plan for a tutoring interaction with the student.

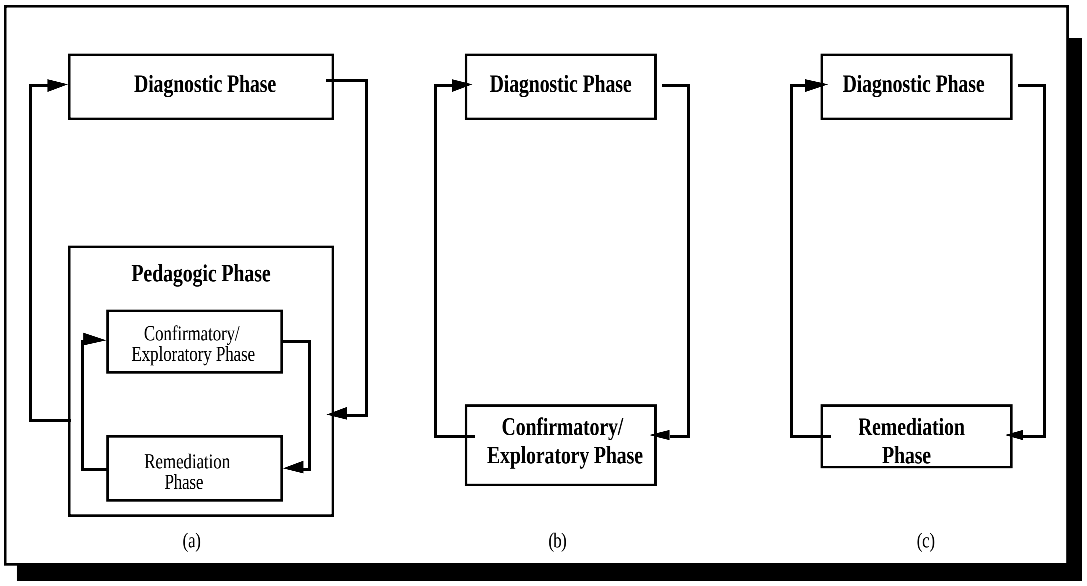


Figure 6.1 Tutoring Cycle of CIRCSIM-Tutor (v.3)

During tutoring CIRCSIM-Tutor (v.3) alternates between two major phases: the diagnostic phase and the pedagogic phase (see Figure 6.1(a)). In the diagnostic phase the student modeler builds a model of the student. Based on this model, the pedagogy expert in the pedagogic phase engages either in confirmatory/exploratory activity or remediation activity. Interestingly these two phases also alternate in the pedagogic phase (see Figure 6.1 (a)). In the confirmatory/exploratory phase either the tutor *confirms* a hypothesis about the knowledge state of the student or it *explores* the underlying cause of a student's error in prediction. If the tutor is successful in achieving either of these goals then the remediation phase is invoked. Here the tutor tries to remediate the current misconception of the student. On the other hand if the tutor fails in its diagnostic endeavor, a default remediation strategy is selected to tutor the student for his/her current problem(s). It is interesting to note that in any case the tutor is always either in the diagnostic/confirmatory/exploratory cycle (see Figure 6.1 (b)) or in the diagnostic/remediation cycle (see Figure 6.1(c)). These phases and cycles are described further in Section 6.5. Comparing this model with the teaching model of CIRCSIM (see Section 6.2), it is obvious that our tutors in the keyboard-to-keyboard sessions are substantially active in diagnosing and remediating the student's problems. This activity is due to the extended opportunity available in these sessions to actively explore the underlying cause of the student's suboptimal behavior in solving a CV problem. Before I describe this model further, it is imperative to specify a view of the student as seen by the tutor and its diagnostic process during tutoring.

6.4 A Tutor's View of the Student

This section briefly describes a view of the student as seen by the tutor and the process through which this view is created. In CIRCSIM-Tutor (v.3) it is the student modeler that handles this job. Initially these ideas were developed by Greg Hume and me working together, but now they have been greatly extended by Greg into a full-fledged student modeler (see Hume, in preparation).

For CIRCSIM-Tutor (v.3) we assume that the student who comes for tutoring may possess a number of misconceptions that are the main source of his/her incorrect predictions for a CV problem. In order to discover the actual misconceptions confusing this student, the tutor adopts a layered approach to diagnosis. Because of the constraints of Protocol 3 the only form of information available to the tutor, at first, is a set of errors (more specifically, wrong predictions of the student). These errors, based upon their individual characteristics, determine a number of error patterns see Figure 6.2 (b) for an example). These are not the actual misconceptions of the student but rather bring the tutor one step closer to finding the actual cause of the student's error. In the next step of diagnosis the tutor uses these error patterns to hypothesize a number of causes (we call them student difficulties) that could be the actual source of suboptimal behavior of the student. See Figure 6.2 (a) for a schematic view of three layers of the tutor's view of the student.

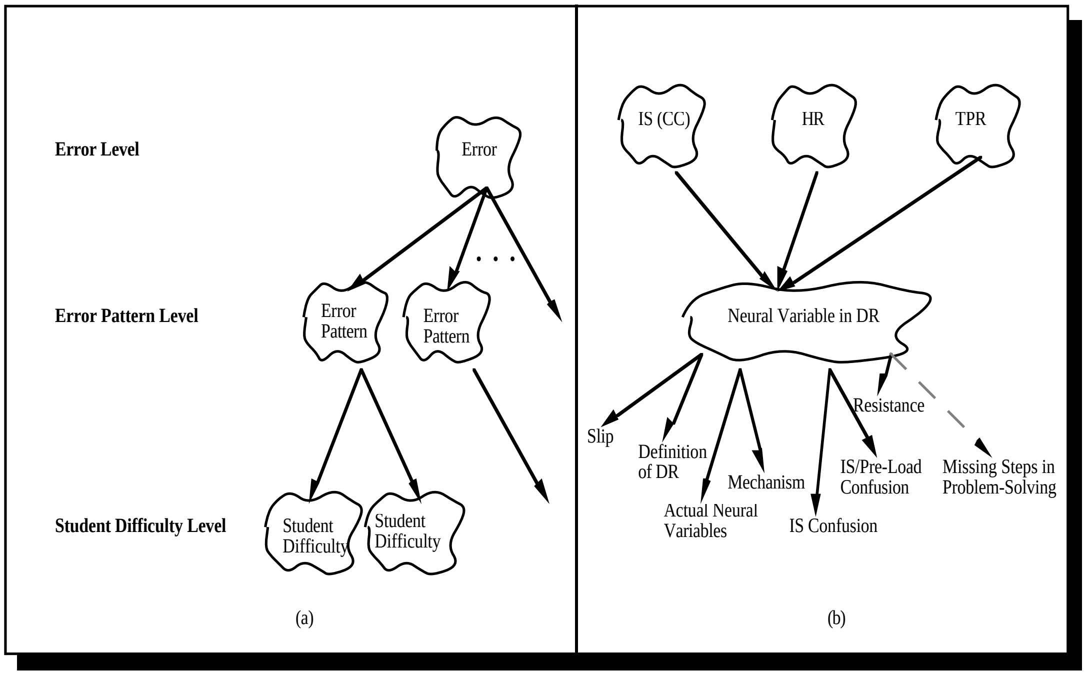


Figure 6.2 A Schematic Representation of the Tutor's View of the Student

The certainty of errors, error patterns, and student difficulties varies in this diagnostic process for the student. Errors are most certain because they are the wrong predictions for a CV problem. One definition of an error pattern makes it almost certain, once it is detected. The student difficulty is a relatively complex concept. Here the tutor is not quite sure of the cause of a student's error. A student difficulty can be a legitimate misconception of the student. For example, an error in predicting the value of a neural variable in DR is caused by the student's incorrect understanding of the definition of DR. Alternatively, it can be a minor flaw in the student's performance (such as a slip). For example, the student has momentarily forgotten that he/she is solving a CV problem for the DR phase and made an incorrect prediction for a neural variable. If no student difficulty exists for an error pattern then the tutor turns to the view as represented by the "providing-missing-steps-of-problem-solving" based teaching scenario (see Section 6.2). The next section describes the way that the pedagogy expert uses this layered model of the student.

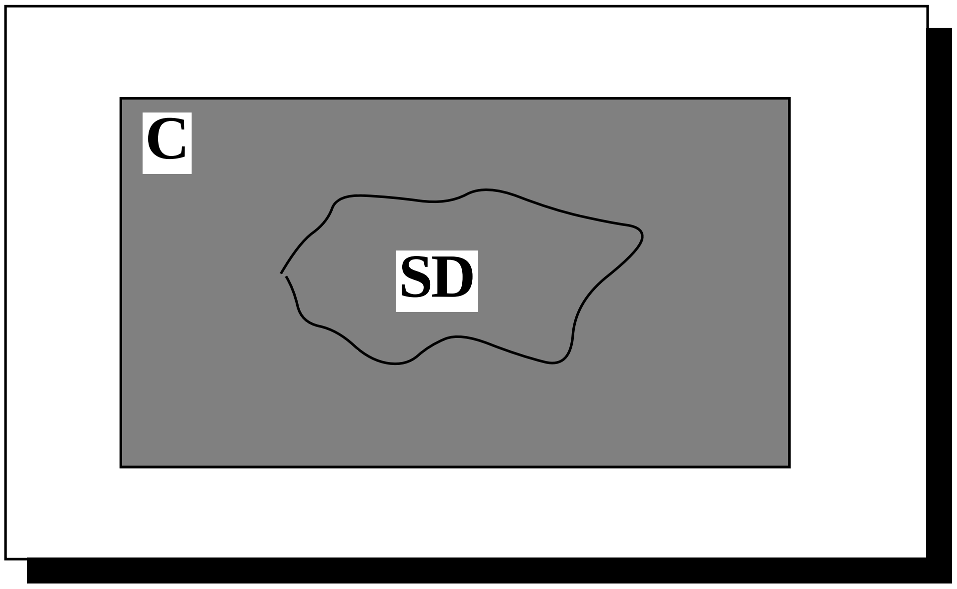


Figure 6.3 A Space of Causes for the Student's Wrong Predictions

It is the error pattern level that substantially reduces the space of potential misconceptions of the student for the tutor's diagnostic process. A large space of potential causes exist for each error in the student's predictions (marked as "C" in Figure 6.3). The error pattern level forces the tutor to view the student only through a limited set of possibilities that are related to the steps in the problem solving process. Thus, only the student difficulties (marked as "SD" in Figure 6.3) that are related to these steps are considered as the potential causes for the student's suboptimal behavior. This reduces the search space for the diagnostic process for the student modeler. It also makes it possible to develop a library of student difficulties from experience. The next section sheds more light on this issue. Figure 6.2 (b) shows an example hierarchy relating errors, error patterns, and student difficulties for CIRCSIM-Tutor (v.3). This figure shows that an error in any neural variable will cause the sensitization of the “neural variable changed in DR” error pattern. The sensitization process, used in the student modeler, allows the detection of all error patterns caused by errors in the student’s prediction. Any of the student difficulties attached to this error pattern can cause wrong predictions for that neural variable.

6.5 <u>Pedagogic Phase</u>

In this section I will elaborate on the activities of the tutor in the pedagogy phase in the tutoring cycle of CIRCSIM-Tutor (v.3) (see Section 6.3). Here I will assume that the diagnostic phase (see Figure 6.1) uses the view of the student as described in Section 6.4.

K40-tu-42-4:	My question to you is, can you define cardiac contractility (CC)?
K40-st-43-1:	I think it is the related to the length tension relationship of the cardiac fibers.
K40-st-43-2:	The more volume of blood within the chamber the farther the heart stretches and the greater the contraction.
K40-st-43-3:	isn’t that Frank-Sterling's law
K40-tu-44-1:	You have indeed described the Frank-Starling law of the heart.
K40-tu-44-2:	But that's not what's meant by contractility.
K40-tu-44-3:	Contractility is the inotropic state of the heart and can be changed without altering the preload (EDV or EDP or R AP).

Figure 6.4 Tutor Confirming a Hypothesis

By default, as soon as the pedagogy expert selects an error pattern from a sensitized set (see Section 6.6.1) the confirmatory/exploratory phase is invoked. Here potential student difficulties associated with the selected error pattern are considered. Each student difficulty, representing a possible misconception of the student, acts as a potential hypothesis for the tutor. The tutor is not sure about its existence in the student’s behavior. After invoking the confirmatory phase the tutor might establish that the selected hypothesis (that the student has a particular student difficulty) is correct or incorrect. If it is correct then the remediation phase is activated. Figure 6.4 shows an excerpt from a keyboard-to-keyboard session that reflects this behavior of the tutor. Here at K40-tu-42-4 the tutor has a hypothesis that the student has a misconception about CC. This misconception in CIRCSIM-Tutor (v.3) is referred to as “IS/Preload confusion.” At K40-tu-42-4 the tutor invoked the confirmatory phase. The student’s reply at K40-st-43 confirmed the tutor’s hypothesis. As a result at K40-tu-44-3 he started the remediation phase. If the tutor fails in establishing a hypothesis then the next one in the list is selected and the cycle is repeated.

If the selected student difficulty is a slip (see Section 6.4), instead of a potential misconception, then the tutor still needs to confirm it before proceeding in the session. Figure 6.5 shows an instance of this case. Here at K26-st-21-1, the student has made an error in predicting a value of the primary variable. Since the problem description in this case explicitly states the direction of change in HR, the tutor here hypothesized that it is a slip. As a result he, at K26-tu-22-1, provided a hint with an expectation that the student will realize the mistake and recover from it. This also provided the tutor a chance to confirm the existence of this slip. At K26-st-23-1 the student’s response confirmed that it was a slip and hence at K26-tu-24-1 the tutor proceeded with the session.

K26-tu-18-1:	Ok, then let's start by telling what parameter you want to predict first (but not how it will change).
K26-st-19-1:	Hr
K26-tu-20-1:	How will it change?
K26-st-21-1:	I
K26-tu-22-1:	Will you re-read page 3 and confirm for me what this experiment is?
K26-st-23-1:	Sorry i got confused it is obviously d
K26-tu-24-1:	What parameter do you want to predict next?

Figure 6.5 Tutor Handling a Student’s Slip

Sometimes during the dialogue the tutor is unable to establish any potential misconception for the student. In such a case the tutor then invokes exploratory phase by asking open ended question(s) to the student to establish and then confirm a potential hypothesis for the student. Figure 6.6 shows an instance of this case. At K12-st-32-1 the student provided an incorrect prediction for TPR. Here the tutor was not sure about the mechanism by which the student predicted the wrong value for TPR. As a result, at K12-tu-33-1, he asked an open ended question to establish a cause for it. At K12-st-34-1 the student revealed a mechanism that in the tutor's view is incorrect. At K12-tu-35-4 the tutor tried to confirm the mechanism that in his opinion is misunderstood by the student. This is established at K12-st-36-1 and as a result the tutor initiated the remediation phase at K12-tu-37-1. Although our tutors exhibit this behavior, we do not plan to implement it in CIRCSIM-Tutor (v.3) because the natural language components cannot understand this language well enough.

K12-tu-31-1:	Now how about TPR?
K12-st-32-1:	I'm thinking that it will increase very briefly but immediately decrease so as to adjust, back to normal the CC
K12-tu-33-1:	By what mechanism will it increase?
K12-st-34-1:	If you increase pressure will you momentarily increase resistance
K12-tu-35-1:	no. You may be thinking of autoregulation.
K12-tu-35-2:	That's slow.
K12-tu-35-3:	Remember that we're dealing with the short period before you get a reflex response.
K12-tu-35-4:	Is this what you had in mind?
K12-st-36-1:	Yes i guess i am not sure then what happens to TPR
K12-tu-37-1:	What is the primary mechanism of control of TPR?
K12-st-38-1:	Radius of arterioles
K12-tu-39-1:	Yes.
K12-tu-39-2:	And what is the primary mechanism by which arteriolar radius is controlled?

Figure 6.6 Tutor in an Exploratory Mode

In certain cases while in the exploratory phase the tutor realizes that following the student's reasoning any further will just complicate matters instead of helping the student.

The tutor then selects a default strategy and takes charge of the current situation at hand. Figure 6.7 shows this behavior of the tutor. At K38-st-187-1, the student has predicted TPR incorrectly. In order to understand the cause of this wrong prediction, the tutor at K38-tu-188-1 initiated the exploratory phase. At K38-tu-192-1, the tutor after understanding the student's problem realized that pursing the student's reasoning still further to remediate the misconception would probably not be beneficial. As a result at K38-tu-192 the tutor started to guide the student to visualize the domain knowledge according to a default strategy.

K38-tu-186-1:	And TPR?
K38-st-187-1:	will decrease
K38-tu-188-1:	Why?
K38-st-189-1:	to allow for the incr. SV and CO
K38-tu-190-1:	I don't understand what you just said.
K38-st-191-1:	by decrease TPR, it will be easier to push the blood out of the let ventricle and it will be easier to accommodate the incr. SV and hr
K38-tu-192-1:	I understand what you are saying, but let's think about the situation this way.
K38-tu-192-2:	What is the physiological input that determines TPR?

Figure 6.7 Tutor Directing the Student's Thinking

In cases where the tutor runs out of the potential misconceptions, a default strategy is invoked to remediate student's problem. The underlying assumption for this strategy is similar to that used in CIRCSIM, i.e., instead of worrying about the actual misconception(s) causing the suboptimal behavior of the student, it makes sense to provide missing steps of problem solving (see Section 6.2). The hope is that (1) it can provide to the student, the steps needed to correctly solve a problem, and (2) these steps will cause the disappearance of the actual misconception causing his/her suboptimal behavior or at least overshadow it. Figure 6.8 shows an example of the use of this default strategy.

Full details of this default strategy are given in Section 6.17. But briefly it has four components: figure out the determinants, decide which is the important determinant in the current case, recall the relationship between the determinant and the parameter to be predicted, and predict the value. At K14-tu-49-2 the tutor asked the student for the determinants of SV. The student responded correctly at K14-st-50-1. Next the tutor combined the second, third, and fourth components and asked the student to establish a value for SV. The student at K14-st-52-1 predicted an incorrect value for SV. At K14-tu-53-1 the tutor elaborated on his last question at K14-tu-51-3. This caused the student to make a correct prediction at K14-st-54-1. It is interesting to note that CIRCSIM-Tutor (v.2)'s behavior is mostly based on this strategy. Unlike CIRCSIM, this default strategy in CIRCSIM-Tutor (v.3) is more actively pursued by the tutor. In fact, steps taken in this strategy are based on our tutor's core reasoning mechanism to solve a CV problem. This mechanism uses a very novel structure called the inference triangle. A detailed view of this structure and its associated reasoning mechanisms are described in Section 6.17.

K14-tu-49-2:	What I was asking is what determines how much blood is ejected from the heart each time it beats (the SV)?
K14-st-50-1:	RAP and CC
K14-tu-51-1:	Good.
K14-tu-51-2:	Well, you made predictions about how RAP and CC would change as a result of the pacemaker malfunction.
K14-tu-51-3:	What do you think will happen to SV?
K14-st-52-1:	0
K14-tu-53-1:	Well, you predicted that RAP would in fact go down (since CO went up) and you predicted that CC would not change (and that's correct) 0.
K14-tu-53-2:	So, what happens to SV?
K14-st-54-1:	D
K14-tu-55-1:	Right, SV goes down.

Figure 6.8 Tutor Using a Default Strategy

Developing a set of student difficulties for each error pattern is a gradual process. Our tutors learn about student difficulties via many sources. The research environment for CIRCSIM-Tutor (see Section 4.5) is one of the sources for this type of tutor learning. I hypothesize that with time the library of student difficulties for CIRCSIM-Tutor will grow and our tutors will put more and more emphasis on misconceptions of the student rather than, as in CIRCSIM, relying on the default strategy to remediate the student's underlying problem.

Once all error patterns have been considered by the pedagogy expert, it brings up some generic (but very essential) topics to discuss with the student. The pedagogy expert does this only when sufficient time is available while tutoring a phase of the CV system. Figure 6.9 shows an excerpt in which the tutor invokes a topic after considering all error patterns detected for the student. In this example the student made correct predictions for all variables in the prediction table for DR. Realizing this, the tutor planned at K46-tu-50-3 to talk about a topic that he thinks it is important for the student to know. At K46-tu-50-4 this topic is invoked. Since the student at K46-st-51-1 demonstrated knowledge of this topic, the tutor proceeded to the next phase of the session. The selection of a topic is based on the curriculum available to the pedagogy expert. A more detailed discussion on this behavior is provided in Chapter VII.

K46-tu-50-1:	Ok, super job.
K46-tu-50-2:	I'd like to think that you must have learned something from CIRCSIM.
K46-tu-50-3:	Let's talk about a few things however.
K46-tu-50-4:	What do CC, HR and TPR have in common?
K46-st-51-1:	They're all sympathetically controlled, after a baroreceptor signal initiate s the action.
K46-tu-52-1:	Right.

Figure 6.9 Tutor Invoking a Generic Topic

6.6 <u>Major Decisions Made By the Pedagogy Expert</u>

This section describes three major decisions that the pedagogy expert makes in CIRCSIM-Tutor (v.3). These are: What to tutor, When to tutor, and How to tutor. Figure

6.10 shows a summary of these decisions supported at different levels of the view of the student.

Major Question	**What to Tutor?**	**When to Tutor?**		**How to Tutor?**
Associated Decision(s)	**Selection**	**Grouping**	**Sequencing**	**Remediation**
Error	YES	YES	YES	NO
Error Pattern	YES	NO	YES	NO
Student Difficulty	YES	NO	YES	YES
Topic	YES	NO	YES	NO

Figure 6.10 Summary of Decisions Supported By the Pedagogy Expert

6.6.1 <u>What to Tutor: Selection Decision</u>. Considering the view of the student described in Section 6.4, the pedagogy expert deals with the *selection* of errors, error patterns, and student difficulties. Considering the nature and organization of CIRCSIM-Tutor (v.3), it is the student whose behavior in the prediction collection phase determines the number of errors. The student modeler in CIRCSIM-Tutor (v.3) is responsible for detecting these errors as soon as the prediction collection phase finishes. Each error has associated pointers to its underlying error patterns and each error pattern in turn has pointers to its underlying student difficulties. As soon as the errors in the student's predictions are identified, the student modeler selects error patterns and student difficulties for the student. These lists of errors, error patterns, and student difficulties are then used by the pedagogy expert to perform decision making during the tutoring phase.

Once selected, the pedagogy expert must consider each error, error pattern, and student difficulty except in cases where certain rules prevent us from doing this. These rules only apply to error patterns and student difficulties. It is a goal of the pedagogy expert to develop plans such that all errors made by the student are discussed during a tutoring session, except in cases where the student decides to discontinue a tutoring session without completing it or where the tutor's evaluation of the student indicates that the student has several gaps in prerequisite domain knowledge and hence it is not worth pursuing the session any further with the student.

K5-tu-21-5:	I'd like you to think about some of the other variables in the table.
K5-tu-21-6:	Especially variables that are immediately and directly determined by HR.
K5-st-22-1:	HR I and CO I.
K5-tu-23-1:	Great.
K5-tu-23-2:	That's where you should have started to begin with.
K5-tu-23-3:	Now what's affected next?

Figure 6.11 An Early Finish of the Tutor's Default Strategy

Once all errors are discussed in a session, there are two major criteria that allow the tutor to select remaining error patterns: time and (student) history. The pedagogy expert has a maximum time limit for each phase of a CV problem. If all errors have been covered well within the limit then the tutor selects one of the remaining error patterns for discussion with the student. As soon as the time is up, the tutor discontinues selecting error patterns and proceeds with the next CV phase or problem. On the other hand, if the student model indicates that an error pattern has been selected in a previous CV phase or problem and the student has demonstrated knowledge that is required to eliminate that pattern then the pedagogy expert drops that pattern from consideration. For example MAP = CO x TPR is an important equation that the tutor wants the student to learn. There is a corresponding error pattern in CIRCSIM-Tutor (v.3), which is sensitized if an error appears in any of three variables in this multiplicative relationship. If the student in a previous phase or problem has demonstrated an understanding of this relationship then from the pedagogic expert's point of view there is no need to again arrange a tutoring interaction about it with the student.

Rules to select a student difficulty are quite straightforward. Each sensitized error pattern points to a set of student difficulties. Initially these are all considered for the tutoring session. As is mentioned in Section 6.3, a major objective of the tutor is to find the actual cause (a student difficulty) that is the source of the student's error. As soon as this source is established the pedagogy expert discontinues selecting student difficulties any further for that error. Each student difficulty points to a set of topics that needs to be considered by the pedagogy expert to confirm/explore and remediate that student difficulty. For example when the default strategy is considered to remediate a student difficulty involving a relationship between parameters, the following four topics needs to be considered: (1) determinant, (2) determinant in the current case, (3) relationship, and (4) value (see Section 6.17). Although this strategy provides a default sequence, the actual selection of each topic depends upon the dialogue carried on by the tutor and the student. Figure 6.11 shows an except from a keyboard-to-keyboard transcript. In this excerpt the tutor started to use this strategy at K5-tu-21-6 by invoking its first topic - "determinant." The student, at K5-st-22-1, provided a response that completed the tutor's strategy. As a result the tutor, at K5-tu-23-2, without invoking the remaining topics of this strategy, proceeded on with the session.

6.6.2 <u>When to Tutor: Grouping and Sequencing Decisions</u>. First consider the grouping decisions of the pedagogy expert while tutoring. Errors in our system are wrong predictions of the student. It is the individual characteristics of these errors that bring them together to form patterns at the error pattern level (see Figure 6.2). Hence error patterns group errors in a natural way. CIRCSIM-Tutor (v.3) is mostly inheriting error patterns from earlier systems/versions that teach about the functioning of the BR reflex (see Chapter III). CIRCSIM makes a distinction between procedure dependent and independent error patterns. Also there are other ways of grouping error patterns, for example, multiplicative relationships (MAP = CO x TPR, CO = HR x SV) could be grouped together. Although these groups have been created at the code level of the student modeler but no distinction has been made between error patterns at the conceptual level. One reason for this is that the grouping of error patterns is a difficult task because until the student is deeply diagnosed by the tutor it is not possible to pinpoint the misconceptions that are the actual cause of a set of error patterns. Our tutors in the keyboard-to-keyboard sessions seem not be engaged in such intense diagnostic activity. It is also possible to group student difficulties according to their individual characteristics but further research is needed to find out the consequences of this effort on the conceptual model.

Sequencing (or ordering) decisions are more interesting in my model of tutoring. These decisions are made at the error level and the error pattern level, and also at the student difficulty level. Since these three levels are hierarchically connected, the question here is at what level (called the base level) this decision process should start. The answer to this question depends on two criteria: the *specificity*, and the *certainty* of the information in these layers. Specificity deals with the actual source of the student's suboptimal behavior. Certainty deals with the probability of existence of given concepts under given conditions. Ideally, the student difficulty level needs to be considered as the base level for this decision because at this level entities are most specific. But unfortunately student difficulties are least certain. Also in certain cases no specific cause has been identified for an error pattern. The error level is most certain but least specific. The error pattern level is a better compromise compared to other two levels. As a result, the ordering decisions of the pedagogy expert revolve around the error pattern level. Interestingly in CIRCSIM ordering decisions are also organized around the error pattern level (note that this system does not have a student difficulty level). In CIRCSIM error patterns are ordered according to a problem-solving algorithm used by our tutors to solve a CV problem. The developers of this system argue that this organization will make the underlying structure of domain problem solving knowledge explicit to the student. In CIRCSIM-Tutor (v.3), the pedagogy expert, on the contrary, bases ordering decisions for error patterns on different strategies. Some of these strategies are domain dependent and others are domain independent. For example in DR the tutor uses a domain independent strategy called expediency. This strategy selects an error pattern whose cause is not serious but the tutor wants to “get it out of the way” of the student’s more serious problems. Littman et al. (1985) called this strategy “prepare the way for the most serious problem.” Besides these domain independent strategies this model also exploits the domain relations (by using the problem-solving algorithm) to order the error patterns. We call one such strategy “core causal chain relevancy.” This strategy orders the errors according to the main causal path from the variable that is first affected in the CV system to the regulated variable. The next chapter lists these strategies in detail.

As soon as error patterns are ordered and one of them is selected the pedagogy expert checks for the errors that are associated with this selected error pattern. For example, assume that the student has incorrectly predicted all three neural variables and as a result the pedagogy expert has selected the “neural variables in DR” error pattern (see Figure 6.2 (b)). Here it is imperative to sequence the neural variables to be discussed with the student. If HR is not the primary variable our tutors prefer to start with HR and then generalize to the other two neural variables. Otherwise, they start with TPR. As soon as an error is selected, the pedagogy expert orders the student difficulties. Chapter VII lists a set of ordering decisions for student difficulties. Each student difficulty, as mentioned before, points to a set of topics that the tutor wants to discuss with the student. The order in which these topics are raised depends upon the dialogue carried on between the tutor and the student.

6.6.3 <u>How to Tutor: Remediation Techniques</u>. In the remediation phase the tutor has to consider how to approach tutoring so as to remediate the misconceptions of the student. In my model of tutoring the pedagogy expert performs this function by using both tutoring knowledge and domain knowledge. This function is achieved in two phases. In the first phase, this model, using various strategies (e.g., use the discovery method) and tactics (e.g., give a pt-hint) develops a response for the student. CIRCSIM-Tutor (v.3) uses a natural language interface to communicate with the student (Evens et al., 1993). The output of the first phase is not in a form to be directly displayed to the student. The second phase is achieved by the communication expert (see Figure 4.2) that converts the output of the first phase into a natural language response for the student. A detailed description of the tutoring knowledge in the form of tutoring strategies and tactics is given in the next chapter.

The way domain knowledge is used by the pedagogy expert to remediate misconceptions forms a novel characteristic of my model of tutoring. Here the pedagogy expert uses different models of the domain to support the remediation process. In other words my model of tutoring, using different models of the domain, assists the student (because, who is, most of the time, actively participating in the learning process) to integrate his/her knowledge of the domain. The type and the nature of the domain knowledge and possible inferences out of it belong to the domain expert (see Figure 5.1). The next sections describe this knowledge in detail. The domain models used in CIRCSIM-Tutor (v.3) are based on the behavior of our tutors.

6.7 <u>Pedagogy Expert in Action</u>

The previous sections describe various decisions the pedagogy expert makes during a tutoring session. This section combines all these decision making processes in a flow chart to make the dynamic behavior of the pedagogy expert explicit. This flow chart is shown in Figure 6.12. Here we assume that the student has completed predicting a column of the prediction table. This dynamic behavior of the pedagogy expert is explained as follows. In order to facilitate the explanation of this behavior, various steps in Figure 6.12 are tagged with numbers in small circles.

The decision making process in Figure 6.12 starts at “1” where the pedagogy expert collects information about the student’s errors, error patterns, and student difficulties from the student modeler. Next at “2” it selects an error pattern for consideration. Let’s call this the current error pattern. At “3” if the current error pattern has multiple errors associated with it then the pedagogy expert selects one of them for consideration. Let’s call this the current error. Next considering the current error pattern, the pedagogy expert decides, at “4,” whether it can form a hypothesis about the cause of the current student error. If it forms a hypothesis then a student difficulty is selected at “6.” If it does not form a hypothesis then a default tutoring strategy is selected (at “5”) to remediate (at “9”) the student’s current error. In the case when the pedagogy expert can hypothesize about the student’s underlying problem, at “7,” it tries to confirm this hypothesis. If this hypothesis is confirmed, at “8,” then a remediation phase (at “9”) is invoked. If the tutor’s effort is not successful at “8” then it again decides, at “4,” whether it can make another hypothesis about the student.

As soon as the remediation phase is completed at “9,” the pedagogy expert checks (at “10”) whether all errors are tutored or not. If all are not yet covered then the tutor, at “2,” repeats the above mentioned process. On the contrary if the tutor has completed tutoring for all the student’s errors then, at “11,” considering the time elapsed during this tutoring, it decides whether to continue tutoring about the current phase of CV system or not. If time does not permit the system to do that then this decision making process halts until a new set of predictions for the next column of the prediction table is available. On the other hand if there is time available for the current phase of the CV system then at “12” the tutor checks for the current list of error patterns. If some error patterns are still left then, at “2,” it repeats the above mentioned cycle, else, at “13,” the system selects a generic topic that it considers important for the student to learn. At “14” this topic is planned for tutoring with the student. At “15,” the pedagogy expert again looks at the clock. If time permits then it selects another topic, else this process is halted for the current phase of the CV system.

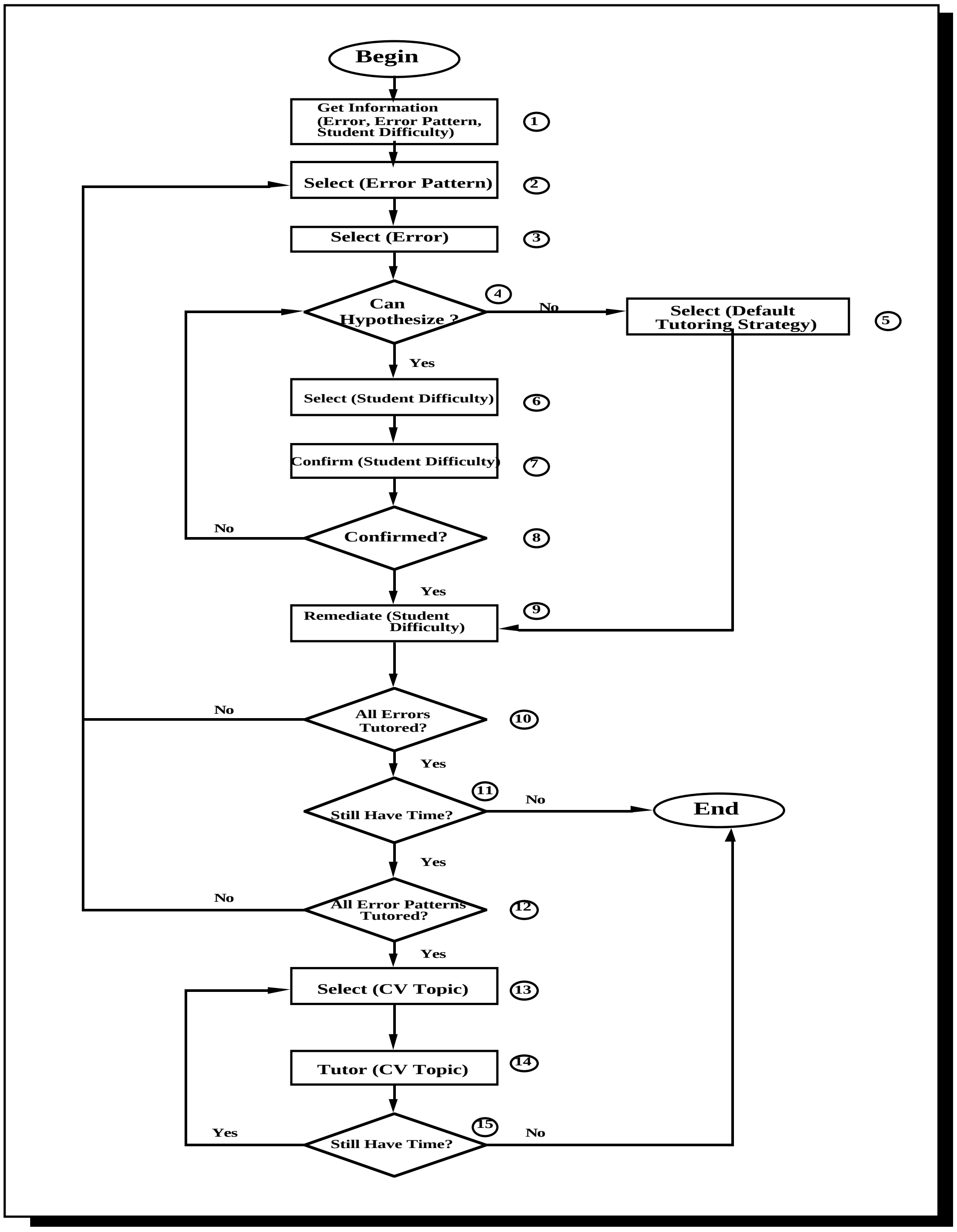


Figure 6.12 Flow Chart Representing the Dynamic Behavior of the Pedagogy Expert

6.8 A Conceptual Model of the Domain Expert

The purpose of this and the next few sections is two-fold: (1) describe a conceptual model of the domain expert (see Figure 5.1), and (2) show how this model influences the pedagogy expert in its decision making.

The domain expert of CIRCSIM-Tutor (v.3) has much broader functionality than just to help the pedagogy expert in its decision making process. The domain expert provides domain intelligence to the whole system (see Figure 4.2). The student modeler uses this expert to build the student model. The communication expert uses it to understand and generate a natural language response to the student. The conceptual model described here serves this broader purpose but I will put more emphasis on its utility as a source of the domain intelligence for the pedagogy expert. It will also be obvious from these sections that the tutoring method of our tutors has also greatly shaped the domain expert of CIRCSIM-Tutor (v.3).

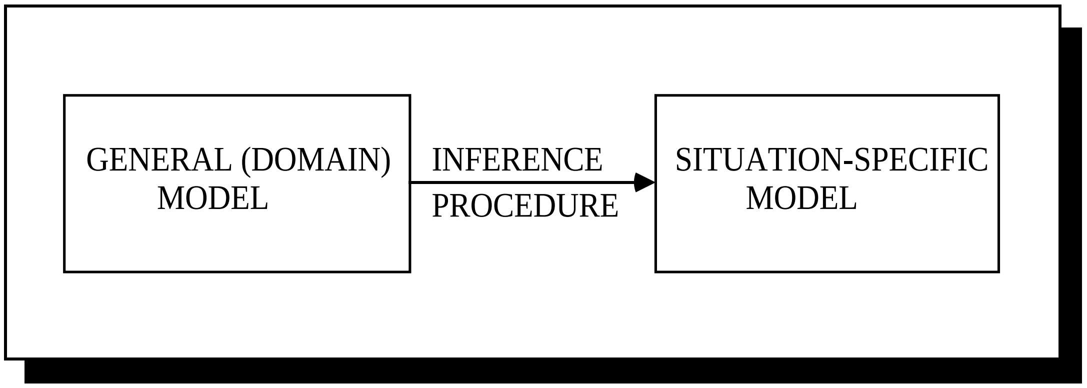


Figure 6.13 Problem Solving: Applying a General Model to Form a Situation-Specific Model (Adapted from (Clancey, 1986))

The type of task used by the domain expert is *prediction*. This expert uses a problem-solving method to predict the qualitative changes for a set of physiology variables in response to a perturbation acting on the CV system of the patient under consideration.

We will use Clancey’s (1986) definition for the general (domain) model, and the situation-specific model. A general (domain) model “describes what is known about the world, for example, knowledge about stereotypic patients and about diseases.” A situation-specific model “is a description of some situation in the world, generally an explanation of how a situation came about.” Figure 6.13 shows a view of problem-solving. In this view “a general model is related to the current situation by applying an inference procedure” (Clancey, 1986).

In these terms, then, the domain expert solving a prediction problem in our domain needs to form a situation-specific model by applying the inference procedure on the domain model of the CV system. Section 5.5.1 describes a problem-solving procedure used by our tutors to solve a CV problem. A version of this procedure has been used by the domain expert of CIRCSIM-Tutor (v.3) (see Chapter VIII). Here we will concentrate more on the general models of the domain used by our tutors while tutoring. One of these models, called the top level concept map, is shown in Figure 6.14 (a). Figure 6.14 (b) shows a possible situation-specific model obtained by performing inferencing, using the problem-solving method of Section 5.5.1.

We will use Clancey’s (1986) definition of the general (domain) model, and the situation-specific model. A general (domain) model “describes what is known about the world, for example, knowledge about stereotypic patients and about diseases;” a situation-specific model “is a description of some situation in the world, generally an explanation of how a situation came about.” Figure 6.13 shows a view of problem-solving. In this view “a general model is related to the current situation by applying an inference procedure” (Clancey, 1986).

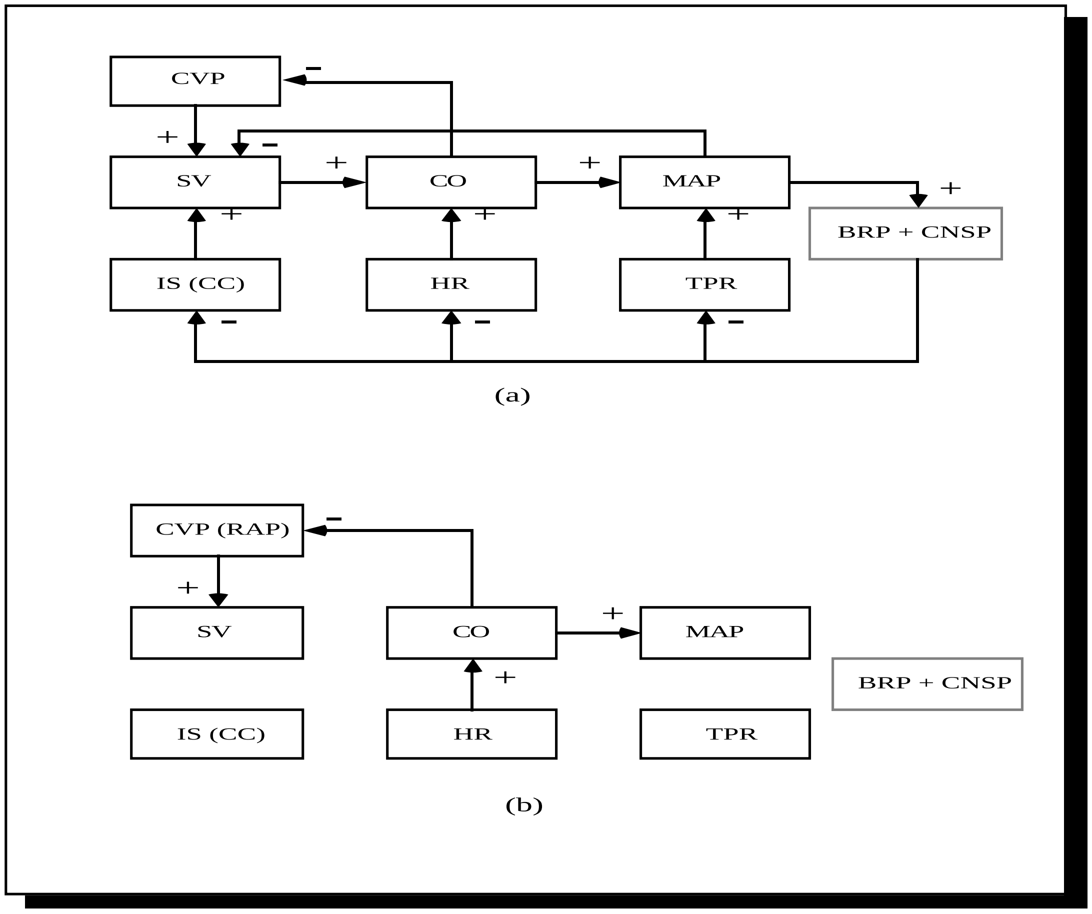


Figure 6.14 (a) A General Model of the CV System, (b) A Situation-Specific Models of CV System (in DR when Artificial Pacemaker is Malfunctioning - Only Major Relationships are Shown)

As soon as a CV procedure is selected (by either the tutor or the student) the domain expert using its inference procedure solves that problem. This activity is performed prior to any interaction that takes place between the tutor and the student. At that point the domain expert offers three types of knowledge to the rest of the system: (1) support knowledge (captured by the general model(s) of the domain), (2) operational knowledge (represented by the problem-solving procedure), and (3) situation-specific knowledge (captured in the situation-specific model of the CV system for the current procedure). These knowledge types help the pedagogy expert to develop plans to interact with the student.

6.9 <u>Domain Knowledge Background for CIRCSIM-Tutor (v.3)</u>

In Medical Physiology courses (and in recommended textbooks), cardiovascular function is taught at a very wide range of organizational levels. These levels extend from the interaction of the system's organ components on one extreme to the physical and chemical events that occur in the individual cells that make up these organs on the other. Understanding the function of the CV system at all of these organizational levels is necessary because physicians use information from all of them to evaluate their patients' health status and because the therapeutic interventions that physicians use act at many different levels. Understanding CV function at the uppermost (organ) level is most important because much of the initial information that a physician can obtain through the examination of patients relates to activity at that level. The primary goal of CIRCSIM-Tutor is to assist students to correctly predict the responses of the CV system at the top level as they relate to the regulation of mean arterial blood pressure (MAP), a process that is essential to the maintenance of adequate blood flow to the individual organs.

When students come to use CIRCSIM-Tutor, they do not know that this is a system goal, except that the predictions that they are asked to make relate to parameters that reflect CV function at the organ level only. Therefore, it was initially thought by the designers and developers that the system would only have to contain CV knowledge at that level. However, it quickly became obvious from the inspection of keyboard-to-keyboard transcripts of tutoring sessions that this was not correct. Both tutors and students use more detailed knowledge, knowledge at deeper organizational levels and knowledge with a somewhat different perspective from that contained at the core organ level. Tutors use this other knowledge to construct hints and explanations, and students use it to explain their thinking and to respond to questions.

At the beginning of the research described in this chapter, I had two options to develop a conceptual model of the domain expert for CIRCSIM-Tutor (v.3). (1) Start with the conceptual model of CIRCSIM-Tutor (v.2) and extend it to a point where it could overcome its weaknesses (see Section 3.6.1), or (2) Restructure this model right from scratch. After some research, I picked this second option. One of the major reasons for doing this was that I found the behavior of our tutors as they perform domain reasoning much more complex than I had anticipated. As a result patching the conceptual model of CIRCSIM-Tutor (v.2) to capture this complex behavior would have created a messy design. This will become quite obvious in the following sections as I describe the conceptual model of the domain expert of CIRCSIM-Tutor (v.3).

I have used interviewing techniques as the dominant method of developing general models of domain. Later I also coded keyboard-to-keyboard transcripts to analyze the behavior of the domain expert further. Section 5.5.1 describes the domain problem-solving behavior of our tutors.

6.10 Nature and a Use of the Domain Knowledge By the Tutor and the Student in Keyboard-to-Keyboard Sessions

One of the major purposes of this section is to describe the nature and the use of the domain knowledge in our tutoring experiments. This we will achieve by using a keyboard-to-keyboard transcript. Figure 6.15 shows selected excepts from a keyboard-to-keyboard session. The problem description used for the CV problem in this session was as follows.

> Mr. SAN is a patient whose cardiac pacemaker are dead. He wears an artificial pacemaker which is the sole determinant of his heart rate. Normally the pacemaker produces a heart rate of 70/min. However, a defect in the pacemaker unit has caused the rate to suddenly change to 120/min.

These excepts are from the DR and RR phases of the CV system. These are unedited except to correct typographical and spelling errors. An analysis of this transcript points to many interesting aspects of the domain knowledge. Some of these aspects are listed below. (Some of these aspects were known well before this research started. See Section 3.6.1 for more details.)

- The nature of the domain knowledge used by the tutor was qualitative rather than quantitative.
- The focus of discussion during problem solving was seven core physiology parameters (see Figure 3.4).
- Occasionally the student and the tutor used domain knowledge with a greater degree of detail that required more fundamental reasoning than simply reasoning about the core physiology parameters (e.g., at instances: K1-tu-44-2, K1-st-45-1, K1-tu-57-3, K1-st-58-1, K1-tu-61-1 in Figure 6.15).
- This detailed knowledge on the part of the student, was some times invoked when she had some difficulty in understanding (e.g., at instances: K1-st-45-1, K1-st-58-1).
- There is a definite pattern in the use of domain knowledge; the tutors use more and more detailed knowledge as students have more difficulty in solving problems. In other words there is a direct relationship between the use of detailed levels of knowledge and the degree of difficulty experienced by the student (e.g. at instances: K1-tu-44-2, K1-tu-57-3).
- Students sometimes seem to change *perspective* (viewpoint) in their reasoning as they encounter difficulties (e.g., at instances: K1-st-23-1, K1-st-69-1). We define a perspective of a model to mean the nature of the models' reasoning. Each perspective of a model provides a focus on alternative means for understanding a real world phenomena which a model is representing/modeling. We will use *viewpoint* as a synonym for perspective. This definition is consistent with the definitions used by White & Frederiksen (1990) and Stevens & Collins (1980).
- Reasoning that results from a change in the perspective deals with entities different from the physiology parameters. These entities are mostly anatomical in nature (e.g., at instances: K1-st-23-1, K1-tu-27, K1-st-69-1).
- Students seem to come back to the same perspective that was set (as goal perspective) by tutors (e.g., at instances: K1-st-28-1, K1-st-58-1).

- Tutors change the level of detail and perspective opportunistically or as a result of the student's change in level or perspective (e.g., at instances: K1-tu-27, K1-tu-29, K1-tu-44-2, K1-tu-53-5, K1-tu-57-3, K1-tu-70-1).
- Tutors opportunistically guide students from different perspectives or levels of knowledge back to the goal knowledge level consisting of core physiology parameters (e.g., at instances: K1-tu-49-2, K1-tu-59, K1-tu-70-1).

The above observations point to the knowledge requirements for our system. These also describe some of the key constraints in shaping the representation of the domain knowledge in our system. We now further elaborate on these observations as follows. This analysis also makes explicit the behavior of our tutors while tutoring in the domain, i.e., the way the pedagogy expert uses domain knowledge while interacting with the student.

.

.

.

K1-tu-16-2: Now, what variable do you think will be affected first and in what direction?

Tutor started dialog using the parametric qualitative causal viewpoint (see Section 6.11) at the top level (see Section 6.11). This is the default level of the tutor.

K1-st-17-1: CO increase

Student responded using the same level and perspective..

K1-tu-18-1: Excellent!

K1-tu-18-2: What variable is affected next and how?

K1-st-19-1: MAP increase

Figure 6.15 Selected Excerpts from a Keyboard-to-Keyboard Tutoring Session

.

.

K1-tu-22-1: What causes MAP to rise?

Tutor and student up to this point constantly have used the parametric qualitative viewpoint at the top level.

K1-st-23-1: Increased amount of blood coming into the right atrium.

At this instant student switched his perspective from the parametric to the anatomical viewpoint (see Section 6.12). It is interesting that at this instant student had a misconception regarding the value of MAP.

.

.

K1-tu-27-1: Well, if one of the earliest things to happen is that the CO goes up, the heart will be removing increased quantities of blood from the central venous compartment.

K1-tu-27-2: Why would there be an increased amount of blood coming into it?

Recognizing that the student has switched perspective, the tutor responded using he anatomical viewpoint.

K1-st-28-1: I guess there would not be increased RAP, but would the TPR increase because of the increased CO?

Student here switched back to the parametric viewpoint at the top level.

K1-tu-29-1: We are dealing now with the DR period.

K1-tu-29-2: That's before there are any neural, reflex changes.

K1-tu-29-3: The TPR is neurally controlled.

K1-tu-29-4: So what do you think would happen to it?

At this instant due to a change in the student's perspective, the tutor also switched back to the parametric viewpoint at the top level.

.

During this interval both the student and the tutor continued to use the parametric perspective at the top level.

Figure 6.15 Selected Excerpts from a Keyboard-to Keyboard Tutoring Session (continued)

.

.

K1-tu-44-2: And when pressure goes up, what structures are affected by the reflex and how?

Here the tutor recognized that the student needs help. As a result he guided the student to think at the intermediate level (see Section 6.11).

K1-st-45-1: The firing of the baroreceptors increases.

The student here followed the tutor, and responded using intermediate level.

.

.

K1-tu-49-2: What change in MAP is triggering this reflex?

K1-st-50-1: MAP increase

The tutor and the student returned to the top level of parametric viewpoint.

.

During this period the student and the tutor remained at the top level of the parametric viewpoint.

.

K1-tu-53-5: When MAP goes up what change occurs to the autonomic out flow to the arterioles?

The tutor here recognized the student's difficulty and switched his viewpoint from parametric to anatomical.

.

The student and the tutor, during this period, continued to use anatomical perspective.

.

K1-tu-57-3: What happens to their (sympathetic vasoconstrictor nerves) firing rate in this situation.

The tutor here switched to the deep level of the parametric viewpoint.

Figure 6.15 Selected Excerpts from a Keyboard-to Keyboard Tutoring Session (continued)

K1-st-58-1: Decreased firing.

> *The student, here, switched from the anatomical to the parametric viewpoint and responded at the deep level (see Section 6.11).*

K1-tu-59-1: Right!

K1-tu-59-2: That's why TPR goes down.

> *The tutor continued to use the parametric viewpoint and brought the student again to the top level by summarizing results.*

.

.

K1-tu-61-1: Think again sympathetic firing is being decreased.

> *The tutor, here, used a hint from the deep level (i.e. momentarily switched from top to deep level).*

.

.

K1-tu-68-2: But why?

K1-st-69-1: To decrease the amount of blood being pumped into the arteries.

> *The student, here, switched perspective from the parametric to the anatomical viewpoint. He is still in the grasp of a misconception.*

K1-tu-70-1: Right , the object is to decrease CO.

> *The tutor converted the student's response from the anatomical viewpoint to the top level of the parametric perspective. (Since the top level in the parametric viewpoint is the default level for the tutor).*

.

> *The tutor and the student continued to use the parametric viewpoint at the top level until the end of this session.*

.

.

.

Figure 6.15 Selected Excerpts from a Keyboard-to Keyboard Tutoring Session (continued)

The type of domain knowledge used predominantly by tutors and students is qualitative and causal in nature. This is, of course, an immediate consequence of the nature of the problem and the nature of the requested solution (see Chapter III). However, there are instances in which reasoning must be based on the absolute value of some parameter. When some parameter must have a value greater than some threshold value in order to cause a change in another parameter, we call this a *conditional* relationship. This kind of relationship is different from the usual relationships, which invariably cause a qualitative change in the variable(s) they affect.

Most of the physiology reasoning needed to solve CV problems centers around the core parameters of the CV system. Tutors in all 45 recorded sessions tried to use and encourage students to acquire reasoning skills to solve CV problems, using the core parameters. The core parameters form a model of the CV system that is sufficient to solve many CV problems. We call this model the *minimal concept map* (see Figure 6.14 (a)), because no simpler model than this can correctly simulate the behavior of the CV system. Hence, one of the main objectives of CIRCSIM-Tutor is to help students to acquire the minimal concept map as their mental model to solve CV problems. The minimal concept map is even simpler than the concept maps used in CIRCSIM-Tutor (v.0) and CIRCSIM-Tutor (v.2).

When students have misconceptions they seem to reason using more detailed knowledge than in the minimal concept map. The tutors also seem to use more detailed knowledge if they are trying to remedy a student misconception or if it is opportune from their point-of-view. This "deeper" reasoning elaborates the causal relationships between the core parameters and introduces new parameters, which serve as intermediate steps in the causal links between the core parameters.

Our empirical studies also have confirmed that there are definite, progressively increasing levels of knowledge that students and tutors traverse while solving problems (see Section 6.14). Each more detailed level elaborates the level immediately above it.

Tutors use these detailed levels to *integrate* domain knowledge. This integration is achieved by using deep reasoning to support understanding at the top level. In our case this is the minimal concept map. Each successive level contains more physiology parameters and causal relationships between them than the level immediately above it. We have identified two elaborated levels and we call them the *intermediate-level* and the *deep-level* concept maps. The conceptual structures for these levels are explained in the next section.

The type of knowledge described above is comprised of a single type of domain concept: a physiology parameter. These domain entities are related via causal relationships between them. The three levels - minimal (top), intermediate, and deep level concept maps - form a progression of qualitative and causal models of the CV system. Each level is sufficiently rich in knowledge to simulate the behavior of the CV system. These three levels also constitute a perspective of the CV system. We call this perspective the *parametric (qualitative and causal) viewpoint*. This is the goal perspective of CIRCSIM-Tutor.

It is also interesting to note from Figure 6.15 that students and tutors sometimes invoke a quite different perspective of the CV system. This perspective is indeed used by students to support their reasoning for the goal perspective. We call this perspective the *anatomical (qualitative and causal) viewpoint,* because this perspective is composed of domain concepts that are the anatomical components of the CV system. Examples of these are: the heart, central nervous system, arterial system, and venous system. While reasoning with this perspective, students tend to use the behavioral aspects of the anatomical components of the CV system, e.g., "the heart will be removing increased quantities of blood from the central venous compartment," "the left ventricle is filled with blood." Most of the time, this perspective is invoked by students when they encounter difficulty in reasoning. This perspective is also invoked by tutors when students switch (to this) perspective or when tutors find it convenient to make a point.

It is interesting to note that classroom instruction teaches both of these perspectives to students but the parametric perspective is emphasized while problem solving. The *scope* of these two perspectives, as invoked by students/tutors, covers the full functionality of the CV system, i.e., each action of the CV system explainable via the parametric viewpoint has an equivalent explanation in the anatomical perspective. When the tutor finds an opportunity to remedy a misconception he switches between the perspectives or elaborates on the parametric viewpoint (see Section 6.14).

6.11 The Multi-Level Parametric Viewpoint

This section describes the building of conceptual structures for the parametric (qualitative and causal) viewpoint of CV system. Here we also show how the two modeling dimensions of *sufficiency* and *elaboration* shape these conceptual structures. Sufficiency is related to the amount of detail and elaboration (also called granularity) to the level of detail.

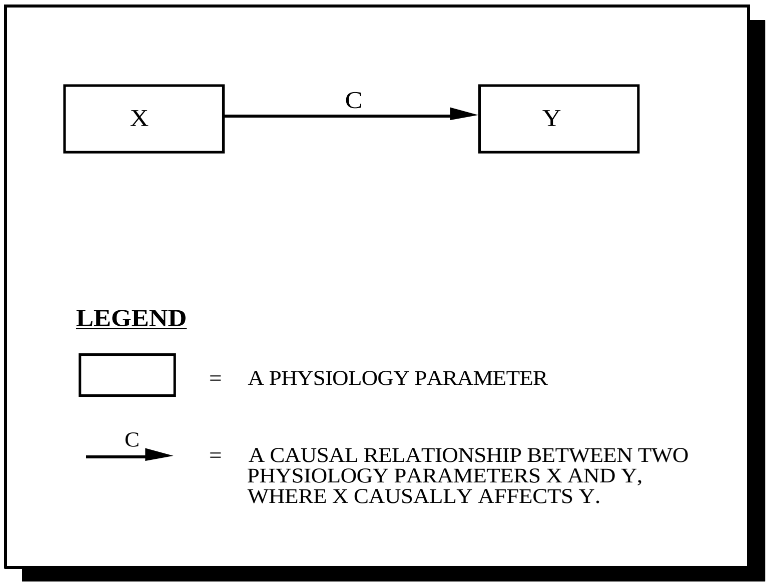

X
C
Y
LEGEND
= A PHYSIOLOGY PARAMETER
C
= A CAUSAL RELATIONSHIP BETWEEN TWO PHYSIOLOGY PARAMETERS X AND Y, WHERE X CAUSALLY AFFECTS Y.

Figure 6.16 A Schematic Representation of the Fundamental Entities in the Parametric Viewpoint

The fundamental domain concepts and relationships used in this perspective of the CV system are *physiology parameters* and *causal relationships* between these parameters, respectively. Schematically these two entities can be represented and related as shown in Figure 6.16.

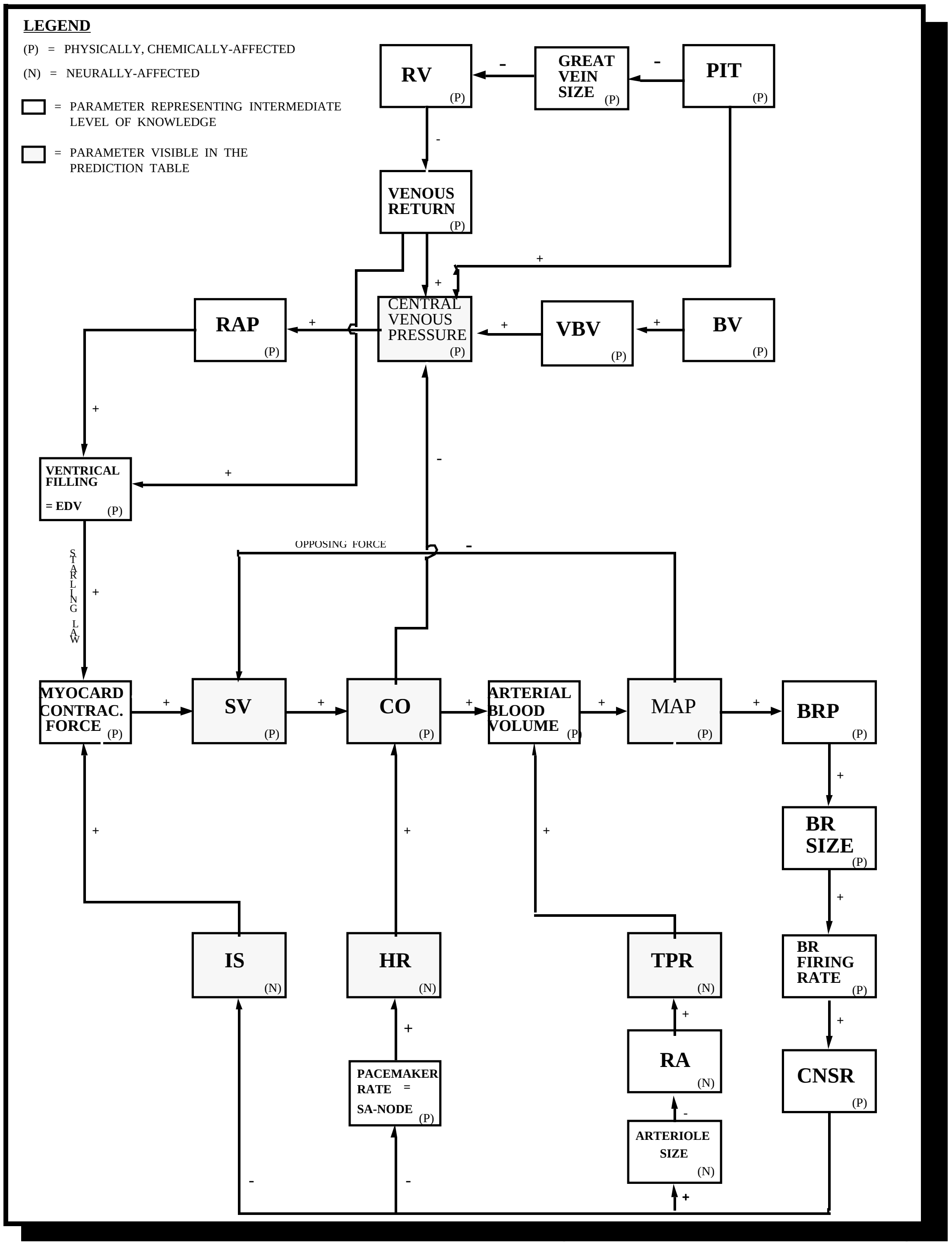

LEGEND
(P) = PHYSICALLY, CHEMICALLY-AFFECTED
(N) = NEURALLY-AFFECTED
= PARAMETER REPRESENTING INTERMEDIATE LEVEL OF KNOWLEDGE
= PARAMETER VISIBLE IN THE PREDICTION TABLE
RV (P)
GREAT VEIN SIZE (P)
PIT (P)
VENOUS RETURN (P)
RAP (P)
CENTRAL VENOUS PRESSURE (P)
VBV (P)
BV (P)
VENTRICAL FILLING = EDV (P)
STARLING LAW
OPPOSING FORCE
MYOCARD CONTRAC. FORCE (P)
SV (P)
CO (P)
ARTERIAL BLOOD VOLUME (P)
MAP (P)
BRP (P)
BR SIZE (P)
BR FIRING RATE (P)
CNSR (P)
IS (N)
HR (N)
TPR (N)
RA (N)
PACEMAKER RATE = SA-NODE (P)
ARTERIOLE SIZE (N)
+
-

Figure 6.17 The Intermediate Level Concept Map

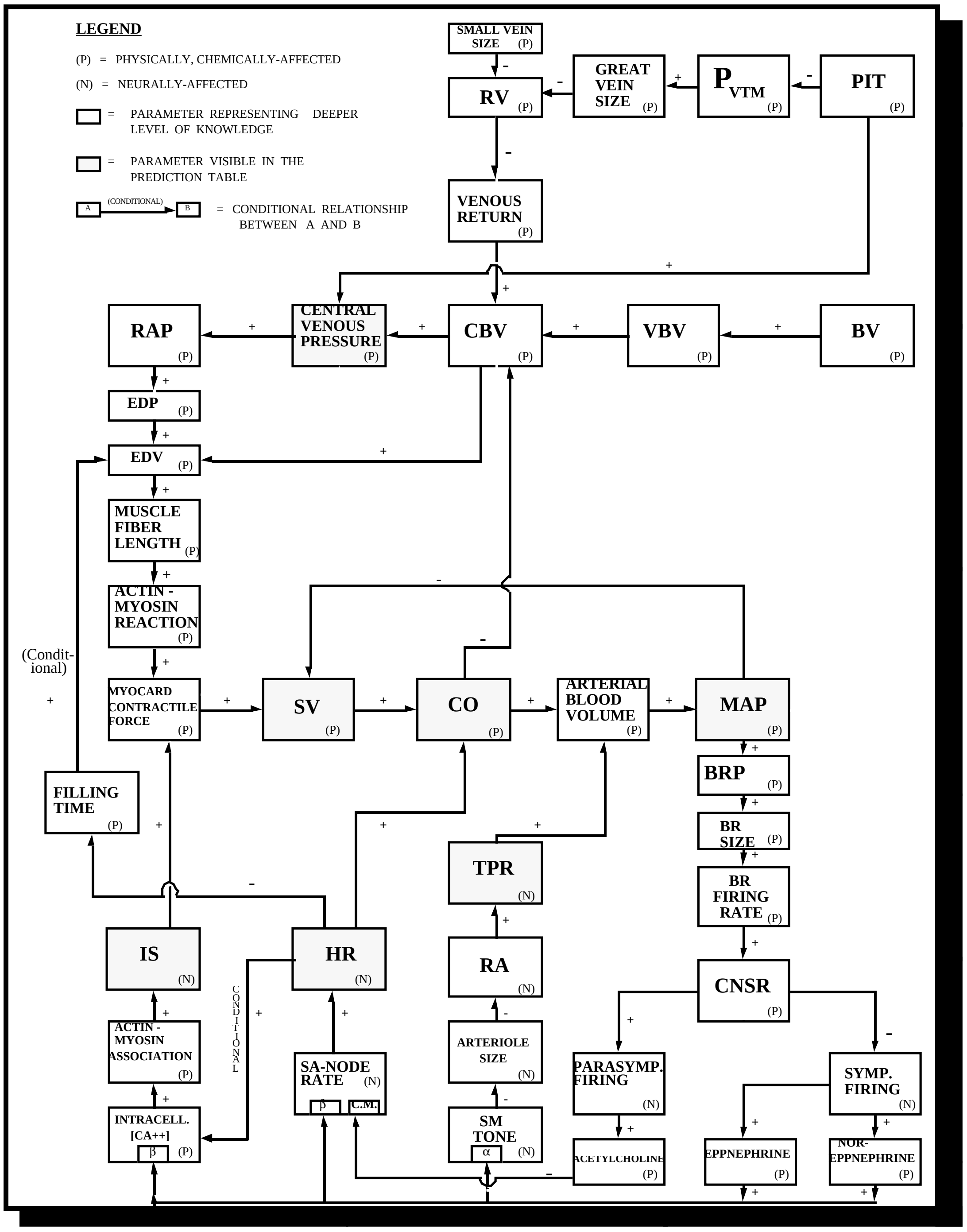


Figure 6.18 The Deep Level Concept Map

The building of the conceptual structures for the parametric viewpoint is mostly based on interviewing our tutors. The minimal concept map (see Figure 6.14 (a)) was obtained by modifying the concept map used in CIRCSIM-Tutor (v.2). The intermediate level (see Figure 6.17) and the deep level (see Figure 6.18) concept maps are new for this view of our domain.

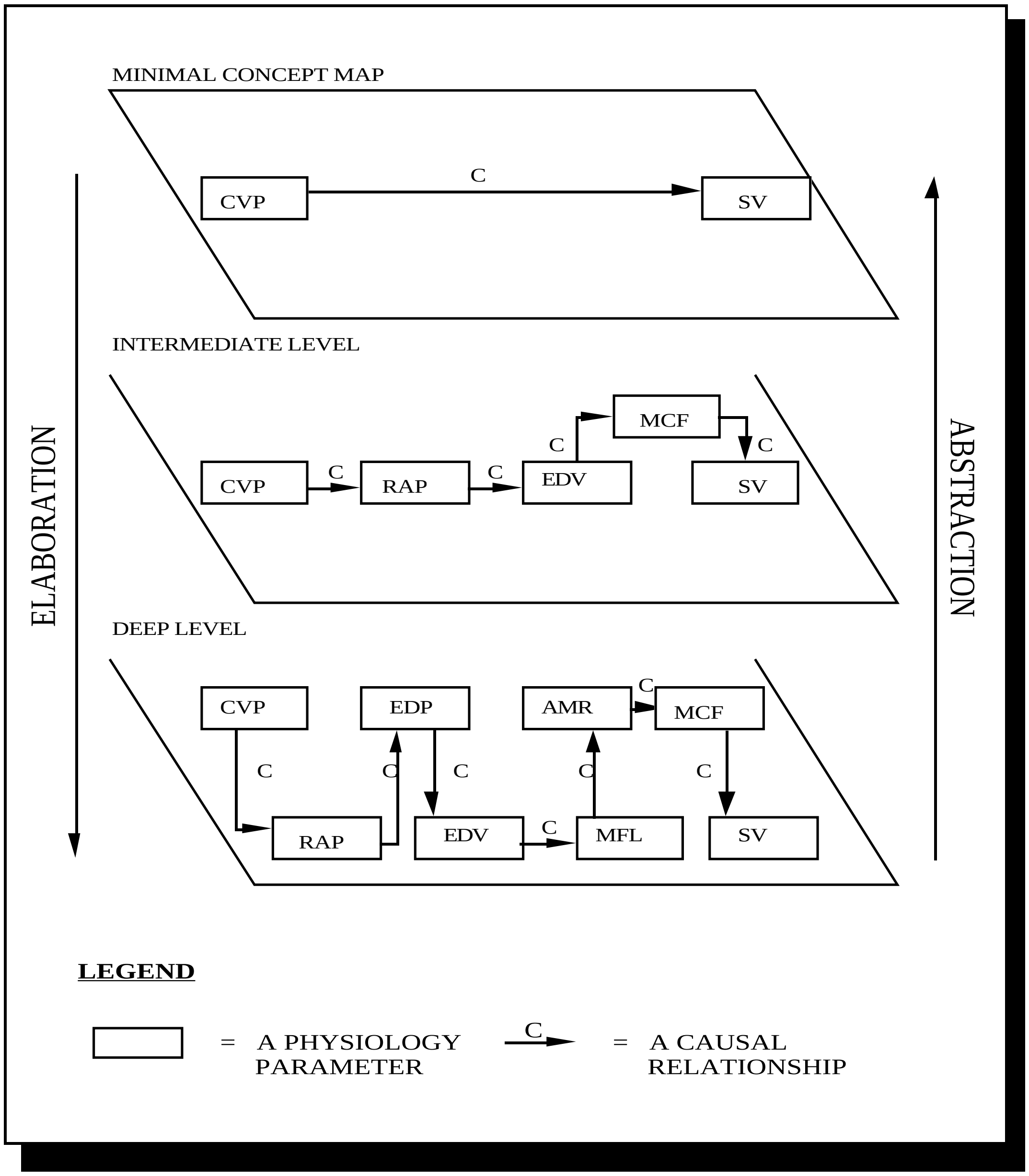


Figure 6.19 A Fragment from the Multi-Level Parametric Viewpoint of the CV System

The minimal concept map is the goal level of the system. It is hoped that students will be able to successfully solve problems at this level after using CIRCSIM-Tutor. The intermediate-level concept map elaborates the causal relationships that are represented at the minimal level, whereas the deep-level concept map elaborates further the causal relationships at the intermediate-level. These increasing levels of detail define the *elaboration* dimension for the parametric viewpoint (see Figure 6.19).

This configuration also allows tutors to help students abstract their reasoning from any of the elaborated levels to the top (the goal) level of the concept map. In other words, the existence of the core parameters at all levels helps tutors to switch their tutoring flexibly along the elaboration/abstraction dimension (see Figure 6.19).

The notion of elaborated levels of knowledge can be generalized to a large number of domains that can be modeled via qualitative and causal modeling processes (de Kleer & Brown, 1983). We have limited our identification to three levels but there can be additional levels in our domain or others. The actual number of levels that are employed depends mainly on pragmatic considerations determined by the educational context in which the ITS will be used and the expected knowledge state of the targeted students. As the elaboration levels increase from the most abstract (top) level, the reasoning tends toward reasoning from first principles in the domain.

We assume that our students possess information at all the levels. We hypothesize that the intermediate level of the concept map acts to provide a source of *cognitive continuity* in student reasoning. By cognitive continuity we mean *gradual* movement in reasoning from the most abstract to the most elaborated level. Reasoning only between the minimal and the deep level concept maps yields poor cognitive continuity and forces the students to jump *abruptly* between the levels. Cognitive continuity also provides the tutor with an opportunity to help the student *systematically* integrate this knowledge by gradually using the elaborated levels of the concept map.

The modeling dimension - sufficiency - also shaped the conceptual structures of Figures 6.14 (a), 6.17, and 6.18. The notion of sufficiency helped us to determine the knowledge details in each level of the concept map. Sufficiency also depends mainly on pragmatic considerations. The minimal concept map contains the core parameters of the CV system. The selection of these core parameters depends upon their importance in solving problems. Most of these parameters can be measured experimentally or calculated. We hoped that the limited number of these parameters at the top level allows students to predict the behavior of the CV system with less cognitive strain. We believe that there is a high probability that medical students will retain this simplified casual model to solve real life medical problems in their professional life.

The sufficiency of the deep level is determined mainly by the amount of physiology knowledge that students are expected to learn from class room lectures. This knowledge may be required to explain causal relationships or correct misconceptions during tutoring. The contents of the intermediate level were determined by the need to tutor student errors/misconceptions at a level below the surface; the deep level elaborates this one level further to give us another chance at tutoring.

6.12 Anatomical Perspective of the CV System

This section describes the conceptual structures that make up the anatomical perspective of the CV system. This perspective, as explained in Section 6.10, has been used by our tutors in the keyboard-to-keyboard sessions and is used to support reasoning from the parametric viewpoint of the CV system. The fundamental domain concept used in this perspective is an anatomical entity, e.g., the heart, arteries, and veins. The following three steps could be used to build this perspective of the CV system.

1) development of a conceptual hierarchy defining the structural relationships between the anatomical concepts,

2) definition of the model of each anatomical concept used in the conceptual hierarchy.

3) combination of steps 1 and 2 to form a functional model of the CV system from an anatomical perspective. This process is described below.

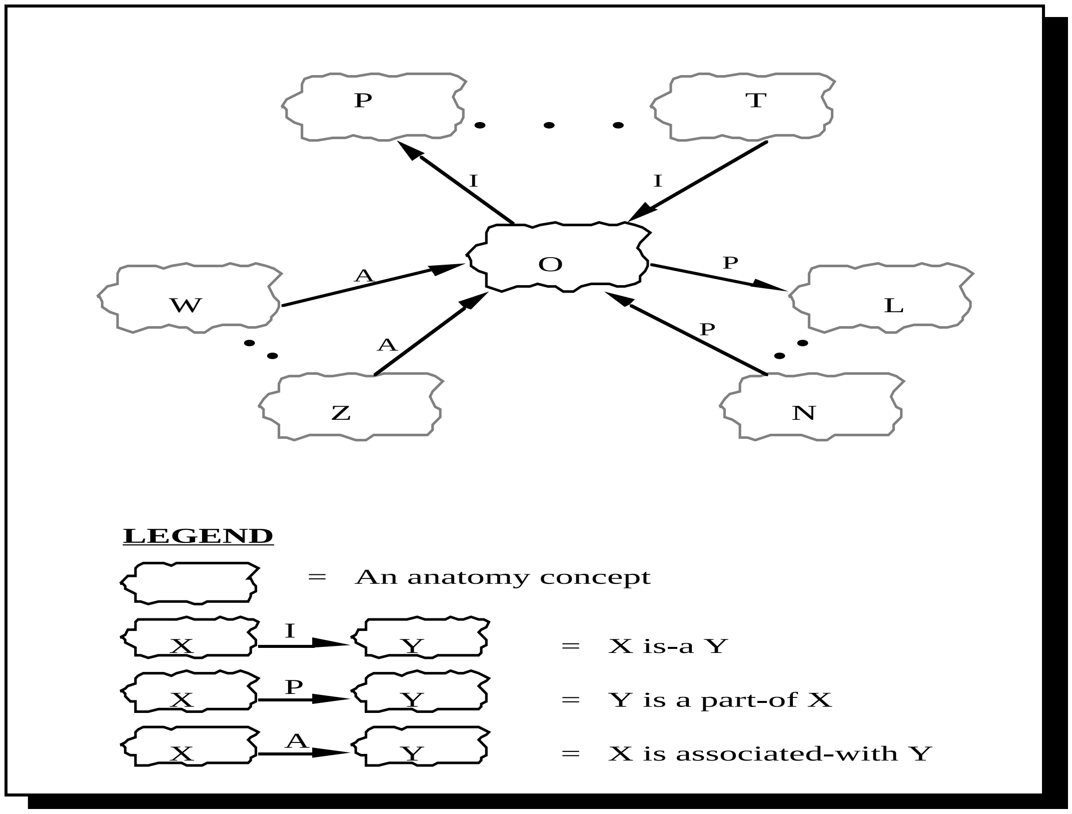


Figure 6.20 An Schematic Representation of the Generic Relationships between the Anatomical Concepts Used in the Anatomical Viewpoint of the CV System

We have developed a conceptual hierarchy, which relates anatomical concepts in two dimensions - *aggregation* and *generalization*. The aggregation dimension defines the part-whole relationships, while the generalization dimension defines the is-a relationships between the anatomical concepts of CV system. A generic schematic representation of an anatomical concept along these two dimensions is shown in Figure 6.20. The conceptual hierarchy was built by first identifying all anatomical concepts used in teaching cardiovascular physiology by interviewing our tutors. The next step was to identify the relationships between the resulting set of anatomical concepts as shown in Figure 6.20. This process yielded the conceptual hierarchy shown in Figure 6.21. This figure also shows the additional relationships between physiology parameters and perturbation concepts within this hierarchy.

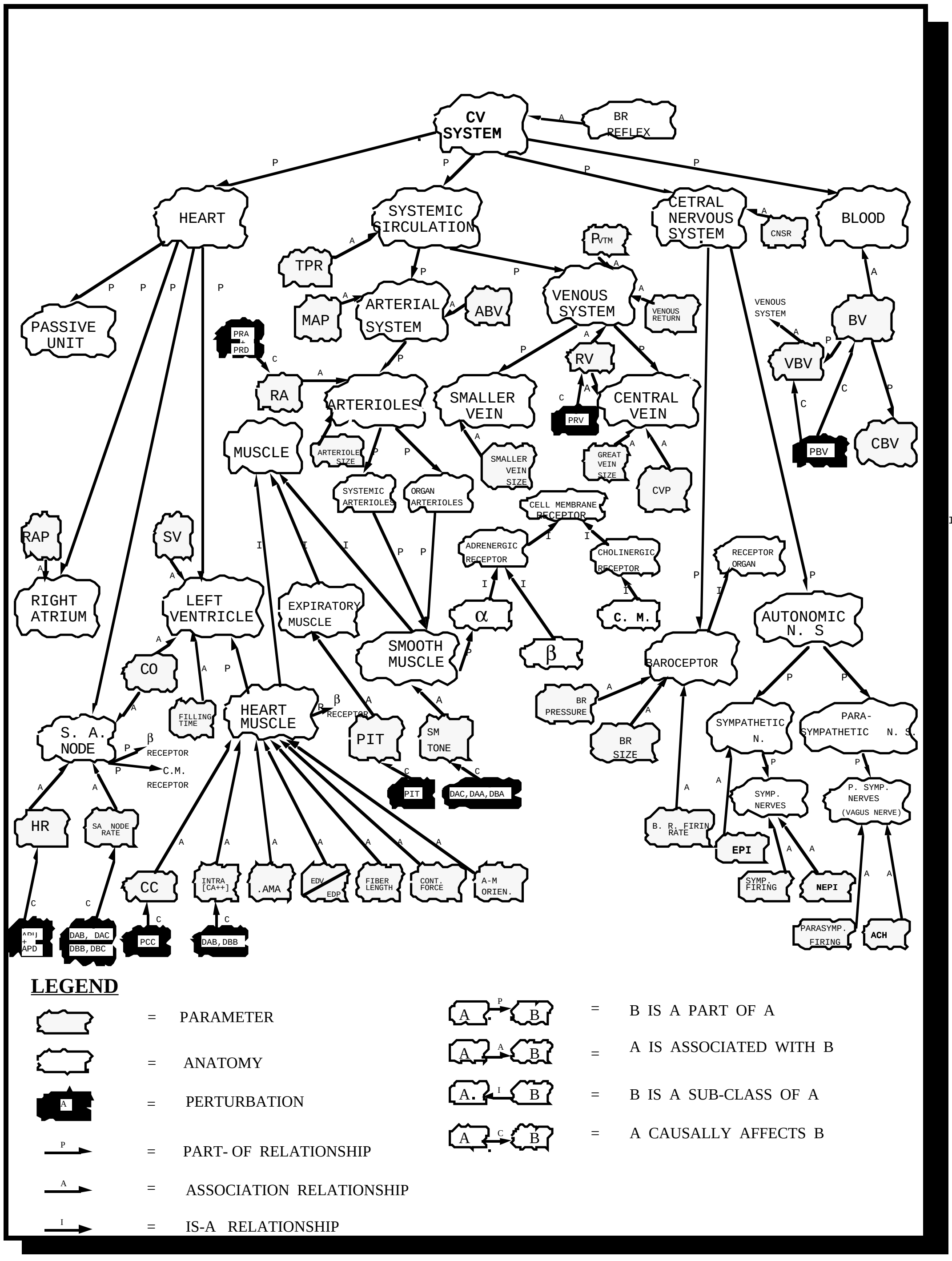


Figure 6.21 A Domain Concept Hierarchy from the Anatomical Viewpoint

These additional structures are explained more fully in the next section. The conceptual hierarchy of Figure 6.21 defines a structural model that represents the physical relationships between the anatomical components of the CV system. We assume that students possess this structural knowledge from classroom lectures in physiology and in anatomy.

The second step towards building the anatomical perspective is assigning behavior to each anatomical component of Figure 6.21. When students and tutors invoke this perspective they reason about the behavior of the anatomical components of the CV system in qualitative terms, e.g., the heart is pumping *more* blood.

The third step towards creating the anatomical perspective of the CV system is to integrate the structural model (see Figure 6.21) and the behavioral model of each anatomical concept to form a functional model.

All three steps define the knowledge structures that help to perform physiological reasoning from the anatomical perspective of the CV system. The functional model defines the higher level functions of the CV system, e.g., supplying blood to different parts of the body via the arterial system, returning blood to the heart through the venous system, the regulation of this process via the central nervous system.

These higher level functions are the behaviors of the major components of the CV system, which, in turn, is the sum of the behaviors of its constituent parts. Hence the domain concept hierarchy along with the behavioral model of anatomical components defines the functional decomposition of the behaviors of the CV system down the concept hierarchy.

Besides facilitating the functional decomposition of the *normal* behaviors of the CV system, the functional model of the CV system from the anatomical viewpoint can also be used to predict the behaviors of the CV system that result from perturbations acting on the system. This full notion of the functionality of the CV system through the anatomical perspective is explained in the next section, which deals with mapping between the identified perspectives. Further research is required to complete the last two steps to develop a full anatomical perspective for CIRCSIM-Tutor.

6.13 Mapping between the Perspectives

The parametric and anatomical perspectives parallel each other. The *scope* of the coverage of a domain phenomenon by these perspectives is the same. In other words both perspectives are capable of explaining a domain phenomenon at equivalent levels of detail. This diversity, in turn, lets students and tutors switch between perspectives at any time during learning and tutoring respectively. An example of the use of these perspectives to explain a physiological action is given in Figure 6.15 (K1-st-69-1 and K1-tu-70-1). In this example the student explains his reasoning using the anatomical perspective. The tutor in a response gives an equivalent explanation from the parametric perspective (since this is the tutor's goal perspective).

For a machine tutor to behave like our human tutors, it is necessary to have a full functional mapping between these two perspectives. Incorporation of this capability not only allows a machine tutor to switch flexibly and opportunistically between these perspectives but also helps it to "understand" the student's responses regardless of which perspective is used.

In CIRCSIM-Tutor a full functional mapping between perspectives can be obtained by augmenting the domain concepts defined in the conceptual structure of Figure 6.21. This section describes the way that we augmented this knowledge and some additional hierarchical knowledge structures that can assist the mapping processes between the two perspectives.

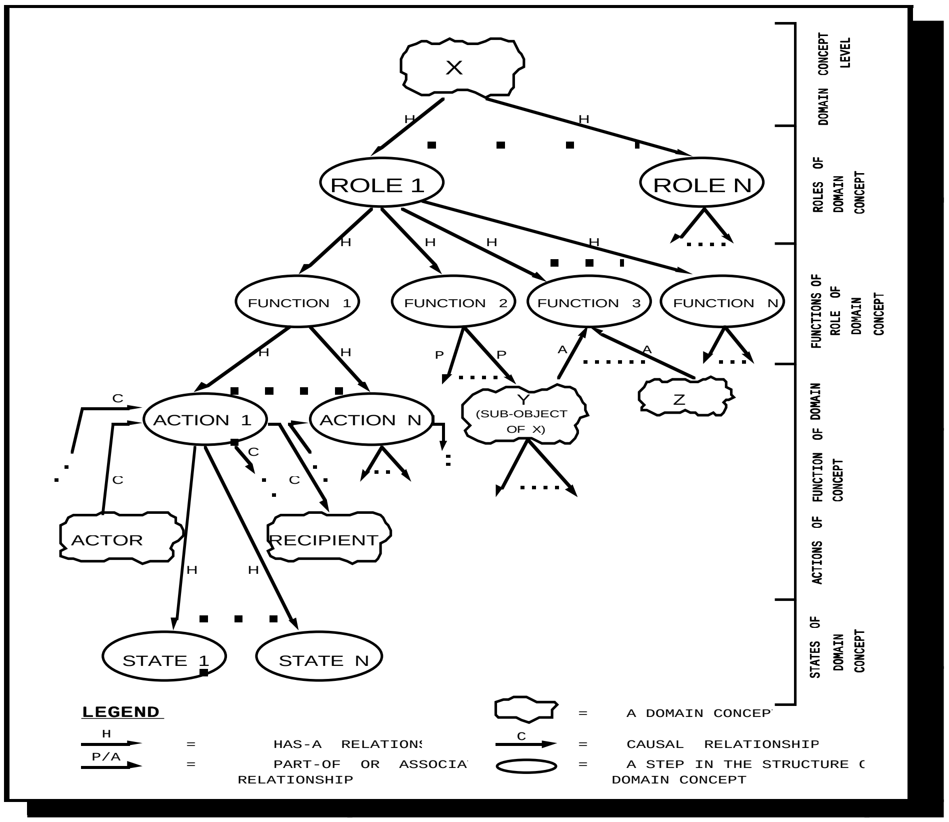


Figure 6.22 The Semantic Structure of a Domain Concept

Mapping between perspectives requires relating entities between perspectives at a functional level. The first step in the process of mapping between perspectives is to identify relationships between the entities of different perspectives. In physiology a parameter is associated with an anatomical object. For example, arterial resistance (RA - a CV parameter) is associated with the arterioles (an anatomy concept). Relationships between parameters and anatomical concepts are shown in Figure 6.21 by links labeled “A”. Perturbations cause physiology parameters to change value. We call the first parameter that is affected by a perturbation the *procedural* parameter. Figure 6.21 also shows the relationship between perturbations and their procedural parameters by links labeled “C.”

Besides these generic relationships we also need relationships that define functional relevance between the concepts of different perspectives. Hence the second step in mapping between the perspectives is to determine functional relevance between the concepts of different perspectives. The concept map provides a notion of the functioning of the CV system from the parametric viewpoint. Section 6.12 provides an equivalent notion of the functioning of the CV system from an anatomical viewpoint. Hence for mapping between perspectives, we need mechanisms that relate these two equivalent notions of the functioning of the CV system.

Figure 6.22 shows a semantic structure that can provide a functional bridge between the two. Each domain concept should be augmented with this structure. An explanation for this semantic structure follows. Each domain concept (X) can be viewed as having a number of roles, and each role can be accomplished by performing a number of functions. If a concept is a part of the aggregation hierarchy then some of its functions might lead to other domain concepts in that hierarchy (because Y in Figure 6.22 is a part of X and hence Function2 is the same as the totality of the roles of Y). Also if a concept has a number of associations with other domain concepts then some of its functions might lead to those associated domain concepts (for example, Z, in Figure 6.22 which is a different type of domain concept than X, is associated with X and hence Function3 is the same as one of the roles of Z). The remaining functions of that domain concept lead to various actions and each action is caused by some actor (another domain concept) and its effect is propagated to some recipient (another domain concept). These cause and effect phenomena lead that domain entity through various states of its existence. An example of a semantic structure for a perturbation concept is shown in Figure 6.23.

Augmenting each domain concept with this semantic structure provides us with multiple (but semantically equivalent) paths when a change is propagated in the CV system. Some of these paths have only physiology parameters (and hence constitute the parametric view of that change) and others have only anatomical objects (and form the anatomical perspective for that change). Changing a perspective for a causal action in the CV system is now equivalent to selecting an alternative (but equivalent) path. The domain independent semantic structure of Figure 6.22 can provide a prime mapping mechanism to switch between various perspectives. The full functional mapping between the parametric and the anatomical perspectives is not yet completely implemented in CIRCSIM-Tutor (v.3).

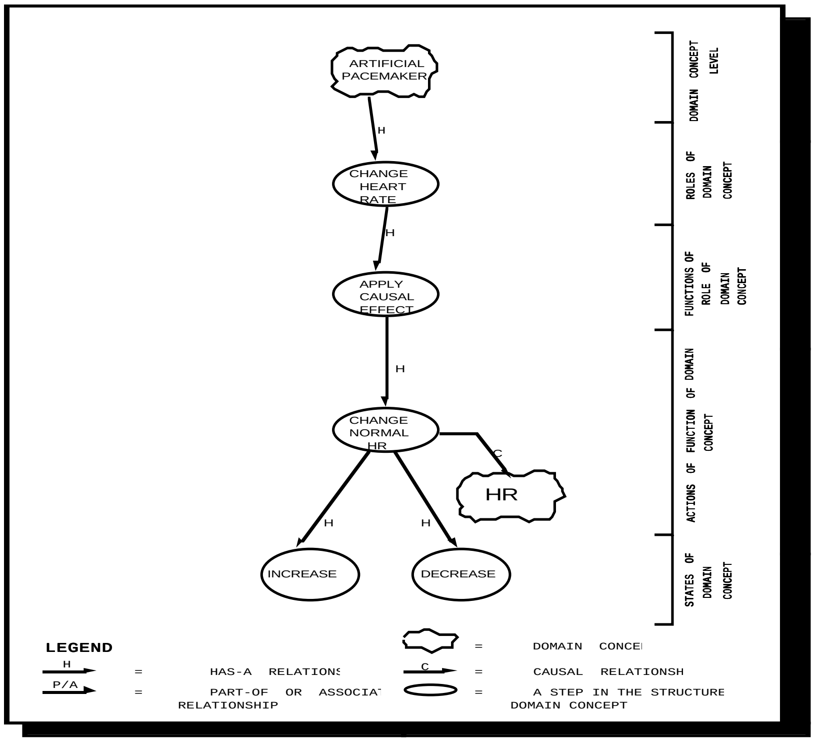


Figure 6.23 The Semantic Structure of a Perturbation

6.14 Model Switching Behavior of Our Tutors: Domain Models Viewed Through the Tutor's Eye

Most of the ideas to develop multiple models for the domain expert were created by interviewing our tutors. These models were also observed while analyzing transcripts of keyboard-to-keyboard sessions. Although physiology is taught at a very wide range of organizational levels (see Section 6.9), it would be interesting to know whether the multiple models, described above, were organized by the characteristics of the domain expert or whether it is the pedagogy expert that shaped the domain knowledge into this form. Also it would be interesting to know which models and transitions between them are most used by our tutors while performing in the keyboard-to-keyboard sessions.

Dr. Allen Rovick and I have coded 24 keyboard-to-keyboard sessions to get more insight about the model switching behavior of our tutor. Three different tutoring protocols have used in our 45 keyboard-to-keyboard sessions (see Section 5.5.2). We have encoded eight sessions from each set representing different tutoring protocols used in our tutoring experiments.

Although we do agree that there may be many more domain models besides those I have described above, we have limited ourselves to two perspectives and three levels (see Section 6.11 & 6.12) for CIRCSIM-Tutor (v.3). For coding purposes, we have categorized the domain knowledge in 24 keyboard-to-keyboard sessions according to these domain models. We have created 12 categories for transitions between models. Figure 6.24 shows these categories and a representative excerpt from a keyboard-to-keyboard session for each category. In Figure 6.24 MPT, MPI, and MPD refer to the top, intermediate, and deep level concept maps. MA is anatomical model of CV system. In this Figure MPT -> MPI represents a transition from the top to the intermediate level concept map. A schematic view of transitions between domain models is shown in Figure 6.25.

Figure 6.26 shows two examples from transcripts in which the tutor systematically guides the student between different parametric models to achieve his goals. In Figure 6.26 (a), at K1-tu-53-5, the tutor guides the student to switch from the top to the intermediate level of concept map (see Figure 6.17). The student at K1-st-54-1 did not provided the answer that the tutor wants, as a result he, at K1-tu-55-2, again guided the student to respond at the deep level of concept map (see Figure 6.18). At K1-st-56-1 the student supplied a wrong answer, as a result, the tutor at K1-tu-57-1 & 2 provided the correct answer for his question. Next at K1-tu-57-3 he wanted the student to provide a correct reply for the value of a CV parameter. After getting this reply the tutor at K1-tu-59-2 summarized this CV mechanism by making a transition from the deep to the top level of concept map.

- **MPT --> MPI**
  - K1-tu-53-5: When MAP goes up what change occurs to the autonomic outflow to the arterioles?
- **MPI --> MPD**
  - K39-tu-60-6: That one does physiologically mostly by changing the sympathetic stimulation o f the ventricle or by changing the circulating epinephrine levels.
- **MPT --> MPD**
  - K39-tu-120-1: When you said that TPR D, you implied that the reflex decreased sympathetic activity.
- **MPD --> MPI**
  - K40-tu-46-2: Do you know of a way to make the heart beat more force fully without changing the fiber length?
- **MPI --> MPT**
  - K40-tu-54-5: And filling is changed by altering RAP or its related variables, EDP.
- **MPD --> MPT**
  - K39-tu-62-1: End diastolic fiber length as measured by end diastolic pressure or end diastolic volume.
  - K39-tu-62-2: Which variable in the predictions table represents preload?
- **MPT --> MA**
  - K39-tu-126-2: When CO D, the ventricle takes less blood out of the atrium and the central venous compartment.
- **MPI --> MA**
  - K42-tu-100-1: If I pump more blood per minute out of the heart (hence out of the venous compartment) ...
- **MPD --> MA**
  - K12-tu-47-2: Venous return means blood returning from the systemic circulation to the heart.
- **MA --> MPT**
  - K39-tu-18-3: When you give a transfusion, most of the blood will end up there.
  - K39-tu-18-4: Now what do you think will be the first variable affected?
- **MA --> MPI**
  - K11-tu-67-2: If the volume of blood in the central veins decreases, what would happen to central venous pressure?
- **MA --> MPD**
  - K2-tu-32-3: With increasing COs, blood is removed more rapidly from the central blood compartment leading to smaller volumes and pressures (including RAP) there.

Figure 6.24 Categories for Domain Model Transitions

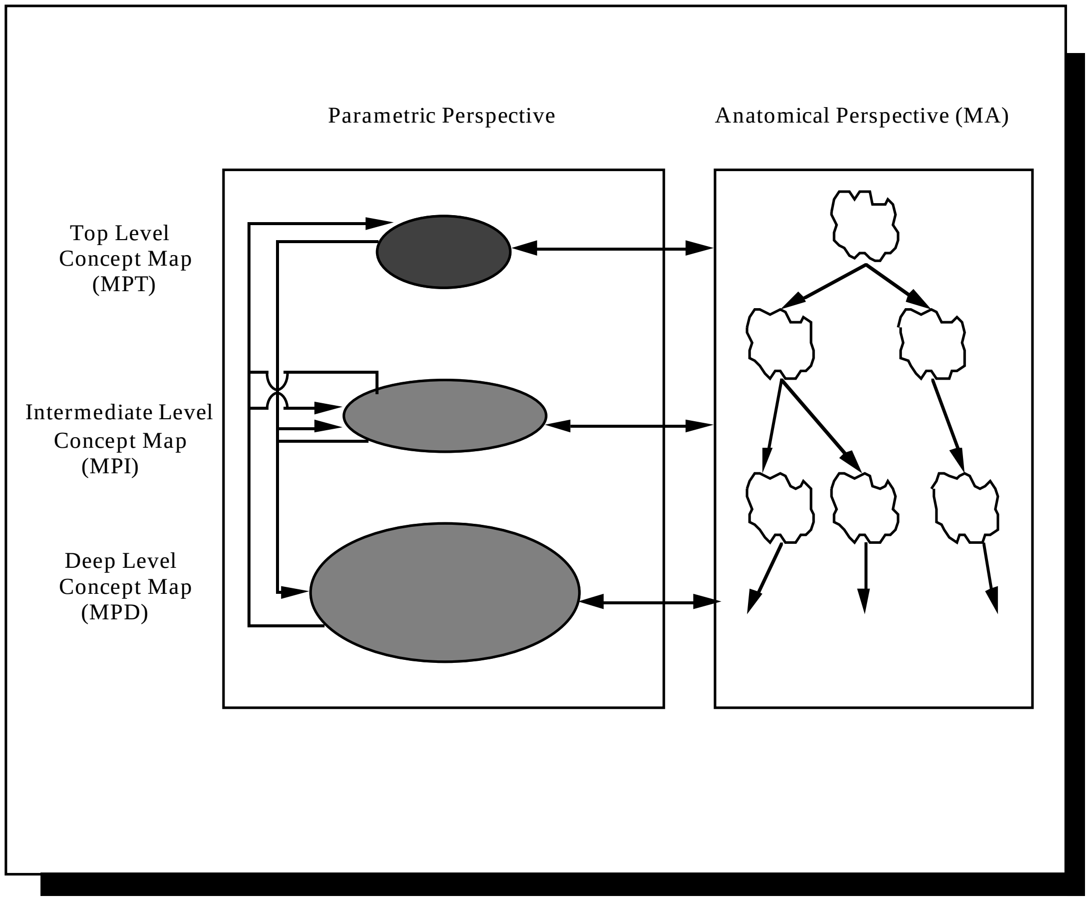

Parametric Perspective
Anatomical Perspective (MA)
Top Level Concept Map (MPT)
Intermediate Level Concept Map (MPI)
Deep Level Concept Map (MPD)

Figure 6.25 A Schematic View of Transitions Between Domain Models

Figure 6.26 (b) shows an excerpt in which the tutor makes a transition between the parametric to the anatomical view point. At K3-tu-53-1 the tutor initiated this transition from the parametric to the anatomical view point. When he thought that the student at K3-st-54-1 understood the domain knowledge under discussion, he made a transition back from the anatomical to the parametric viewpoint.

Figures 6.27, 6.28, and 6.29 show the result of encoding of keyboard-to-keyboard sessions. Some of the results are described as follows. Refer at Section 6.10 for more observations about this behavior.

K1-tu-53-5:	When MAP goes up what change occurs to the autonomic outflow to the arterioles?
K1-st-54-1:	The efferent outflow causes vasodilation of arterioles.
K1-tu-55-1:	That's right.
K1-tu-55-2:	What nerves are affected and in what way?
K1-st-56-1:	Sympathetic cholinergic nerves.
K1-tu-57-1:	no. They're not part of the baroceptor reflex.
K1-tu-57-2:	The sympathetic adrenergic vasoconstrictor nerves are.
K1-tu-57-3:	What happens to their firing rate in this situation.
K1-st-58-1:	Decreased firing.
K1-tu-59-1:	Right.
K1-tu-59-2:	That's why TPR goes down.

(a)

K3-tu-53-1:	The venous return may not change for a couple of minutes but what about the rate at which blood is being removed from the central blood compartment?
K3-st-54-1:	That rate would increase, perhaps increasing RAP???
K3-tu-55-1:	You are correct the rate of removal of blood would increase because CO is going up.
K3-tu-55-2:	But if you take blood out of the central venous compartment faster than it is returning, what happens to the central venous (I.E. RAP) pressure?

(b)

Figure 6.26 Tutor Systematically Guiding the Student to Make Transitions Between the Domain Models

If we compare the total number of transitions between domain models under the three tutoring protocols then there is a dramatic increase in sessions where Protocol 3 is used (see Figures 6.27, 6.28, and 6.29). It is difficult to pinpoint exactly the reason for this active behavior. It could be due to the nature of Protocol 3 or some other uncontrolled variable. Our tutors (like any other tutor) learn while tutoring. It is reasonable to assume that instead of the domain expert it is the pedagogy expert that is learning. One aspect of this learning deals with “How to tutor.” It is possible that with time and experience our tutors find model switching to be the most effective method of remediating misconceptions. It should be noted that before this research, our tutors were not consciously aware of their model switching behavior.

Model Transition	MPT -----> MPI	MPI -----> MPD	MPT -----> MPD	MPD -----> MPI	MPI -----> MPT	MPD -----> MPT	MPT -----> MA	MPI -----> MA	MPD -----> MA	MA -----> MPT	MA -----> MPI	MA -----> MPD	TOTAL
K1	1	1	1	0	0	2	1	0	0	1	0	0	7
K2	0	0	1	0	0	2	1	0	0	0	0	1	5
K3	1	1	4	1	1	3	1	0	0	1	0	0	13
K4	1	0	0	0	2	0	0	0	0	0	0	0	4
K5	4	0	0	0	4	0	0	0	0	0	0	0	8
K6	1	0	0	0	1	0	0	0	0	0	0	0	2
K7	3	1	1	0	3	2	2	0	0	1	1	0	14
K8	0	0	1	0	0	1	0	0	0	0	0	0	2
TOTAL	12	3	8	1	11	10	5	0	0	3	1	1	55

LEGEND

MX ---> MY = THE TUTOR HAS SWITCHED FROM MODEL X TO MODEL Y.
HERE X OR Y CAN BE PT, PI, PD, or A.

PT = PARAMETRIC MODEL (TOP LEVEL CONCEPT MAP)

PI = PARAMETRIC MODEL (INTERMEDIATE LEVEL CONCEPT MAP)

PD = PARAMETRIC MODEL (DEEP LEVEL CONCEPT MAP)

A = ANATOMICAL MODEL OF CV SYSTEM

Figure 6.27 Model Transitions in Sessions Using Protocol 1

Among models, our tutors make more transitions between the three levels of parametric viewpoint of CV system than between the parametric and the anatomical viewpoint. In most cases the number of transitions from the top to the intermediate and the deep level concept maps (i.e., MPT -> MPI & MPT -> MPD) is the same as the number of transitions from the intermediate and the deep levels to the top level concept map (i.e., MPD -> MPT & MPI -> MPT). One reason for this symmetry is that the top level concept map is the goal level of our tutors. For Protocol 3 the most used transitions are MPT -> MPI, MPT -> MPD, and MPD -> MPT. The number of transitions between the parametric and the anatomical viewpoint is almost negligible in the first two protocols. This behavior is significantly noticeable in Protocol 3. The most popular transition here is from the anatomical model to the top level concept map, i.e., MA -> MPT (see Figure 6.29).

Model Transition	MPT -----> MPI	MPI -----> MPD	MPT -----> MPD	MPD -----> MPI	MPI -----> MPT	MPD -----> MPT	MPT -----> MA	MPI -----> MA	MPD -----> MA	MA -----> MPT	MA -----> MPI	MA -----> MPD	TOTAL
K9	3	1	0	0	2	0	0	0	0	0	0	0	6
K10	4	1	1	1	2	2	1	0	0	0	0	1	13
K11	2	0	0	0	3	0	1	0	0	0	1	0	7
K12	3	2	2	1	2	2	0	0	1	0	0	1	14
K13	2	0	0	0	2	0	0	0	0	0	0	0	4
K14	1	0	0	0	1	0	0	0	0	0	0	0	2
K15	1	0	0	0	1	0	0	0	0	0	0	0	2
K16	2	0	0	0	1	0	0	0	0	0	0	0	3
TOTAL	18	4	3	2	14	4	2	0	1	0	1	2	51

LEGEND

MX ---> MY = THE TUTOR HAS SWITCHED FROM MODEL X TO MODEL Y.
HERE X OR Y CAN BE PT, PI, PD, or A.

PT = PARAMETRIC MODEL (TOP LEVEL CONCEPT MAP)

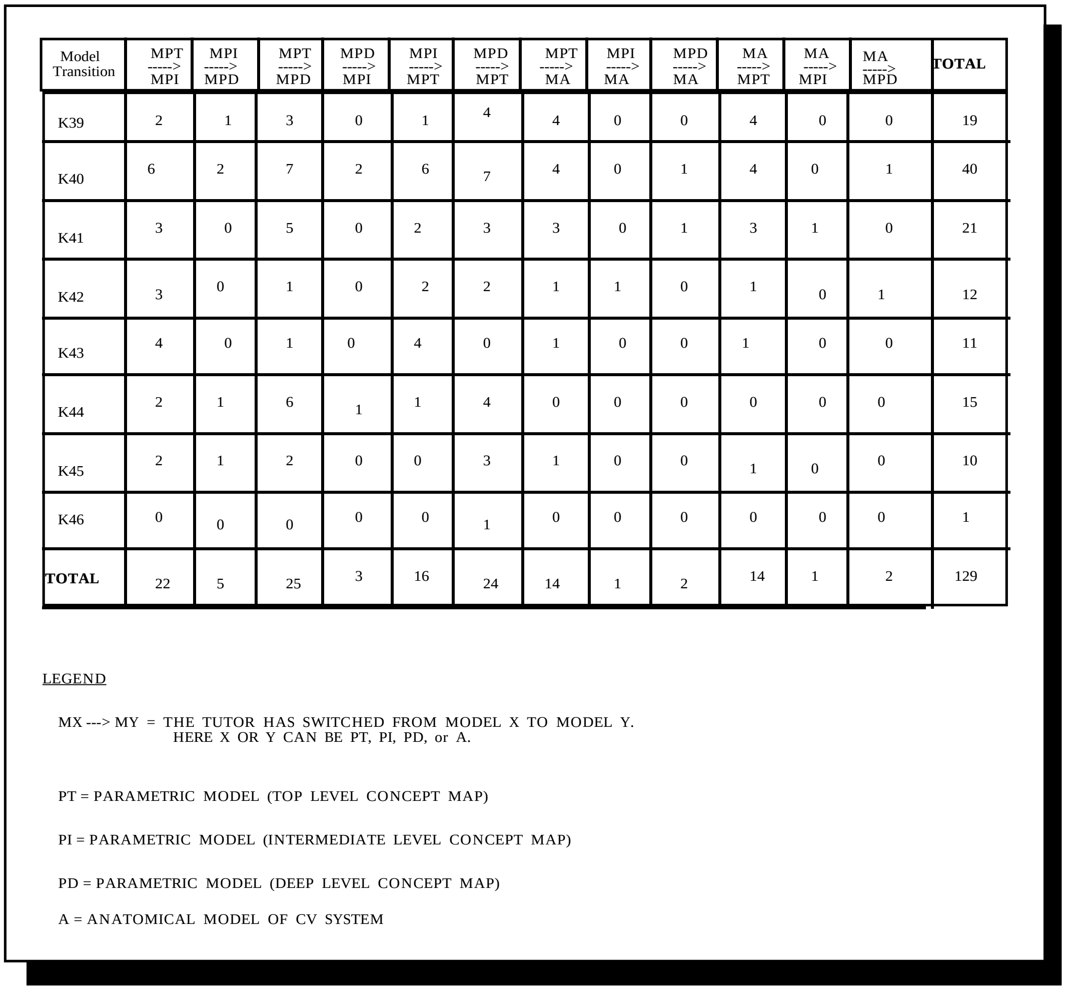
PI = PARAMETRIC MODEL (INTERMEDIATE LEVEL CONCEPT MAP)

PD = PARAMETRIC MODEL (DEEP LEVEL CONCEPT MAP)

A = ANATOMICAL MODEL OF CV SYSTEM

Figure 6.28 Model Transitions in Sessions Using Protocol 2

These results indicate that our tutors certainly make use of the models described in the previous sections. Their remediation method in Protocol 3 depends heavily on model transitions to tutor student's misconceptions.



Model Transition	MPT -----> MPI	MPI -----> MPD	MPT -----> MPD	MPD -----> MPI	MPI -----> MPT	MPD -----> MPT	MPT -----> MA	MPI -----> MA	MPD -----> MA	MA -----> MPT	MA -----> MPI	MA -----> MPD	TOTAL
K39	2	1	3	0	1	4	4	0	0	4	0	0	19
K40	6	2	7	2	6	7	4	0	1	4	0	1	40
K41	3	0	5	0	2	3	3	0	1	3	1	0	21
K42	3	0	1	0	2	2	1	1	0	1	0	1	12
K43	4	0	1	0	4	0	1	0	0	1	0	0	11
K44	2	1	6	1	1	4	0	0	0	0	0	0	15
K45	2	1	2	0	0	3	1	0	0	1	0	0	10
K46	0	0	0	0	0	1	0	0	0	0	0	0	1
TOTAL	22	5	25	3	16	24	14	1	2	14	1	2	129

LEGEND

MX ---> MY = THE TUTOR HAS SWITCHED FROM MODEL X TO MODEL Y.
HERE X OR Y CAN BE PT, PI, PD, or A.

PT = PARAMETRIC MODEL (TOP LEVEL CONCEPT MAP)

PI = PARAMETRIC MODEL (INTERMEDIATE LEVEL CONCEPT MAP)

PD = PARAMETRIC MODEL (DEEP LEVEL CONCEPT MAP)

A = ANATOMICAL MODEL OF CV SYSTEM

Figure 6.29 Model Transitions in Sessions Using Protocol 3

6.15 Integration Between Roles of the Tutor: Shared Inference Processes

As we have seen in the previous sections that the domain expert uses different models to perform reasoning in the domain. These models are the general models of the domain (see Section 6.8). These multiple models are also used by the pedagogy expert to perform tutoring in the domain. One of the functions that the pedagogy expert performs with these models deals with remediating misconceptions of the student.

In a way, these multiple domain models are shared resources that both experts use for their own purposes. We can still ask: are there other resources that both of these experts share to produce effective tutoring behavior? One type of domain knowledge that can also be a potential resource is the problem-solving algorithm. This algorithm, as we have seen in Section 5.5.1, is used by the domain expert to develop a situation specific model of the domain. In this and following sections we will describe a study that is primarily geared to understanding the nature of integration between the two experts. In this study we will concentrate on the problem-solving algorithm of the domain expert.

6.16 <u>Research Approach</u>

We assume that the domain expert and the pedagogy expert result from the *collaborative* behavior of various cognitive processes. The cognitive processes that support the roles of the domain expert and the pedagogy expert are denoted by "Ds" and "Ts" respectively.

One of the traditionally successful methods of performing cognitive analysis is protocol analysis (Ericsson & Simon, 1993). Analyzing only the transcripts of tutoring sessions will not solve our problem because in the transcripts the tutor's protocol is a result of the interaction between the D and the T types of cognitive process and therefore it is difficult to observe this interaction, clearly. It would be extremely helpful if we could observe the tutor performing one of the two roles in isolation and then compare it with the integrated behavior in the tutor’s protocol of tutoring transcripts. Fortunately, in our case the tutor is also the domain expert, hence it is possible to observe the behavior of the tutor performing the domain task that he assigns to the student in the tutoring situation. The research approach we have taken consists of the following steps:

(i) Model the behavior of the tutor, solving domain problems, in isolation. This will delineate a set of cognitive processes (Ds).

(ii) Using a set of Ds (obtained from above step) and Ts (obtained from the research work reported in this thesis) analyze the tutor's protocol in the tutoring transcripts. This analysis, we hypothesized, would yield a clear interaction between D and T types of cognitive processes and hence between the two roles of the tutor.

6.17 Skilled Tutor As Domain Expert

In this section we will concentrate on the problem-solving behavior of the *domain expert*. Using various knowledge acquisition techniques we have identified a set of cognitive processes (Ds) which our tutors use while solving a task in the domain of cardiovascular physiology. These processes together form a knowledge structure that we call the *Inference Triangle*. This structure is used by the domain expert in a variety of ways to solve a problem in the domain. We have used the think-aloud method as a prime source to delineate cognitive processes (Ds), which together constitute the problem-solving behavior of the domain expert. We have conducted a set of think-aloud sessions. Section 4.6.3 describes the method used in these sessions.

6.17.1 The Inference Triangle: A Qualitative Causal Reasoning Tool Used by the Domain Expert. In this section we will describe various cognitive processes (Ds) used by the domain expert in solving CV problems. These processes were combined in different ways by the domain expert to form high level operations. A task structure for the problem-solving behavior of the domain expert is built out of a sequence of these high level operations. The prime technique for this analysis was protocol analysis (Ericsson & Simon, 1993). We have also used other knowledge acquisition techniques (e.g., interviews with the domain expert) to add details to this analysis.

We have identified three basic cognitive processes (Ds) used by the domain expert in solving a CV problem. These are defined as follows:

I) Collection. This is a function that is applied to a concept C and yields a set of concepts U. Each member of U will then have a relationship R with C. The nature of the relationship (R) between C and the members of U, in our domain, is *causal.* The type of concepts used in this function can be parameter, anatomy, or perturbation (Khuwaja et al., 1992). An example of use of this function by the domain expert is: “RAP is, of course, a determinant of SV ...”.

II) <u>Selection</u>. This is also a function. Input to this function is a concept C and a set of concepts U. Each member of U has a relationship R with C. The output of this function is a set of concepts V that is a subset of U. The nature of the relationship R, in our domain, is causal and the type of concept C can be parameter, anatomy, or perturbation. An example of use of this function by the domain expert is: “...this is a matter of prior knowledge, in this instance the predominating factor that influences SV happens to be RAP ...”.

There is a special case for each of above functions which we describe as follows. Each concept C has a state at each instant in time. This state can be represented by qualitative values, increase (+), decrease (-), or no change (0). In the special case the input to the collection function is a concept C and the output is a set of state values of C that it had back in time (e.g., in some phase of CV system). An example of this case is: “The original procedure was a beta blocker which decreases the HR, and we had a reflex effect through the parasympathetics on HR ...”. In the case of the selection function, the input is a set of state values for a concept C and the output is a state value of C, which is also a member of the set of input values to C.

III) <u>Inference-calculation</u>. This function takes two concepts C1 and C2 and a state value of C1 as input. Its output contain a state value of C2. C1 and C2 have a predefined relationship R between them. An example use of this function by the domain expert is: “...so I indicate in DR that mean arterial pressure has fallen”. A special case also exists for this function. In this case input is a state value for the concept C at time t1. Output of this function is a state value of C at time t2. An example use of this case is: “so I put a decrease here in TPR ... “.

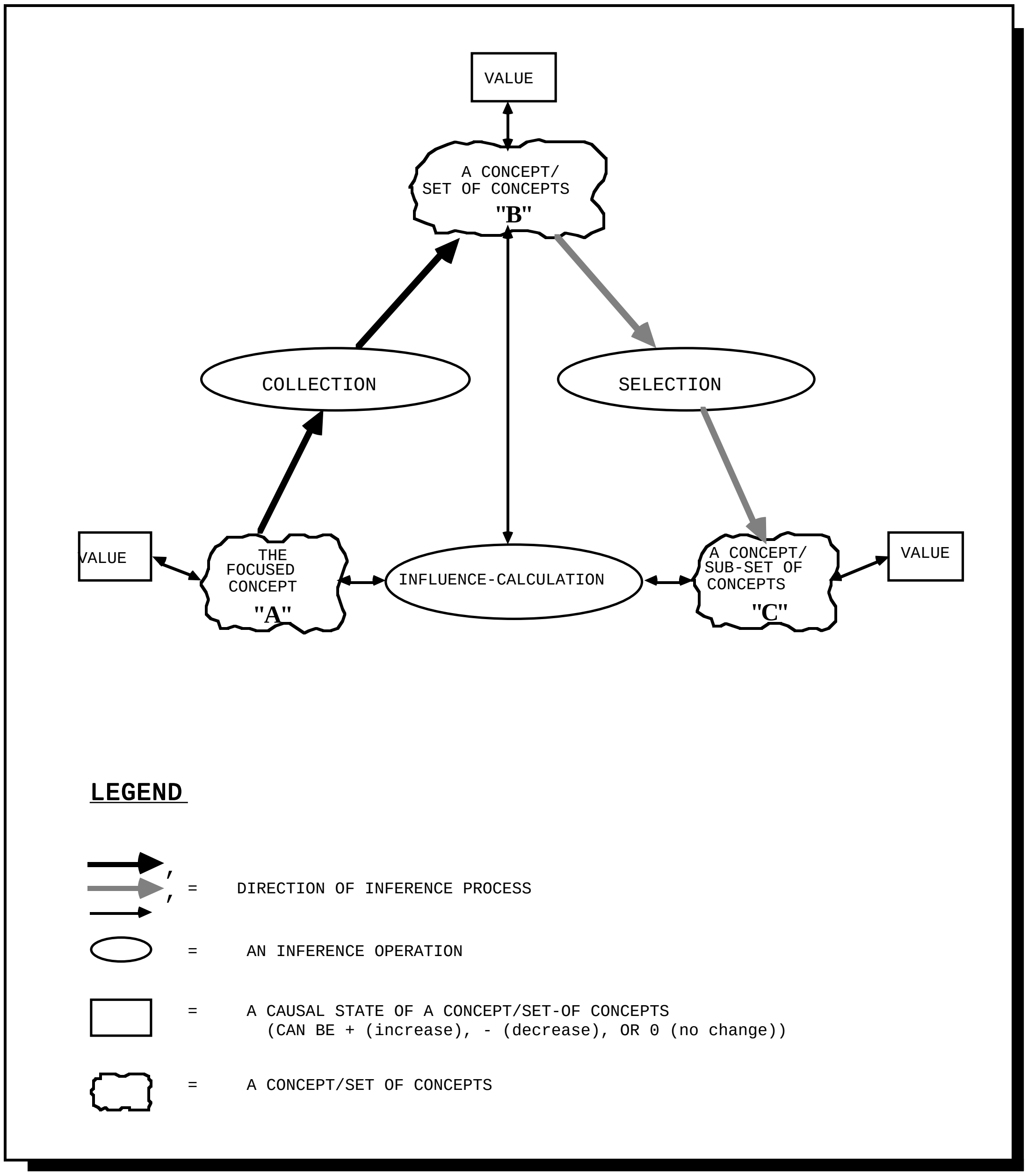

VALUE
A CONCEPT/ SET OF CONCEPTS "B"
COLLECTION
SELECTION
VALUE
THE FOCUSED CONCEPT "A"
INFLUENCE-CALCULATION
A CONCEPT/ SUB-SET OF CONCEPTS "C"
VALUE
VALUE
LEGEND
= DIRECTION OF INFERENCE PROCESS
= AN INFERENCE OPERATION
= A CAUSAL STATE OF A CONCEPT/SET-OF CONCEPTS (CAN BE + (increase), - (decrease), OR 0 (no change))
= A CONCEPT/SET OF CONCEPTS

Figure 6.30 The Inference Triangle

These three functions are interrelated and form a knowledge structure that we call the inference triangle (see Figure 6.30). There are some special characteristics of this triangle, which we describe below. Figure 6.30 can also be viewed as a directed graph. The domain expert traverses it to perform reasoning in the domain. A repeated use of this knowledge structure enables the domain expert to develop a situation-specific model from the domain model of CV system.

Traversing successfully from node A to node B, in this structure, requires the domain expert to use knowledge from the general model of the domain. A successful traversal from node B to node C requires the domain expert to consider situation-specific knowledge (e.g., “The procedure variable is to block beta adrenergic receptors ... and we know that when we put beta blocker in, what we are going to basically do is reduce the tonic activity of those particular tissues.”) in conjunction with the general model of CV system. A traversal between nodes A and C yields the prediction for the state value of the concept at either A or C (depending upon the situation at hand). A complete traversal of this knowledge structure yields a prediction for any CV parameter.

The domain expert traverses the inference triangle in a number of different ways while solving problems. In other words the domain expert uses these three cognitive processes in different sequences, as follows:

(a) collection ---> selection ---> influence-calculation

(b) collection + selection ---> influence-calculation

(c) collection + selection + influence-calculation

(d) collection + influence-calculation ---> selection

(e) collection ---> selection + influence-calculation

In case (a), the domain expert uses collection, selection, and then influence-calculation functions, in this sequence. But each of these functions can be used separately. In case (c), for example, all three functions are used at the same time, i.e., case (c) is a true *compilation* (Anderson, 1983) of (a). By compilation we mean that the domain expert has mastered these functions so that he can use all three as a *unit* rather than treating each as a separate entity. This behavior of the domain expert is not surprising and agrees with the findings in the literature on expert-novice differences (Chi et al., 1988). Examples of the use of each compiled form of these cognitive processes, by the domain expert, are shown in Figure 6.31.

These three cognitive processes together form three high level operations which the domain expert uses to perform the prediction task. These operations are defined as follows:

I) **<u>Spreading operation</u>.** This operation determines the *direction* of inference used by the domain expert. In this operation inferences are made from the cause to its effect (i.e., in the *forward* direction). For example, in an instance, the domain expert used this operation to predict the value of CVP from CBV by propagating the causal effect/influence. Here CBV is the source variable and CVP is the affected variable. Figure 6.32 schematically shows this operation by instantiating the inference triangle.

- collection + selection: "... and total peripheral resistance is a determinant of mean arterial pressure..."
- collection + selection + influence-calculation: "TPR is a neural variable and therefore it does not change in DR."
- collection + influence-calculation: "... and now I have the two determinants of SV moving in opposite directions ..."
- selection + influence-calculation: "... so it is going to be up in SS."

Figure 6.31 Examples For Compiled Cognitive Processes

II) **<u>Originating operation</u>**. This operation is the opposite of the spreading operation, i.e., here inferences are made from the effect towards its cause. Using this operation the domain expert predicted a variable by reasoning *backwards* from its source. An example use of this operation is shown in Figure 6.32.

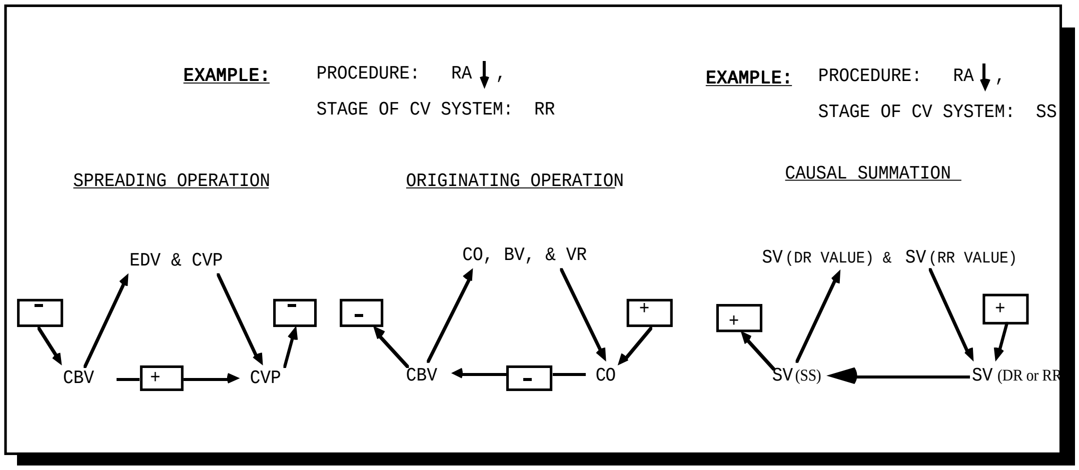


Figure 6.32 Examples of the Spreading, Originating, and Casual Summation Operations

Spreading

- collection -> selection -> influence-calculation
- collection + selection -> influence-calculation
- collection + selection + influence-calculation
- collection -> selection + influence-calculation

Originating

- collection -> selection -> influence-calculation
- collection + selection -> influence-calculation
- collection + selection + influence-calculation
- collection + influence-calculation -> selection
- collection -> selection + influence-calculation

Causal summation

- collection -> selection -> influence-calculation
- collection + selection -> influence-calculation
- collection + selection + influence-calculation
- collection + influence-calculation -> selection
- collection -> selection + influence-calculation

Figure 6.33 Possible Sequences of Cognitive Processes in the Spreading, Originating, and Causal Summation Operations

III) <u>Causal summation</u>. Like the spreading and the originating operations this operation is also composed of all three cognitive processes. This operation enables the prediction of a variable, say X, at time instance T_n by causally summing (this requires both general and situation-specific knowledge of the domain) the state values of X at time instances T_m and T_o. Figure 6.32 shows an example use of this operation by the domain expert.

All three of these operations are fundamental to the solution of a CV problem. From Figure 6.32 it is clear that these operations are obtained by using the inference triangle. The domain expert traverses this triangle in variety of ways for each operation. Figure 6.33 lists all possible sequences of cognitive processes for each operation. A repeated use of these operations/inference triangle by the domain expert created a task structure, which is shown in Figure 6.34 in the form of a flow chart. Compare this flow chart with Figure 5.2. The problem description requires the prediction of each of seven CV variables in three stages (Michael et al., 1992). This is accomplished by the domain expert using the inference triangle repeatedly. The selection of the next variable for prediction, depended, most of the time, upon the result of using the inference triangle. For example, in a think-aloud session, the domain expert predicted CO after applying inference triangle on HR. The very next variable predicted was MAP, as a result of applying the inference triangle to CO. Hence the inference triangle not only enabled the domain expert to predict CV variables but it also provided a means for *selecting* the next variable for prediction, during problem-solving. The task structure of Figure 6.34 is fixed, that is, the domain expert always used it to solve all CV problems in these sessions. This result is not surprising and agrees with the findings of Wielinga and Breuker (1990). The representation of this task structure has been simplified here. An other view of it can be found in Section 5.5.1.

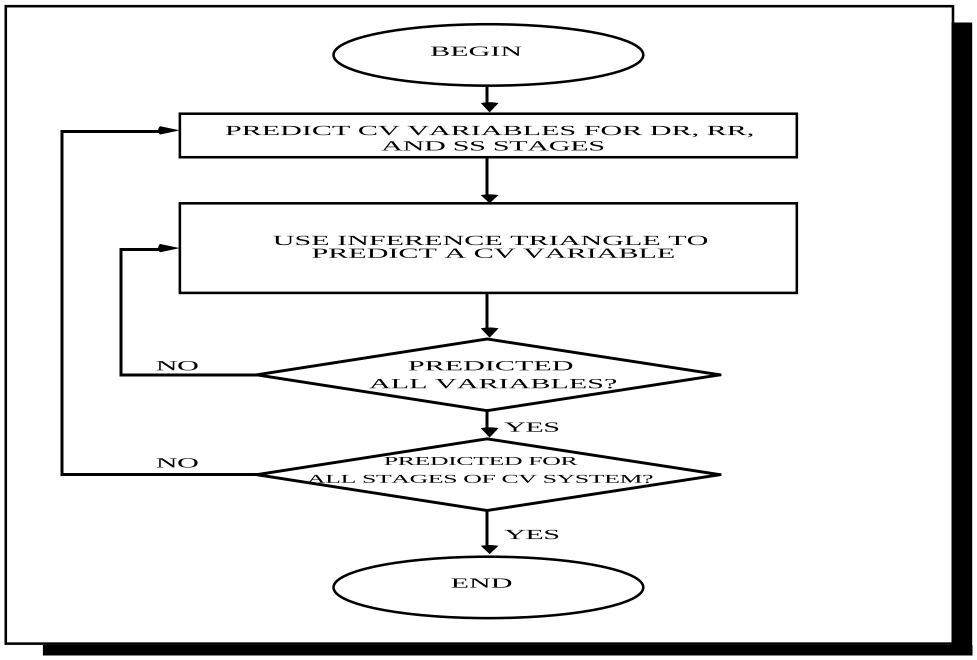

BEGIN
PREDICT CV VARIABLES FOR DR, RR, AND SS STAGES
USE INFERENCE TRIANGLE TO PREDICT A CV VARIABLE
NO
PREDICTED ALL VARIABLES?
YES
NO
PREDICTED FOR ALL STAGES OF CV SYSTEM?
YES
END

Figure 6.34 Flow Chart Representing the Task Structure Used by the Domain Expert

6.17.2 <u>Think-Aloud Sessions: Results</u>. The taxonomy of the domain expert's actions (cognitive processes and inference operations) described in the previous sections was used as the basis for a coding scheme on protocols in these think-aloud sessions. This was done to achieve more insight into the domain expert's reasoning process. All of the cognitive processes, along with their different compiled forms, were counted in these sessions. Each cognitive process was also categorized with reference to the inference operation (spreading, originating, and causal summation) that contained it. These results are presented in Figure 6.35. Out of 190 instances of cognitive processes and their compiled instances, 102 were used as a part of the spreading operation, 51 as the originating operation and 37 as the causal calculation. The spreading operation, by definition, is typical of a prediction oriented task, hence this result is not surprising. In the protocols of these sessions, most of the time an originating operation was used when the CV variable in focus had multiple interacting causal influences from more than one CV variable. The prediction of such a variable requires the selection of the actual (domain) source of causal influence on it. The causal calculation operation is only relevant in SS. In a CV problem, only 7 out of 21 predictions are required for this phase of the CV system. This explains why this inferencing operation was used least often.

	COLLECTION	SELECTION	INFLUENCE-CALCULATION	COLLECTION + SELECTION	COLLECTION + INFLUENCE-CALCULATION	SELECTION + INFLUENCE-CALCULATION	COLLECTION + SELECTION + INFLUENCE-CALCULATION	TOTAL
SPREADING OPERATION	5	5	21	20	4	2	45	102
ORIGINATING OPERATION	2	9	10	4	16	0	10	51
CAUSAL CALCULATION	6	0	13	7	0	5	6	37
TOTAL	13	14	44	31	20	7	61	190

Figure 6.35 Think-Aloud Sessions - Results

- **A Highly Articulated Instance:**
“ ... so that means the RAP is going to go up and that would have an effect on SV, causing SV to rise. The fall in MAP also causes the SV to rise because that is the decrease in after load and since the effect of right arterial pressure changing simultaneously with the change in CC, will dominate then the SV goes up, ... It is going up for two reason: the decrease in, excuse me, the increase in RAP and the decrease in Map. And the opposing effect is the decrease in MAP. And the opposing effect is the decrease in CC. So the two things, which include the change in RAP dominate ... “

- **A Least Articulate Instance:**
“I am going down the line ... decrease SV ... “

Figure 6.36 Examples of Different Levels of Articulation

In these sessions, the domain expert expressed his reasoning aloud at many *levels* of articulation. In the most articulated instances the domain expert took as small steps as possible while traversing the inference triangle. On the contrary in the least articulated instances (i.e., in the highly compiled mode) the domain expert combined (compiled) the cognitive processes together as much as possible while using the inference triangle. An example of these two extremities in predicting the value for SV is shown in Figure 6.36. The compiled cognitive process, collection + selection + influence-calculation is the action most used by the domain expert.

6.18 Skilled Tutor As Expert in the Domain and in the Process of Tutoring

In this section we will analyze the tutor's protocol in the transcripts of our tutoring sessions. Our purpose for this analysis is to observe the *interaction* between the two roles (domain expert and pedagogy expert) of the skilled human tutor.

6.18.1 Nature of *Integration* between Two Roles of the Skilled Human Tutor. In this section we will analyze the *nature of integration* between the two roles (domain expert and pedagogy expert) of the skilled human tutor. For this we will use a traditionally successful method of protocol analysis (Ericsson & Simon, 1993). In Section 6.17 we have developed a problem-solving model of the domain expert. Just using this model as a basis for a coding scheme on protocols of the tutor in the tutoring sessions will not satisfy the goals of this study. We need, also, a model of the pedagogy expert and we need to use it as part of the coding scheme, together with the model of the domain expert, on protocols of the skilled tutor in the tutoring sessions. The model of the pedagogy expert developed in this research is quite extensive. We are convinced that full use of this extensive model along with the model of the domain expert as a coding scheme will make the process of protocol coding and analysis quite complicated. We therefore will use only a few of the actions of the pedagogy expert, as part of the skilled tutor's model, for protocol coding and analysis. The criteria for selecting the pedagogy expert's action is extensive *usage* and its *importance* in the process of tutoring. The pedagogy expert's actions selected for our purposes are: (asking) *question*, (giving) *explanation*, and *summarizing* domain knowledge. These actions are domain *independent* and also used in most tutoring methods, e.g., coaching (Breuker, 1990). As mentioned above, our tutors use the *Socratic* method of tutoring. In this method the tutor constantly asks questions to probe and remedy student misconceptions. Here the tutor tries, as much as possible, not to convey the domain knowledge in an expository way. The tutor's major strategy is to use a sophisticated hinting process (Hume et al., 1993) so that the student *discovers* knowledge by him/her self. Our tutors use hinting extensively. Hints appear in different forms (Hume et al., 1993). These three tutor's actions (asking a question, giving an explanation, summarizing) are also the most common forms for hinting.

Before we present the results of this analysis we would like to clarify a few more points about our tutoring experiment. Eight sessions, which are considered for this study, were conducted by two tutors (AAR and JAM). We are convinced that the reasoning method of these tutors, during problem-solving and tutoring is not *radically* different. Hence we will not make any further distinction in our analysis and results.

6.18.2 <u>Analysis</u>. Each instance of the use of the inference triangle (cognitive processes and their compiled forms) was identified in the tutoring transcripts. We have also specified the inference operation (spreading, originating, and causal summation) and tutor's actions (question, explanation, and summary) accompanying these instances. The results of a coding of transcripts (K39 - K46) are presented in Figure 6.37. From now on we will use the names inference triangle and cognitive process(es) interchangeably. This is because any use of a cognitive process (or a compiled form of it) results in a use (part or whole) of the inference triangle. The tutor has used the inference triangle in two different ways. In the first way the tutor has himself used it to *demonstrate* a use of inferencing in the domain. An example of this case is: "there are 3 main determinants of SV: RAP (filling or preload, CC (contractility) and MAP (afterload)". In this example the tutor has used the collection operation (left arm of the inference triangle) to make a point. This usage of the inference triangle is abbreviated in the Figure 6.37 as TU. In the second way the tutor has *pointed* at, part or whole of the inference triangle, for the student to use. This usage of the inference triangle is abbreviated in the Figure 6.37 as PSU. An example for this case is: “What are the determinants of CO?” In this example the tutor has pointed to the collection operation in a question form. The student needs to use this operation to come up with the correct answer. Hence the tutor guides the student in our tutoring situation either by *demonstrating* a use of the inference triangle or by *pointing* to an appropriate part of the inference triangle for the student to use and discover knowledge.

EXPERT TUTOR'S ACTION	COGNITIVE PROCESSES							INFERENCE OPERATION			WHO IS DOING INFERENCING	
	COLLECTION	SELECTION	INFLUENCE CALCULATION	COLLECTION + SELECTION	COLLECTION + INFLU.-CALCU.	SELECTION + INFLU. -CALCUL	COLLECTION + SELECTION + INFLU.-CALCU.	O	S	CS	TU	PSU
QUESTION	20	7	139	198	0	4	77	63	272	112	0	445
EXPLAINATION	15	1	6	28	3	1	44	29	57	10	94	0
SUMMARY	9	6	4	16	0	3	24	23	29	10	64	0
TOTAL	44	14	149	242	3	8	145	115	358	132	158	445

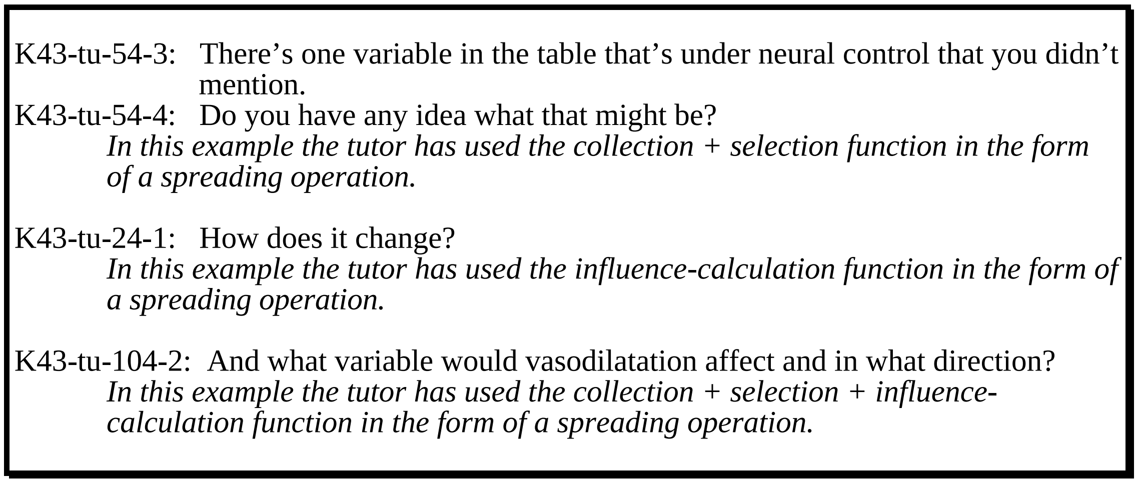
Figure 6.37 Tutoring Sessions - Results

Figure 6.37 shows that the tutor’s action, the question, has been *combined* with most of the cognitive processes and their compiled forms. This ability gives the tutor a powerful way of emphasizing different parts of the inference triangle, which can enable the student to come up with the correct solution of the problem. Some of these combinations are shown in Figure 6.38. It is interesting to note that the tutor asked more questions using the collection + selection operation. This result is contrary to the use of the fully compiled version of the inference triangle (i.e., collection + selection + influence-calculation) by the domain expert during problem-solving (see Section 6.17.2).

We believe that one reason for this is that in the tutoring situation the tutor tries to ensure that the student learns the *basic* reasoning behind the correct prediction for a problem. Figure 6.37 also indicates that more spreading operations were used with the question form. This is not surprising because the task at hand is prediction oriented and the spreading operation is typical of this type.

In the tutoring transcripts, explanation was also combined with all of the cognitive processes (see Figure 6.37). This combination allows the tutor to demonstrate different ways to use domain knowledge. Some examples of this combination are shown in Figure 6.39. Figure 6.37 shows that unlike the question form more explanations were given by the tutor using the collection + selection + influence-calculation operation. Comparing this result with the results in Figure 6.35 shows that the collection + selection + influence-calculation is the *favorite* operation when it comes to demonstrate the use of domain knowledge by the skilled human tutor acting in both roles. Figure 6.37 also shows that more spreading operations were used with the explanation action.

Figure 6.37 shows that the tutor's action, summary, was used with most of the cognitive operations. Figure 6.40 shows some examples of this action.



K43-tu-54-3: There's one variable in the table that's under neural control that you didn't mention.
K43-tu-54-4: Do you have any idea what that might be?
In this example the tutor has used the collection + selection function in the form of a spreading operation.

K43-tu-24-1: How does it change?
In this example the tutor has used the influence-calculation function in the form of a spreading operation.

K43-tu-104-2: And what variable would vasodilatation affect and in what direction?
In this example the tutor has used the collection + selection + influence-calculation function in the form of a spreading operation.

Figure 6.38 Cognitive Functions as Questions

K39-tu-174-3: Co and TPR are the determinants of MAP.
In this example the tutor has used the collection function in the form of an originating operation.

K39-tu-122-3: The reflex accomplishes its work by changing the autonomic outflow to the neurally controlled variables, our old friends TPR, HR and CC.
In this example the tutor has used the collection + selection function in the form of a spreading operation.

K39-tu-124-2: When HR D it causes CO to D.
In this example the tutor has used the collection + selection + influence-calculation function in the form of a spreading operation.

Figure 6.39 Cognitive Functions as Explanations

All in all, in eight sessions our tutors generated more questions than explanations or summaries. Our tutors also used more spreading operations than originating or causal calculations. Figure 6.37 shows that isolated processes (collection, selection, influence-calculation) and partially compiled processes (collection + selection, collection + influence-calculation, selection + influence-calculation) were used more often than the fully compiled inference triangle (collection + selection + influence-calculation). Every pedagogy expert's action (question, explanation, summary) accompanied at least one cognitive process used in the inference triangle.

K40-tu-90-6: MAP went up in DR. So the baroceptor reflex does what you said.
K40-tu-90-7: It lower TPR in order to lower MAP.
In this example the tutor has used the collection + selection + influence-calculation function in the form of spreading operation.

Figure 6.40 Cognitive Function as Summary

So far we have described the different ways in which the inference triangle was used in generating the tutor's response. In the tutoring transcripts we have also identified the instances in which the pedagogy expert used the inference triangle to generate tutoring strategies. Again here our purpose is not to specify a full classification of tutoring strategies but rather to show some uses of the inference triangle by the pedagogy expert in generating various aspects of the tutor's protocol. We will describe three tutoring strategies that the tutor generates by using the inference triangle.

K40-tu-90-7: It low err TPR in order to lower MAP.
K40-tu-90-8: How does the reflex cause TPR to go down?
Here the tutor hinted the student to use the collection and then the selection functions in the forward direction (spreading operation) to determine the mechanism through which he reflex causes TPR to change.
K40-st-91-1: Vasodilation of the vasculature.
K40-tu-92-1: Correct.
The tutor acknowledged the student's correct answer.
K40-tu-92-2: And how does it cause that dilatation to occur?
At K40-st-91-1 the student has responded with a correct intermediate step in the causal chain which links the reflex and TPR, but the tutor here is looking for the step which links dilation and the reflex. As a result he again hinted the student to use the collection and then he selection functions in the forward direction to determine the mechanism through which the reflex causes vasodilatation to occur.
K40-st-93-1: Through the ANS.
K40-tu-94-1: Correct again.
The tutor acknowledged the student's correct answer.

Figure 6.41 A Use of the Inference Triangle to Generate a Directed Line of Reasoning

I) <u>A repeated use of the inference triangle to generate a directed line of reasoning</u>. The directed line of reasoning is one of the common strategies used by our tutors to enable the student to understand the causal mechanism between two physiology concepts. In the directed line of reasoning the tutor guides the student from concept A to concept B by *repeatedly* using the inference triangle. An example use of this strategy is shown in Figure 6.41. In this example the tutor tried to make the student understand the mechanism that causes TPR to change (K40-tu-90-8).

II) <u>The inference triangle as a causal equation</u>. This strategy is used by our tutors to use the inference triangle in the form of a causal equation. An example use of this strategy is shown in Figure 6.42. In this example the student at K39-st-71-1 was unable to use the cognitive process hinted by the tutor at K39-tu-70-3. As a result the tutor adopted this strategy (K39-tu-72-2), which produced better understanding on the part of the student (K39-st-75-1).

K39-tu-70-2:	Come back to MAP.
K39-tu-70-3:	What are its determinants?
K39-st-71-1:	MAP depends on systole and diastole, however I'm not seeing them directly on the data.
K39-tu-72-1:	You are thinking of a way to calculate the approximate value of MAP.
K39-tu-72-2:	I'm thinking of a causal statement that says MAP =.
K39-tu-72-3:	Finish it.
K39-st-73-1:	MAP = pressure that {INTERRUPTED STUDENT INPUT}
K39-ti-74-1:	Write an equation using only variables in the prediction table that says MAP =.
K39-st-75-1:	MAP = TPR X RAP ?
K39-tu-76-1:	Close.

Figure 6.42 The Inference Triangle as a Causal Equation

III) <u>Shift in the direction of inference operation</u>. Our tutors used this strategy for two purposes. In the first case this strategy helped the tutor to *confirm* or *reinforce* the student's reasoning. In the second case if the tutor's reasoning in the forward (spreading operation) or backward (originating operation) direction is not working then the tutor switches the direction with the hope that it might be successful. An example of each of these cases is given in Figure 6.43. In Figure 6.43 (a) the student did not know about the concept of cardiac contractility (K44-st-61-1). The tutor at K44-tu-62-1 hinted that the student should use the collection and then the selection functions in the backward direction (originating operation) so that he could understand the mechanism for CC. At K44-st-63-1 the student replied correctly but this did not convince the tutor that the student clearly understood about CC. As a result the tutor at K44-tu-64-1 again hinted that the student should use the collection and then the selection functions but this time in the forward direction. In Figure 6.43 (b), at K13-tu-37-3, the tutor hinted the student should use the collection and then the selection functions in the backward direction (originating operation) to determine the cause of the change in RAP. The student at K13-st-38-1 did not demonstrate the right use of the inference triangle hinted at by the tutor.

As a result the tutor at K13-tu-39-2 hinted that the student should use the same function but this time in the forward direction. This strategy worked and produced a correct response from the student at K39-st-40-1.

K44-tu-60-1:	Can you define cardiac contractility?
K44-st-61-1:	Not really.
K44-tu-62-1:	Do you know what physiological inputs determine its value?
K44-st-63-1:	Ca
K44-tu-64-1:	How does Ca determine contractility?
K44-st-65-1:	A direct ratio of the amount of calcium to excite the cardiac muscle fibers .
K44-st-65-2:	Along with the Na channels and nerve stimulation sympathetic.
K44-tu-66-1:	You're right, changing sympathetic inputs to the heart DOES change contractility by varying the amount of Ca that is available inside each cardiac cell to bring about e-c coupling.

(a)

K13-tu-37-3:	First, what parameter determines the value of rap?
K13-st-38-1:	Venous return and peripheral resistance influences return
K13-tu-39-1:	Not in the way that you seem to think.
K13-tu-39-2:	If co is made to vary what effect will that have on the central venous compartment or rap?
K13-st-40-1:	Rap will drop due to faster emptying than fulling
K13-tu-41-1:	So, since you predicted that in dr the co i what must you predict will happen to rap?
K13-st-42-1:	Rap d
K13-tu-43-1:	And if rap d what will happen to SV?

(b)

Figure 6.43 Examples Showing a Shift in the Direction of Use of the Inference Operation. (a) To Confirm the Student's Reasoning, (b) To Suggest Another Way of Reasoning

6.19 <u>Shared Knowledge: "Glue" Between the Domain and the Pedagogy Experts</u>

Skilled tutoring requires expertise in the subject matter (domain knowledge) and in the process of tutoring (Galdes, 1990; Khuwaja et al., in preparation (a)). Several papers by early researchers made us realize the importance of domain knowledge in the process of tutoring. For example, Stevens et al. (1982) from their research in WHY system concluded that

> In much of psychology, there has been a bias towards emphasizing highly general, domain-independent mechanisms that are supposedly central to the instructional

> process. Our work demonstrates that such a perspective is incomplete without a detailed consideration of domain-specific knowledge, its representation and <u>its interaction with more general aspects of cognition</u> (p. 13).

Although a great deal of work has been done in formalizing domain and instructional knowledge (Polson & Richardson, 1988), we are still ignorant about one of Stevens et al.'s conclusions: *interaction* of the domain knowledge with other types of skilled tutor's knowledge.

We believe that one of the major reasons for the effectiveness of the skilled human tutor lies in the nature of interaction between his different types of knowledge. This interaction provides the "glue" between different types of expertise and the different roles of the skilled human tutor.

In the previous sections we have seen that the domain and the pedagogy expert share two different types of knowledge: general domain models and the inference triangle. These knowledge types are the characteristic of the domain expert. But these have also facilitated the generation of tutoring responses and strategies by the pedagogy expert. In other words the two roles of our skilled tutors used the same knowledge types for different purposes. We believe that these knowledge types provide the "glue" that integrates different types of expertise in the skilled human tutor and makes the whole process of tutoring effective.

6.20 <u>**Theoretical Orientation of the Model of Tutoring: Metaphors that Explain Our Tutor's Behavior**</u>

The previous sections described a conceptual view of the model of tutoring that I have developed for CIRCSIM-Tutor (v.3). In this section we will discuss the theoretical orientation of this model. This model is based upon the behavior of our tutors in the keyboard-to-keyboard sessions. Our tutors do not explicitly follow any theory while tutoring, instead their behavior is purely based on their extensive experience as physiology teachers and researchers in automated teaching systems (see Chapter III & IV). The purpose of finding the theoretical orientation of this model of tutoring is two-fold: (1) it will help us better understand the behavior of our tutors, and (2) this will provide us with an opportunity to visualize the characteristics of our model in comparison with the models available in the ITS/educational literature. In this section we will also address the question: How the goals of CIRCSIM-Tutor (v.3) are achieved by the use of this model of tutoring?

This model of tutoring is in the tradition of the second-order theory of tutoring (Ohlsson, 1991). Putnam (1987) has called this theory "the diagnostic/remedial perspective of tutoring," whereas Littman et al. (1985) has called it "misconception-based tutoring." This theory assumes that the learner possesses some representation of the subject matter but this representation is either incomplete or incorrect or both. "The job of the teacher is to provide remediation for the discrepancies between the learner's representation and the complete and correct representation" (Ohlsson, 1991, p. 35). In other words, the goal of this theory is that the learner ultimately integrates his/her view of the domain into a correct, coherent, and desired model of the domain. I call this theory the integration theory of tutoring. In accordance with this theoretical orientation this model puts more emphasis on remedying the student's misconceptions.

The behavior of our tutors could be explained using the jigsaw-puzzle metaphor (see Section 2.3.4), which is consistent with the integration theory of tutoring. According to this metaphor the activities of the tutor could be visualized as if he/she is solving a jigsaw-puzzle (see Figure 2.2). This puzzle represents the learner's mental representation of the subject matter (see Figure 2.2 (b)). When the student comes for tutoring the tutor assumes that the student's domain knowledge at that point in time is analogous to a partially completed puzzle, i.e., the student possess domain knowledge but not in a complete and integrated form. The tutor classifies the pieces of this puzzle into five different categories. (1) Chunks of domain knowledge that are correct, (2) chunks that are assumed correct by the tutor, (3) chunks that are missing, (4) chunks that are distorted due to a misconception, and (5) chunks that are dubious.

Immediately after the prediction collection phase, the tutor classifies the knowledge of the student roughly as either assumed correct or missing or distorted. Only in the tutoring phase the tutor gets a chance, to some degree of certainty, to classify the knowledge of the student according to these five categories. As is clear from the previous sections, the types of student's knowledge that get most attention are missing and distorted chunks. It is possible that, during the tutoring phase, the tutor changes his classification about a piece of knowledge. For example, if this piece of knowledge is assumed to be correct, the tutor might discover during the tutoring phase that it is instead a distorted chunk of knowledge. In this case, this model will give more attention to this piece of knowledge. The goal of the tutor here is that it at least fixes missing and distorted pieces of the puzzle so that a clear picture may emerge. This puzzle solving process is partially guided by the teacher's knowledge of the domain (see Figure 2.2 (a)).

Unlike a person solving a jigsaw-puzzle, the teacher in real life does not have physical access to the student's mental state of the subject matter knowledge. He/she can only convey the domain knowledge using various teaching actions (e.g., asking questions, providing summaries) such that it facilitates the student to integrate his/her knowledge into a coherent, correct, and desired mental model of the domain.

In CIRCSIM-Tutor (v.3) the goal of the tutor is that the student learns a mental model of the CV system and a problem-solving procedure that guides him/her to solve CV problems. In other words the student should be able to develop a correct situation-specific model for a CV problem. The tutor in our situation is neither teaching a complete general model of the CV system in an expository way nor conveying a full problem-solving algorithm to the student. Instead, the tutor hopes that the student will acquire these if tutored about only the missing and the distorted chunks of knowledge. This may sound like a big assumption, but the evaluation study of the method of our tutors suggests that the student does learn in our tutoring session.

Once different pieces of the puzzle have been identified, the tutor tries to solve this puzzle, i.e., he starts the remediation phase. The activities of the tutor in this phase turns this two dimensional puzzle into a three dimensional one. These activities could be described by another metaphor called the zoom lens metaphor (see Section 2.5.3). Reigeluth & Stein (1983) have described this metaphor in the context of the elaboration theory of instruction. This theory has many characteristics that are similar to the model of tutoring described here. Viewing the remediation process of this model of tutoring through the zoom lens metaphor is similar in many respects to studying a picture through a zoom lens on a movie camera. A person starts with a wide-angle view that allows him/her to see the complete picture and its parts but without details. If the person wishes he/she can zoom in using the lens to see more details of the parts of the picture. In this metaphor, Reigeluth & Stein (1983) assume that "instead of being continuous, the zoom operates in steps or discrete levels" (p. 340). After studying a part of the picture the person can zoom out again to the wide angle view to see other parts of the picture and analyze the context of the inspected parts with the whole picture. The person can continue the zooming in and zooming out operation at several levels and parts of the picture to analyze the picture at sufficient depth.

Notice that the zoom lens metaphor analogously describes the domain model switching behavior of our tutors mentioned in Section 6.14. In other words, the tutor traverses different levels of this puzzle while solving a jigsaw-puzzle. The major objective of the tutor here is to fix a piece of the puzzle at some deep level such that the puzzle is solved at least at the surface level. Interestingly, the remediation process of our tutors use different models of the domain. The elaboration dimension of the parametric viewpoint (see Figure 6.19) form a simple-to-complex sequence. This sequence "provides meaningful application-level learning" (Reigeluth & Stein, 1983, p. 337). This sequence also ensures that the student is always aware of the context and importance of the different ideas that are being taught. A form of this simple-to-complex sequence has also been used in Ausubel’s (1968) subsumptive sequence, in Bruner’s (1966) spiral curriculum, and in Norman’s (1973) web-learning.

The educational philosophy of the model of tutoring described here is also a version of the view in which learning is described as a process of model tuning. This view of learning is developed by Collins (1985) in the context of WHY system (see Section 2.4.3.1). Unlike our model of tutoring, the tutoring scenario proposed by Collins puts heavy demands on the diagnostic phase of tutoring. With the current state-of-the-art AI research it is not possible to meet these demands. Our model instead uses a view of the student that is much more pragmatic and satisfies the needs of the diagnostic processing required in our tutoring context.

CHAPTER VII
SYSTEM VIEW OF CIRCSIM-TUTOR (V.3)

7.1 Introduction

In this chapter we will discuss the system model of CIRCSIM-Tutor (v.3). This model is the outcome of the system phase (see Chapter IV). Again the only aspects with which I am concerned, in developing this model, are the domain and pedagogy knowledge used in CIRCSIM-Tutor (v.3).

One of the major advantages of the ITS development framework, described in Chapter IV, is that each stage of development is independent of each other in the sense that each uses a different development methodology. For example, in the conceptual phase human tutoring expertise shapes the conceptual model whereas in the system phase the focus is on the system issues and this produces the system model. In other words, the system model is not based on the behavior of our human tutors. Instead it depends upon system issues that are either domain dependent (e.g., educational context) or independent (e.g., considerations of how to represent curriculum and domain knowledge separately in an ITS).

There are two major advantages of separating the conceptual and system phases. First, the conceptual model, if based on the behavior of human tutors, as is the case in CIRCSIM-Tutor, concentrates more on the expertise and more or less ignores the context in which it exists. The activities in the system phase on the other hand provide a vehicle to broaden the focus of the conceptual phase by making explicit the issues that are implicitly assumed in human tutoring behavior. For example, the pre-session behavior of our tutor (see Section 5.5) was ignored by the developers of the early versions of CIRCSIM-Tutor. With the help of this ITS development framework, I have realized the importance of this behavior in the development of CIRCSIM-Tutor (v.3). Second, the system phase promotes the development of a generic ITS that is flexible and can be used for different domains. In this case the system model acts as a generic engine that is fueled by a conceptual model of some domain.

The system model described in this chapter combines many different ideas in the ITS field. This model is an advance on Lesgold's (1988) view of an architecture of an ITS (see Section 2.5.8). It is based on Ohlsson's (1987) design hypothesis (see Section 2.5.10). It also generalizes Woolf's (1984) planning architecture (see Section 2.5.13), for use at several levels of pedagogy decision making during a tutoring session.

7.2 System Point-Of-View: Context Dependent Issues

As has been mentioned before, in the system phase, the instructional system issues shape the system model (see Section 4.1). There are two types of issues. Issues that are dependent upon the context in which the ITS is used, for example, the domain, the educational setting, and the student population using the system. The second type deals with what we call the context independent issues. These issues are related to developing an ITS that is generic and flexible, that can be used in many different domains and educational settings. This section discusses the context dependent system issues, whereas the next section discusses context independent system issues in detail.

It is the system phase that can use prescriptions from the Instructional System Design (ISD) field. There are many ISD models that could be used to systematically consider context dependent system issues for an ITS. See (Reigeluth, 1987) for some examples of ISD models.

Research on CIRCSIM-Tutor has been going on for six years (see Chapter III). Decisions regarding many context dependent system issues have already been crystallized by the earlier versions. It has already been determined that this system will provide a problem-solving environment to the student; CIRCSIM-Tutor will be a part of a physiology course; the default tutoring method of this system will use the discovery method. See Chapters III and IV for a detailed discussion of many of these issues. All these decisions need to be made before the development of an ITS begins.

Most of the context dependent system issues for CIRCSIM-Tutor (v.3) were inherited from its earlier versions. Two major changes that this system introduces compared to its earlier versions are: the tutoring protocol and the CV procedure set. CIRCSIM-Tutor (v.3) uses Protocol 3 (see Section 5.5.9). It also now has a set of large number of CV procedures from which it can select a problem for the student. Section 7.5.2 sheds more light on this issue.

7.3 System Point-Of-View: Context Independent Issues

One of the objectives in developing the system model is to make the design of the ITS as general as possible so that it can be used to develop systems in several domains. Several context independent issues need to be considered in achieving such a system. The system model described here has combined and improved many ideas in the ITS field. In this section we briefly describe these ideas and the improvement this model has made.

As I explained in Chapter II, it is highly desirable for the design of an ITS to combine model-based and curriculum-based themes into a single model. Model-based ITS's emphasize cognitive models of expertise for their domain tasks, whereas curriculum-based ITS's organize their architectures purely around subject-matter. Any model combining these themes at least needs to distinguish between curriculum knowledge and domain knowledge. Lesgold et al. (1989) define the curriculum knowledge as "the specification of the goal structure that guides the teaching of a body of expertise" (p. 342). Common wisdom in the ITS field says that "expertise can be split apart easily ... and that curriculum is a natural hierarchy of goals and subgoals to teach the natural units of expertise" (Lesgold et al., 1989, p. 342). As we have seen in Chapter V and VI, our tutor's behavior is fundamentally based on their model of expertise but they do not simply use this breakdown of expertise to perform effective tutoring in the domain. Instead, selection of goals (e.g., to remedy misconceptions) and domain knowledge depend upon, for example, their teaching method and the student's difficulty in understanding. Also in order for CIRCSIM-Tutor (v.3) to be a part of a physiology course the system needs to make decisions that suit the curriculum of the physiology course. These all require that CIRCSIM-Tutor (v.3) should support both curriculum and model-based themes within a single system model.

The system model I have developed for CIRCSIM-Tutor (v.3) is based upon Lesgold's (1988) framework for knowledge representation in an instructional system (see Section 2.5.8). Like Lesgold's framework my model makes a distinction between the domain and curriculum knowledge. But unlike Lesgold's framework, it explicitly considers pedagogy knowledge. The basic assumption underlying this model is that the tutor in a tutoring situation makes decisions using its pedagogy knowledge. This knowledge uses curriculum knowledge to organize interaction with the student. The curriculum knowledge, in turn, has its foundation in the underlying domain knowledge. As a result, this organization, naturally introduces structure in the tutoring session and provides an explicit way of monitoring the progress of the student in achieving the goals of the system.

This system model views tutoring as problem-solving or planning. The major ingredients of this view are goals, strategies, and tactics. According to Ohlsson's (1987) principle of teaching plans, the plan generation process, used by the tutor, uses strategies to generate plans for the goals of the tutoring system. The terminal ingredients of these plans are tactics that represent the tutor's actions (e.g., ask a question, give a summary). Strategies, in this view of tutoring, determine the methods of the classical problems of pedagogy (i.e., selection, sequencing, and presentation of the subject matter).

There are several planning architectures available in the ITS literature. For the purposes of this system model I have generalized Woolf's (1984) planning architecture to provide a mechanism for pedagogical decision making. This planning mechanism divides the decision making process into different hierarchically organized levels. Each level successively refines the decision making process into a form such that a customized tutoring plan is generated for the student. Unlike Woolf's Discourse Management Network (DMN), this planning mechanism is general enough to be used for all types of decision making, e.g., to select different exercises for the student or to choose a tutor action (e.g., ask a question, give a hint).

Notice that this organization of the system model does not depend upon the domain issues, instead it is motivated by the idea that the system model be as independent as possible of domain specific considerations.

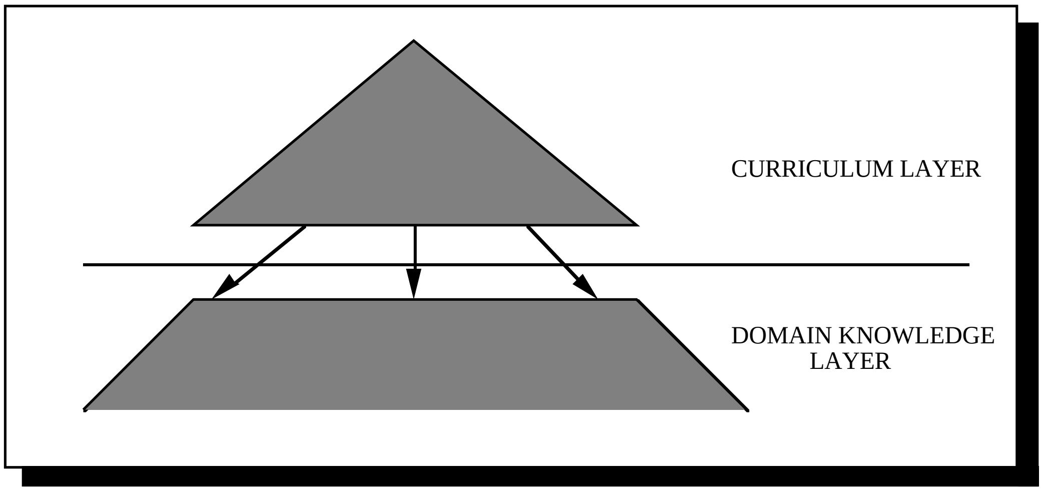


Figure 7.1 Knowledge Layers in Lesgold's (1988) Framework (Meta Cognitive Layer is not Shown Here)

7.4 <u>Dimensions of the System Model</u>

Figure 7.1 indicates the organization of the curriculum and the domain knowledge layers of Lesgold's (1988) framework. In this framework the curriculum layer (also called the goal lattice layer) contains a hierarchy of goals (or topics). The domain knowledge layer contains the domain knowledge that the system is designed to teach. "One way to think about that knowledge is that it is a model of expert capability in the domain. Such knowledge includes both procedures and concepts (i.e., both procedural and declarative knowledge)" (Lesgold, 1988, p. 121). Goals and subgoals in the curriculum layer point to the issues or chunks of knowledge in the knowledge layer.

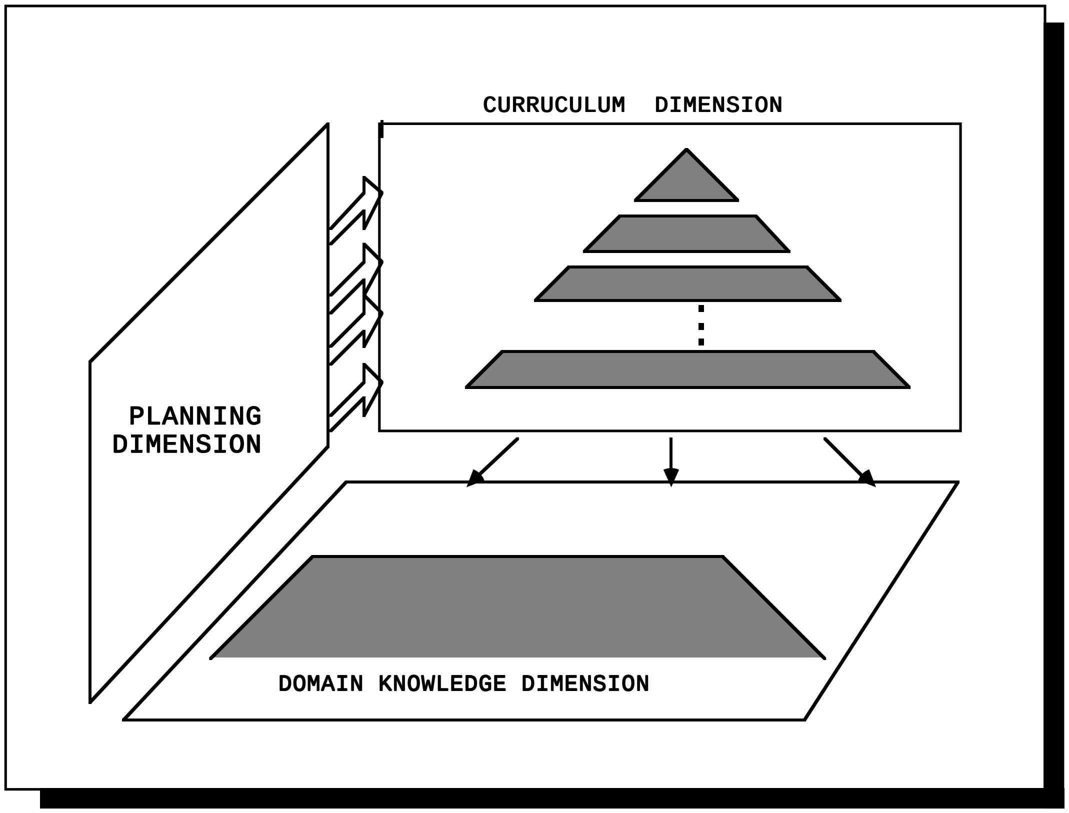


Figure 7.2 Knowledge Dimensions of the System Model

Lesgold's framework does not explicitly consider planning information, rather here a distributed control mechanism is proposed that drives the system. This organization severally limits the generality of the ITS because changing a planning mechanism (or a conceptual model) for the system requires fundamental restructuring of entities in different layers of this framework.

A schematic representation of the system model for CIRCSIM-Tutor (v.3) is shown in Figure 7.2. This model organizes the knowledge into three dimensions: (1) the planning dimension, (2) the curriculum dimension, and (3) the domain knowledge dimension. These dimensions are explained in the following sections. Notice that here planning knowledge in the form of pedagogical decision making is handled separately and explicitly. Arrows in Figure 7.2 indicate that the planning knowledge uses curriculum information that in turn points to or accesses the domain knowledge.

7.4.1 **<u>Planning Dimension: Fueled By the Pedagogical Prescriptions of the Conceptual Model.</u>** It is the planning dimension that makes this system model different from Lesgold's (1988) framework. This dimension takes the view that tutorial decision making is a kind of planning. It is also this dimension that uses the pedagogical prescriptions of the conceptual model. In other words the tutoring theory is represented by the tactics, strategies, and goals used in the planner. Therefore, this dimension controls the activities of the tutoring system.

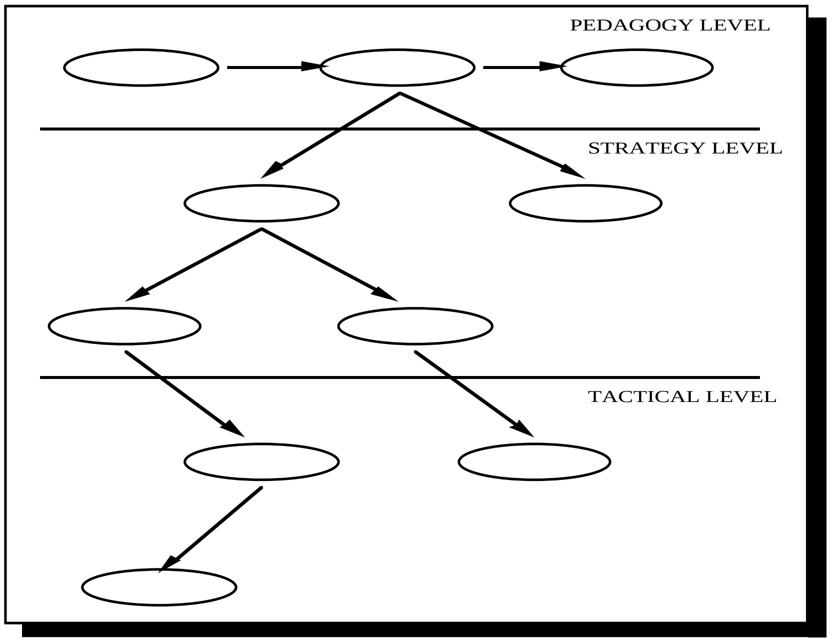


Figure 7.3 A Schematic View of the Generic Planning Mechanism Used in the Planning Dimension

As mentioned, I am using a generalized form of Woolf's (1984) DMN to implement the planning mechanism in this dimension. This planning mechanism views the tutor as making decisions at different levels. These levels basically refine the tutor's decision making process. Figure 7.3 shows a schematic view of this general planning mechanism.

Theoretically there is no limit to the number of planning levels that can be used in the planning dimension but for the purposes of this system model I have limited these to three levels. These levels are: pedagogical, strategical, and tactical. These are also the levels used in the Woolf's (1984) DMN. One of the major assumptions behind this planning mechanism is that the tutor while making pedagogical decisions jumps between different states. Each state corresponds to a high level decision, a strategy, or a tactic. The pedagogy level contains pedagogy states that represent the tutor's high level decision making. Here, the most commonly used states are "select," "deliver," and "complete." While tutoring, the decision making process revolves around these states. Notice that these states correspond to the tutor's major decisions of selecting, sequencing, and remediating topics/errors/error patterns/student difficulties. Repeated transitions between these states form a cycle in which the tutor decides what to talk about next, determines how to talk about it and then evaluates what the student has said in response to the tutor's action. This cycle is quite noticeable in tutoring transcripts (see Chapters V and VI).

In order to implement a high level decision, the tutor considers various strategies. This is the time when the tutor jumps to the strategy level. This level contains strategy states. Depending upon the type of high level decision, the tutor may need to consider one or more strategies to refine its decision making process. Once appropriate strategies have been decided on, the tutor jumps to the tactical level. This level contains tactical states. Each tactic represents an atomic action of the tutor. Each strategy ultimately employs a set of tactics that the tutor needs to consider in order to achieve its current goal. The decision making process viewed through this planning mechanism requires the tutor to jump repeatedly between different tutoring levels and states. This planning mechanism is very general in the sense that it can be used to make pedagogy decisions for different pedagogical styles (see Section 2.5.11). Considering the nature of the behavior of our tutors in the keyboard-to-keyboard session the planning dimension of this system model uses a mixed style of pedagogy, i.e., it combines a global plan-based style with local opportunistic control (see Section 2.5.11). Section 7.5 describes in detail the planning behavior of this system model for a mixed pedagogy style. This planning dimension does not contain a uniform representation of different tutoring levels and states, instead it breaks these levels and states into various spaces to mimic the tutor's different types of decisions. Section 7.5 discusses this issue in detail.

7.4.2 Curriculum Dimension: Provides Goals For Tutoring. This dimension is similar to the Lesgold's (1988) curriculum layer or goal lattice layer. This dimension contains a hierarchy of goals (see Figure 7.2). In the field of Instructional System Design the words goal, topic, and objective are used interchangeably. Here I will not make a distinction between them. Lesgold's framework proposes a flat representation of the hierarchy of goals. On the contrary I propose a systematic breakdown of the goal hierarchy. This systematic breakdown is based on the observation that the tutor makes decisions at different levels. The most common levels are the course level, the exercise level, the unit level and the lesson level. At each level the tutor has a different set of goals. Although these goals are interdependent, at each level their nature, organization, and constraints are different. So instead of using a flat representation of the goal hierarchy, it is more advantageous to use a layered representation of the curriculum. One advantage of this organization is that it promotes the integration of the curriculum and model-based themes of the ITS. Also because of this layered organization, it is possible that the goals in each layer are represented using some customized approach.

It is the planning dimension that uses the curriculum dimension to obtain the goals for the tutoring session. Section 7.5 describes the actual contents of the curriculum dimension of CIRCSIM-Tutor (v.3).

7.4.3 Domain Knowledge Dimension: Containing the Actual Knowledge that Needs to be Communicated to the Student. This dimension contains the domain knowledge that the system is intended to teach. This dimension is similar to Lesgold's domain knowledge layer. In CIRCSIM-Tutor (v.3) this knowledge is in the form of multiple qualitative models of the CV system (see Chapter VI). This knowledge is accessed from the goals and subgoals in the curriculum dimension. It is the planning dimension that uses this knowledge but only through the curriculum goals. See Chapter VI for a detailed description of the nature and organization of domain knowledge in CIRCSIM-Tutor (v.3).

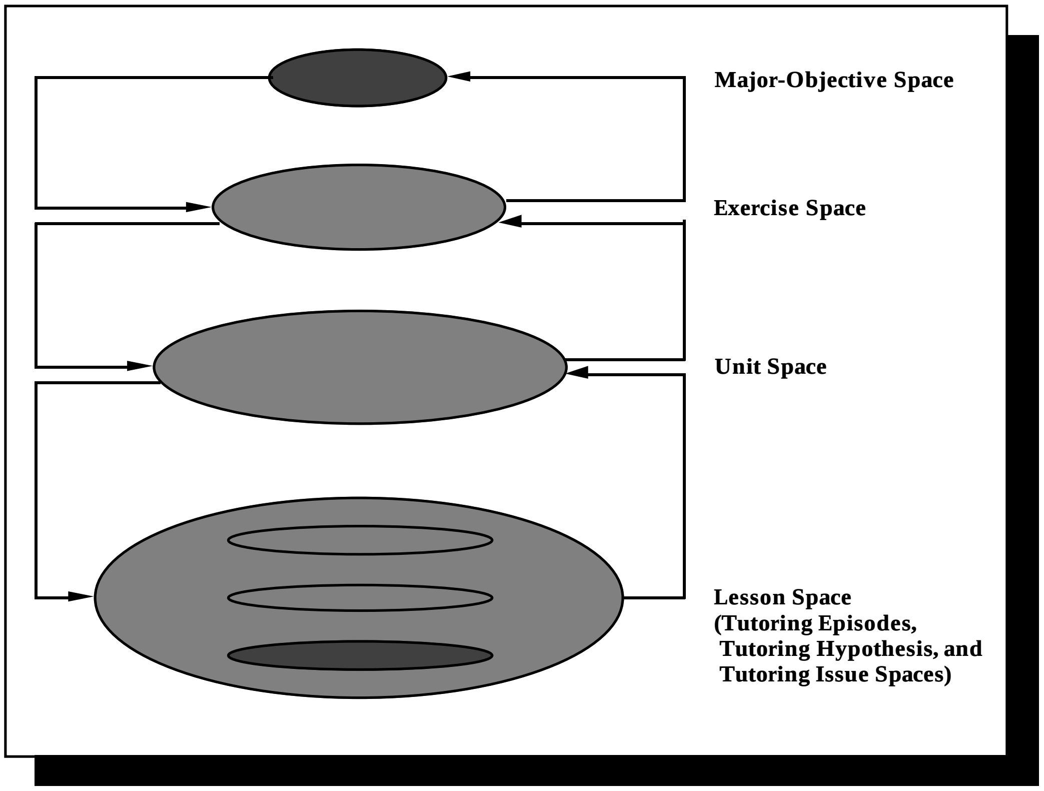


Figure 7.4 System Model as a Set of Tutoring Spaces

7.5 <u>Tutoring Spaces: Another View of the System Model</u>

Human tutors make a number of decisions in order to take action in a tutoring situation. These decisions can be classified into many different categories. A very broad classification is pre-session and in-session decisions (see Chapter V). Instructional system designers commonly distinguish the decision making process of a teacher into following levels: course level, exercise level, unit level, and lesson level (see Romiszowski (1981) for more details). One of the major advantage of this division is that it makes this decision making process modular and easy to manage.

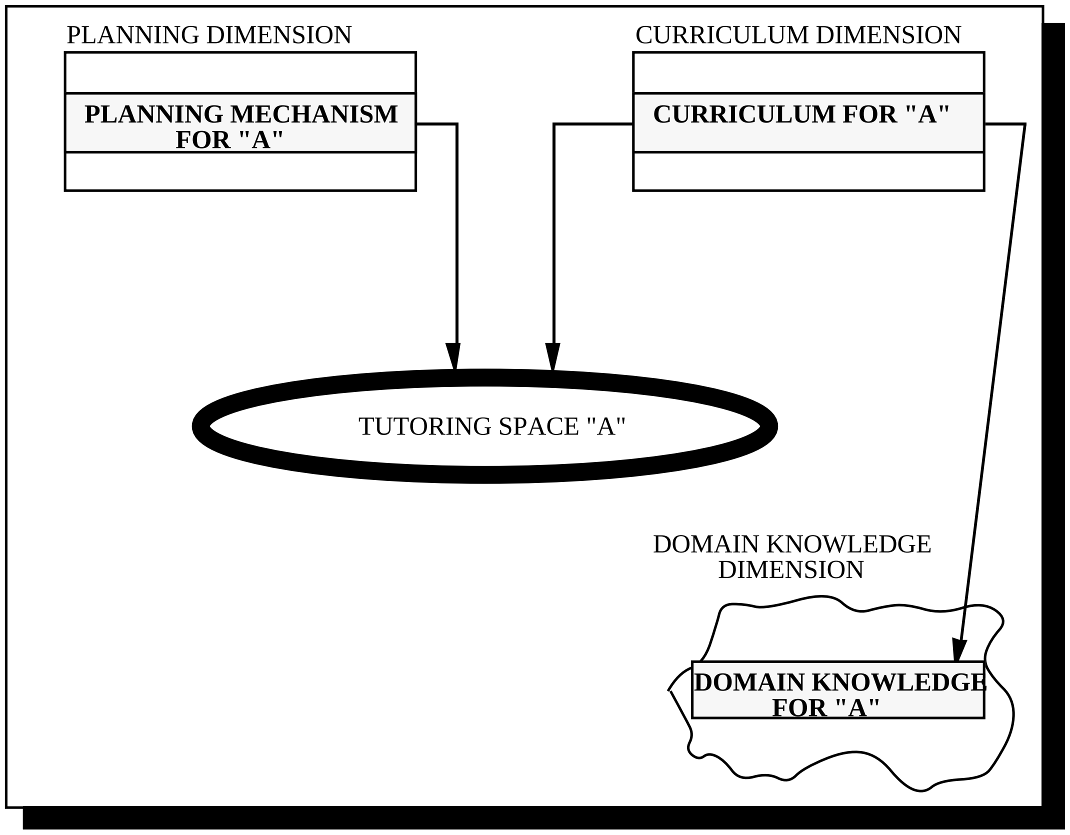


Figure 7.5 A Tutoring Space and Its Connections to Different Types of Knowledge

One view of the system model is described in Section 7.4. In this view the system model consists of three knowledge dimensions. In this section we will describe another view of this model. In this view the system model is consists of layers (I call these tutoring spaces). Each space deals with one major class of decision that the tutor makes. Each space has its own goal representation, constraints suitable to the nature of that space, a planning mechanism, and pointers to the domain knowledge. For example, it is common to tutor problem-solving using a set of exercises or domain problems (Wenger, 1987). The system model using this organization groups all the domain problems and their decision making process into a single layer. This layer will be used by the system model when it tries to select a domain problem for the student. For CIRCSIM-Tutor (v.3), the system model consists of four major spaces. These are the major objective space, the exercise space, the unit space, and the lesson space. These spaces are hierarchically organized and for this reason the overall decision making process is modular. Figure 7.4 shows this view of the system model. Figure 7.5 shows a tutoring space and its connections to different types of knowledge. This figure shows that this model views the tutor as performing decisions in different tutoring spaces. Each tutoring space is driven by a generalized form of Woolf's planning mechanism (see Section 7.4.1). Each space also has its own goal organization and constraints in the curriculum dimension. Finally each space has its associated domain knowledge in the knowledge dimension. The tutor, viewed through this organization of the system model, makes decisions by making transitions between different tutoring spaces. Notice that adding a space in this model is very simple. The designer simply has to add appropriate goal objects in the curriculum dimension, domain knowledge items in the domain knowledge dimension and strategies and tactics in the planning dimension. In other words the designer has to worry about the contents rather than the organization and overall framework of the new tutoring space.

As I have mentioned, we are using a generic form of Woolf's planning mechanism for the system model. With this tutoring space view of the system model, the planning dimension now contains a multilevel mechanism that provides pedagogic decision making for the tutoring system. A schematic view of this planning dimension is shown in Figure 7.6. Here the planning mechanism described in Section 7.4.1 is used for each tutoring space. Once a decision making process for a tutoring space is finished, the system model invokes the planning mechanism in the next tutoring space. The tutor is here viewed as jumping between different tutoring states (see Section 7.4.1). Each tutoring state is connected to another via tutoring links. There are six different types of links. These have been classified into two types: default links and metalinks. This classification is the one used in Woolf's DMN (Woolf, 1984). The default links define the sequences of states normally traversed by the tutor. From a pedagogical viewpoint, these transitions correspond to default tutorial decisions. The metalinks represent metarules that can move the tutor to any state in the planning dimension when their conditions are satisfied.

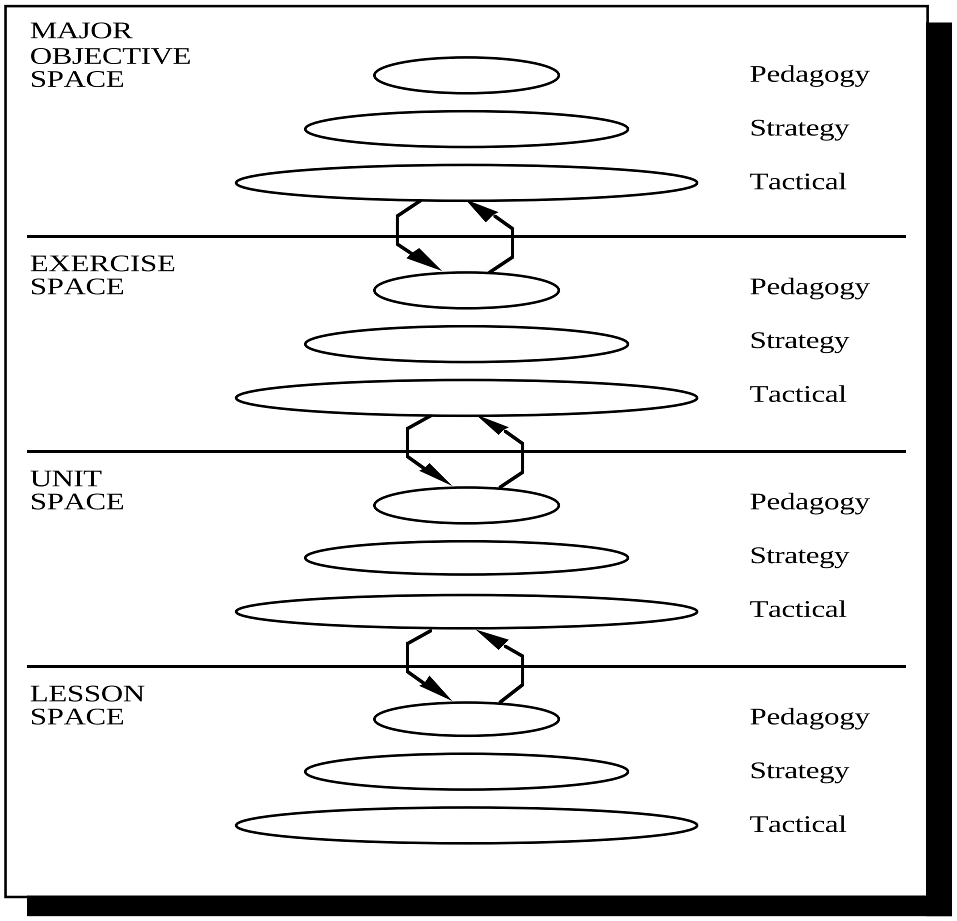


Figure 7.6 A Multi-Level View of the Planning Dimension

Figure 7.7 shows a schematic view of transitions between tutoring states. Here only default links are shown. Besides the above classification, there are three different types of tutoring links. The first one is the progression link. The system model jumps from one tutoring state to another of the same planning level (this could be pedagogical, strategical, or tactical level in any tutoring space) using this type of link. This transition represents the progression of activities of the tutor in the tutoring session. In Figure 7.7 links from states “n” to “z” and “z” to “o” represent this type of transition. The second type of link is called an in-level refinement link. This transition allows a tutor’s action to be refined to its more specific and detailed form. In Figure 7.7 links from states “p” to “z” and “z” to “q” represent this type of transition. The third type of link is called a between-level refinement link. These links are like in-level refinement links except that the tutor can traverse these links to switch between different planning levels. For example, if the tutor wants to refine a strategy into its associated tactics, then the tutor needs to jump from the strategy level to tactical level. These links allow the tutor to accomplish this goal. In Figure 7.7 links between states “l” to “z,” “m” to “z,” “z” to “r,” and “z” to “s” represent this type of transition. All three types of links can be either default or metalinks. In the following sections we will briefly describe the content and purpose of each tutor space used in the system model of CIRCSIM-Tutor (v.3).

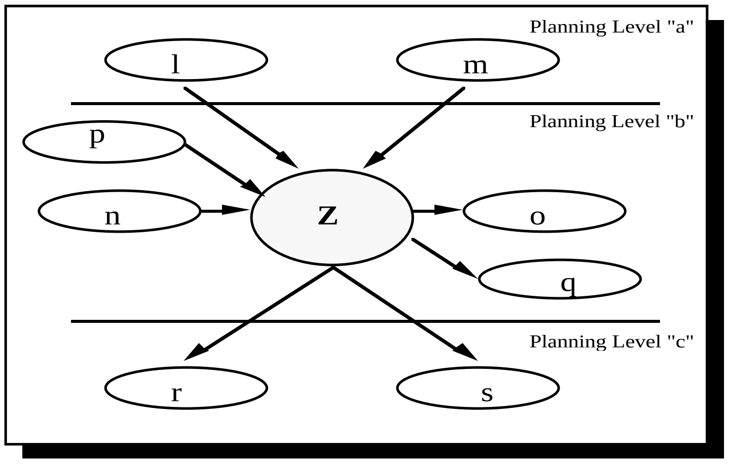


Figure 7.7 Transition Between Tutoring States

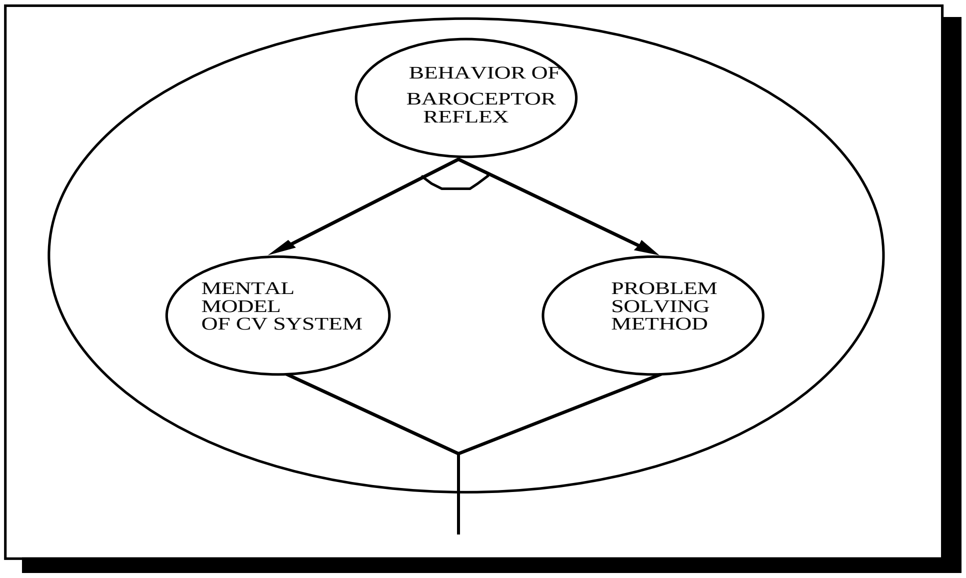


Figure 7.8 Goal Organization for the Major Objective Space

7.5.1 <u>Major Objective Space: Organizing a Tutoring Session Around the Major Goals of the System.</u> The main purpose of this tutoring space is to decide on the major objectives of the system, to select and sequence goals. It also monitors the failure/success of each selected goal. Figure 7.8 shows the goal hierarchy for this space. The major objective of CIRCSIM-Tutor (v.3) is for the student to learn about the functioning of the baroreceptor reflex. This objective is achieved by two subgoals: that the student build a mental model of CV system (i.e., internalize the top level concept map) and that the student acquire a problem-solving algorithm that enables him/her to solve a CV problem (see Figure 7.8). Notice in Figure 7.8 that both of these subgoals need to be satisfied to achieve the major objectives of the system. In CIRCSIM-Tutor (v.3) these two subgoals are not tackled separately. Instead it is assumed that helping the student solve a set of CV problem successfully enables the system to achieve both subgoals of this tutoring space.

Figure 7.9 shows the organization of planning states for the major objective space. This figure divides these states into three planning levels: pedagogical, strategic, and tactical (see Section 7.4.1). Only default links are shown by solid arrows.

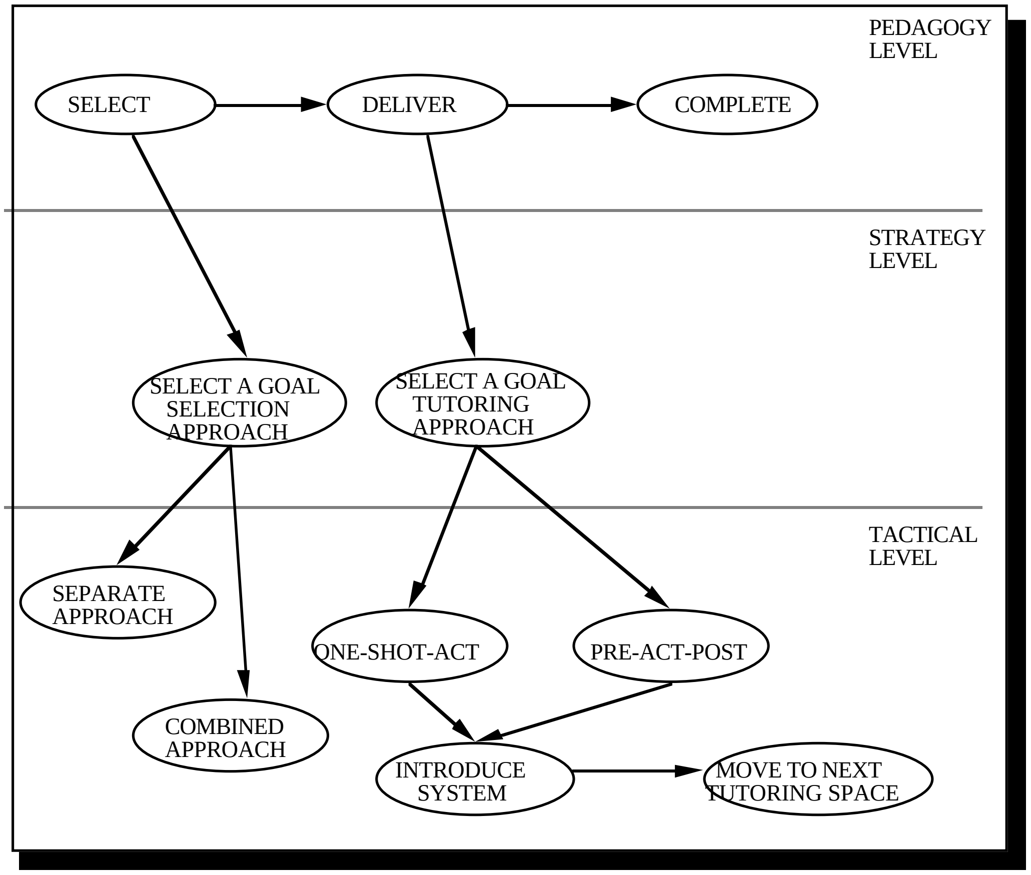


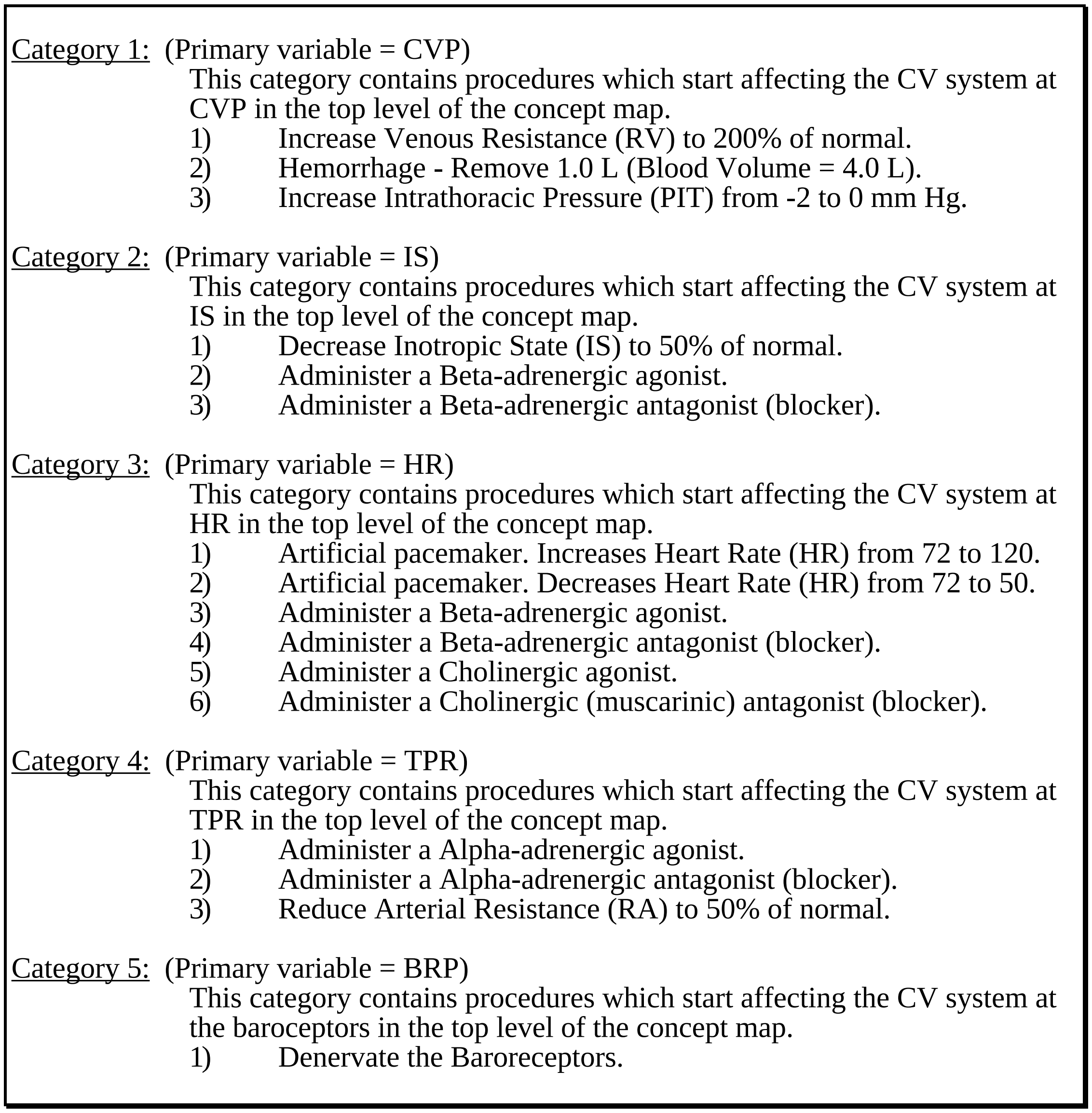
Figure 7.9 Planning States for the Major Objective Space

In the “select” state the system collects goals from the goal hierarchy (see Figure 7.8). Here the system needs to select and sequence the goals. This is achieved by considering the strategy “select a goal selection approach” in the strategy level. This strategy basically points to two tactics in the tactical level. The first tactic, “separate approach,” enables the system to separately handle two subgoals in the goal hierarchy. This tactic is not yet supported by the system. Recently a fellow researcher, Sudnya Sukthankar, developed a computer system (Sukthankar et al., 1993) that enables the student to acquire a general model of the CV system (i.e., top level concept map). This system is called the concept map builder. If this system is integrated with CIRCSIM-Tutor (v.3) then it is possible to invoke this tactic to separately handle two subgoals. If the student model suggests that the student using CIRCSIM-Tutor (v.3) is too confused then the system might use this tactic and suggest that the student first play with the concept map builder program before trying again to solve CV problems. The second tactic, “combined approach,” treats two subgoals as a single unit.

Once the goals are selected and sequenced, the system jumps to the “deliver” state. Here it decides about the tutoring approach for the selected goal. After considering the tutoring strategy “select a goal tutoring approach,” the system selects one of the two available tactics. The “pre-act-post” tactic enables the system to first administer a pre-test to the student. It then allows the student to play with CIRCSIM-Tutor (v.3) by solving CV problems. Once the student has achieved a satisfactory performance, the system will administer a post test. This tactic is not yet supported by the system. The second tactic “one-shot-act” ignores pre and post tests and directly allows the student to solve CV problems. Next the system jumps to the “introduce system” state where it introduces the selected system. At present it can only introduce CIRCSIM-Tutor (v.3). In the future if other systems such as the concept map builder are integrated with CIRCSIM-Tutor (v.3) then depending upon the selected tactic this state will introduce the student to the selected system. Next the tutor jumps to the “move to next tutoring space” state. This state enables the system to switch from the major objective space to the exercise space because the major objective space is designed to reason only about the major objectives of the system. In order to achieve (or tutor about) these objectives (or goals), the system needs to next decide about the CV problems. Once the tutor leaves the major objective space, the activities in this space are temporarily halted until the tutor again jumps back to it. At that time it decides whether the current goal in this space is satisfied by the activities of the tutor in the previous spaces. Depending upon this result, the tutor makes the next move. This decision about the success or failure of the current goal is considered in the "complete" state. Here a metarule brings the tutor back to the "select" state, where the tutor decides whether to select new goal, retry the current goal or halt the activities of the system.

7.5.2 <u>Exercise Space: Developing a Personalized CV Problem Set For the Student.</u> This is the second tutoring space after the major objective space (see Figure 7.4). This space is activated once the approach to achieve the major objectives of the system has been decided about. The purpose of this space is to select a CV problem for the student. Figure 3.2 shows the procedures used in earlier versions of CIRCSIM-Tutor. These versions also do not explicitly reason (or plan) in selecting a procedure for the student. CIRCSIM-Tutor (v.3) has made a big leap in this respect. Here the total number of procedures has greatly been extended. The domain model in CIRCSIM-Tutor (v.3) is powerful enough to solve all these CV procedures (see Chapter VIII). This space is wholly dedicated to choosing a customized set of CV procedures for the student. Two major determinants in developing these instructional plans are the goals of the system and the student capability in solving CV problems. In this section we will describe various classifications of these CV procedures and a set of rules that enable the system to develop a personalized problem set for the student.

7.5.2.1 <u>Different Approaches to Classifying CV Problems.</u> Each CV problem describes a perturbation that perturbs some component of CV system, e.g., hemorrhage affects blood volume (BV) - a CV parameter. For CIRCSIM-Tutor (v.3) our tutors (AAR and JAM) have selected thirteen different kinds of perturbations. Appendix B contains a list of all these perturbations. These perturbations have been divided into three classes: basic procedures, perturbations involving drug effects, and artificial pacemaker procedures. The perturbations in the basic procedure set have been inherited from earlier versions of CIRCSIM-Tutor. The drug set contains six different types of perturbations. The artificial pacemaker set contains two perturbations. Most of these perturbations initially affect a single CV parameter in the concept map. Some of the drugs, on the other hand, affect two parameters in the concept map simultaneously. For example, beta-adrenergic antagonists decrease HR and IS. Due to the nature of the artificial pacemaker it is possible to increase or decrease the heart rate.



Category 1: (Primary variable = CVP)
This category contains procedures which start affecting the CV system at CVP in the top level of the concept map.
1) Increase Venous Resistance (RV) to 200% of normal.
2) Hemorrhage - Remove 1.0 L (Blood Volume = 4.0 L).
3) Increase Intrathoracic Pressure (PIT) from -2 to 0 mm Hg.

Category 2: (Primary variable = IS)
This category contains procedures which start affecting the CV system at IS in the top level of the concept map.
1) Decrease Inotropic State (IS) to 50% of normal.
2) Administer a Beta-adrenergic agonist.
3) Administer a Beta-adrenergic antagonist (blocker).

Category 3: (Primary variable = HR)
This category contains procedures which start affecting the CV system at HR in the top level of the concept map.
1) Artificial pacemaker. Increases Heart Rate (HR) from 72 to 120.
2) Artificial pacemaker. Decreases Heart Rate (HR) from 72 to 50.
3) Administer a Beta-adrenergic agonist.
4) Administer a Beta-adrenergic antagonist (blocker).
5) Administer a Cholinergic agonist.
6) Administer a Cholinergic (muscarinic) antagonist (blocker).

Category 4: (Primary variable = TPR)
This category contains procedures which start affecting the CV system at TPR in the top level of the concept map.
1) Administer a Alpha-adrenergic agonist.
2) Administer a Alpha-adrenergic antagonist (blocker).
3) Reduce Arterial Resistance (RA) to 50% of normal.

Category 5: (Primary variable = BRP)
This category contains procedures which start affecting the CV system at the baroceptors in the top level of the concept map.
1) Denervate the Baroreceptors.

Figure 7.10 CV Procedure Categories Based on the Didactic Goals

In CIRCSIM-Tutor (v.3) we have introduced procedure combinations for the first time. In a procedure combination a perturbation is selected to perturb the CV system and then the student is asked to predict the responses for the predictions table variables. Once this procedure is completed, the system introduces a second perturbation on top of the first one in the CV system. Again the student is asked to predict the responses for prediction table variables. Obviously, these combination problems are relatively challenging for the student to solve. Appendix B contain a list of 45 different procedure combinations selected for CIRCSIM-Tutor (v.3).

Category 1:

This category contains procedures with a default difficulty level = "simple."

1) Increase Venous Resistance (RV) to 200% of normal.
2) Hemorrhage - Remove 1.0 L (Blood Volume = 4.0 L).
3) Install artificial pacemaker. Increase Heart Rate (HR) from 72 to 120.
4) Install artificial pacemaker. Decrease Heart Rate (HR) from 72 to 50.
5) Reduce Arterial Resistance (RA) to 50% of normal.

Category 2:

This category contains procedures with a default difficulty level = "moderate."

1) Decrease Inotropic State (IS) to 50% of normal.
2) Administer a Cholinergic agonist.
3) Administer a Cholinergic (muscarinic) antagonist (blocker).
4) Administer a Alpha-adrenergic agonist.
5) Administer a Alpha-adrenergic antagonist (blocker).

Category 3:

This category contains procedures with a default difficulty level = "difficult."

1) Administer a Beta-adrenergic agonist.
2) Administer a Beta-adrenergic antagonist (blocker).
3) Denervate the Baroreceptors.

Category 4:

This category contains procedures with a default difficulty level = "challenging."

1) Increase Intrathoracic Pressure (PIT) from -2 to 0 mm Hg.
2) Any two procedure combination in sequence (e.g. PRA after DBB).

Figure 7.11 CV Procedure Categories Based on the Default Procedure Difficulty Level

In order to make the system capable of choosing a personalized set of procedures for the student, these CV procedures and their combinations have been classified further. These classifications are described as follows.

One classification of these CV procedures has been based on the didactic goals of CIRCSIM-Tutor (v.3). This classification divides thirteen perturbations into five categories (see Figure 7.10). All perturbations affecting one of the five critical CV parameters have been grouped into a single category. For example in Figure 7.10, three different perturbations affect CVP first among the parameters in the prediction table. (The primary variable is our term for the first variable in the prediction table to be affected.) The fundamental assumption behind this classification is that if the student solves at least one problem from each of these five categories then he/she will be able to understand the behavior of CV system at an acceptable level.

Category 1:
A procedure description in this category has a default difficulty level = 1.
Reason: Primary variable is explicitly given.

Category 2:
A procedure description with a default difficulty level = 2.
Reason: Primary variable is implicitly given in the problem description.

Category 3:
A procedure description with a default difficulty level = 3.
Reason: Procedure variable is explicitly given.

Category 4:
A procedure description with a default difficulty level = 4.
Reason: Procedural variable is implicitly given in the problem description.

Figure 7.12 CV Procedure Categories Based on Default Procedure Description Difficulty Level

Another classification of the CV problem is based on their level of difficulty. This classification divides CV problems into four categories (see Figure 7.11). The first category contain procedures that are graded by our tutors as “simple.” The second category contain procedures that are moderately difficult. The third category contain relatively difficult procedures and the final category has challenging CV problems.

The final classification is not based on the CV procedures themselves but on their problem descriptions. Each problem description of a CV procedure describes the first action of the perturbation on CV system. A procedure description can explicitly or implicitly describe the affect of this action on either primary or procedure variable. There are four categories of problem description (see Figure 7.12). From their experience, our tutors think that problem descriptions that explicitly state the primary or procedure variables are easier for the student to understand than the description that implicitly convey this information. Also problem descriptions that contain information about the primary variable are easier than descriptions that only contain information about the procedural variable. A complete list of possible descriptions for thirteen perturbations is given in Appendix B.

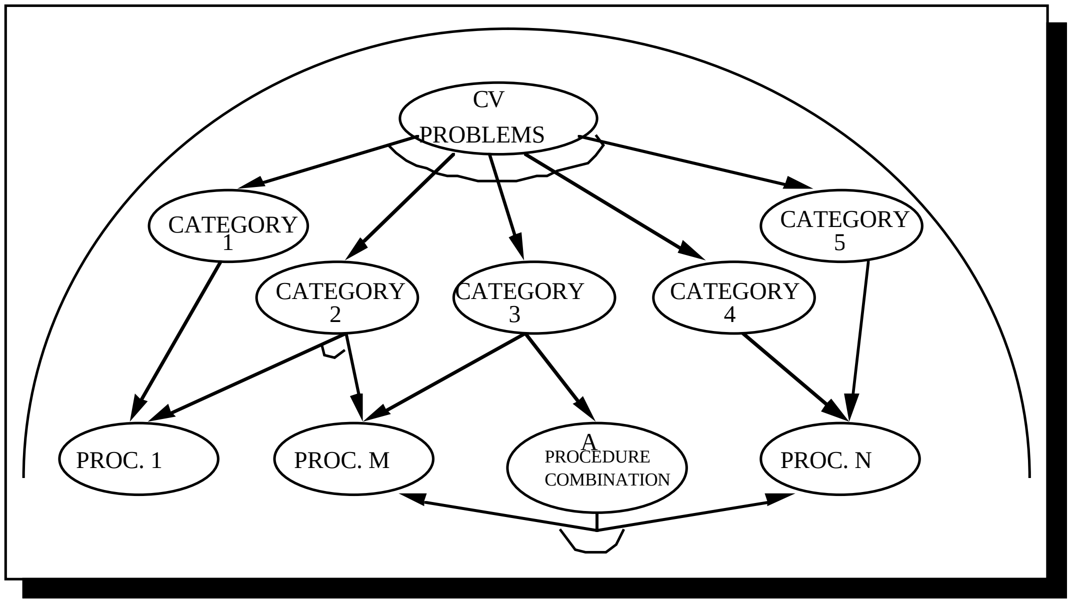


Figure 7.13 A Partial Goal Organization For the Exercise Space

Figure 7.13 shows a partial goal hierarchy for the exercise space of the system model for CIRCSIM-Tutor (v.3). This hierarchy shows that this space will achieve its goals if the student successfully solves a procedure from each of the five basic categories. These categories are based on the didactic goals of the system (see Figure 7.10). Each category node in Figure 7.13 has a pointer to its associated procedure node or procedure combination node.

7.5.2.2 <u>Developing a Personalized Problem Set for the Student</u>. Figure 7.14 shows the planning states for the exercise space. Like the major objective space, this space also divides these states into three levels. The functions of the "select," "deliver," and "complete" states at the pedagogy level were explained in Section 7.5.1. In order to select a goal (in this space this is a CV procedure) the system model uses various strategies and tactics. The first strategy considered here is "who should choose the next procedure." There are two possibilities: either the tutor or the student makes this decision. If it is the tutor that is making this decision then the next strategy considered is "select procedure category." Here the system selects a category of procedures for the student. This categorization is based on the didactic goals of the system. The next step is to select the procedure default difficulty level for the student. Once this is done, next the tutor decides about the procedure description level. At this stage if the tutor ends up with a list of procedures then it selects a procedure randomly from this list. The next step for the tutor is to decide about the way to present the selected procedure to the student. Here it first selects "describe procedure" and then "setup tutoring environment" tactics. Once it has reached this stage, the system has finished goal selection. Next it jumps to the deliver state which ultimately forces the tutor to use next tutoring space, the unit space, to accomplish its goals.

If the system model decides that it is the student who should select the next CV procedure then still the same set of strategies are considered by the system as in the case when the tutor is making the decision. The only difference comes when the system model decides to present the selected set of procedures to the student. Here instead a menu is provided by the system to the student to select a CV procedure. These activities are accomplished by the “give menu,” “ask for choice,” and “setup tutoring environment” tactics. The system considers its goals and the knowledge state of the student while developing a menu for the student. All the CV procedures in this menu are kept well within the reach of the student’s capability. Next we briefly describe the logic used in deciding about various strategies in the exercise space.

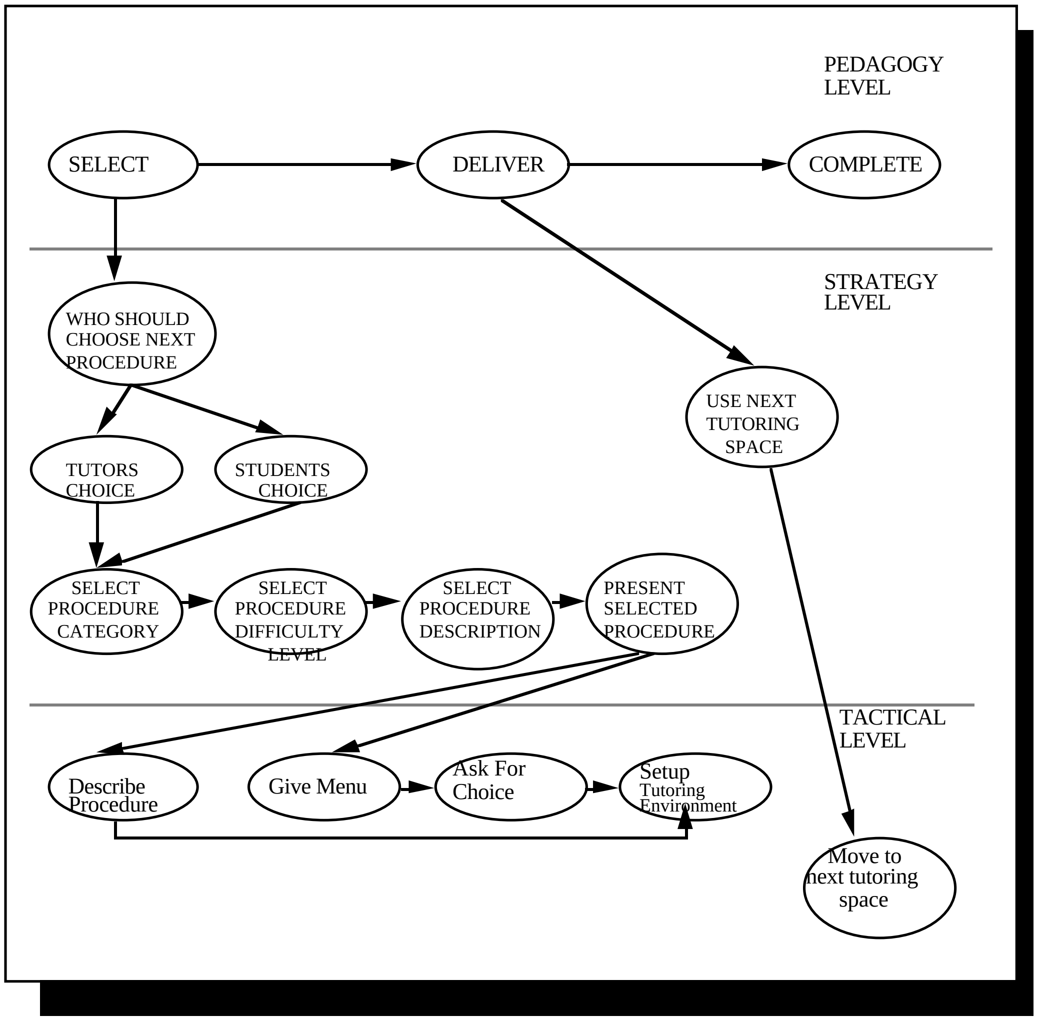

PEDAGOGY LEVEL
SELECT
DELIVER
COMPLETE
STRATEGY LEVEL
WHO SHOULD CHOOSE NEXT PROCEDURE
USE NEXT TUTORING SPACE
TUTORS CHOICE
STUDENTS CHOICE
SELECT PROCEDURE CATEGORY
SELECT PROCEDURE DIFFICULTY LEVEL
SELECT PROCEDURE DESCRIPTION
PRESENT SELECTED PROCEDURE
TACTICAL LEVEL
Describe Procedure
Give Menu
Ask For Choice
Setup Tutoring Environment
Move to next tutoring space

Figure 7.14 Planning States for the Exercise Space

This space uses two concepts from the student model to decide about various strategies. (1) Single problem global assessment (SPGA): This is the assessment of the student's knowledge state for a single CV problem. Possible assessment values come from the following discrete set {-1, -1, 0, 1, 2}. Here -2 and +2 represent poor and very good respectively. (2) Successive single problem global assessment (SSPGA). This is the assessment of the student knowledge over successive problems. This variable can take a value from the set {-1, 0, +1}. Here +1 indicates that the student's performance has improved, 0 means it has not changed, -1 means it has gotten worse. The following rule is used to decide about the strategy "who should choose next procedure."

Current SSPGA = 0	Current SSPGA = +1	Current SSPGA = -1	Current SSPGA = nil
SPGA	SPGA	SPGA	SPGA
0 + Description Level	0 + Description Level	0 + Description Level	
+1 + Difficulty Level	+1 + Difficulty Level	+1 + Difficulty Level	+1 + Description Level
+2 + Difficulty Level, + Description Level	+2 + Difficulty Level, + Description Level	+2 + Difficulty Level, + Description Level	+2 Not Possible

Figure 7.15 Rules to Decide About Procedure Difficulty and Description Levels

```
If SPGA of the last problem is negative (-1 or -2) or
        it is the first procedure
Then the tutor will make decisions about the procedure category, difficulty level,
        and description
Else the student will make these decisions
```

Now assume that it is the tutor who is making these decisions. In this case the following rules apply to select a procedure for the student.

```
If it is the first procedure
Then select a procedure from category 1.  This procedure must have a
        difficulty level of 1 and a description value of 1

If it is the second procedure
Then select a procedure from categories 2, 3, or 4

If the student's performance (SPGA) is good, and
        this is the third procedure
Then introduce category 5
```

For the second and following procedures the system uses the strategy shown in Figure 7.15 to make decisions about the procedure difficulty and description levels. Figure 7.15 can be read as follows. If SSPGA = 0 and SPGA = 0 then increase the current procedure description level by one for the next procedure. Keep the difficulty level for the next procedure the same as the current procedure. If SPGA is negative in any of the cases in Figure 7.15 then for the next procedure keep the procedure difficulty and description levels the same as for the current procedure.

```
If this is the fourth procedure and
        the tutor has decided about the last three consecutive procedures
Then stop tutoring and suggest the student to do prerequisite reading before
        continuing to solve CV problems

If there is no procedure available having the required
        difficulty or description level
Then give a single increment to either procedure difficulty or description
        level as required
```

7.5.3 Unit Space: Taking Care of the Tutoring Protocol. This is the third space of the system model (see Figure 7.4). The main objectives of this space are to divide a CV problem into its major units and then plan for the tutoring of these units. As we have seen in Section 5.5.1 our tutors divide their problem solving methods into three major phases: DR, RR, and SS. While tutoring our tutors consider each of these phases one at a time. The sequence in which these phases are considered is always the same, i.e., DR, then RR, and finally SS.

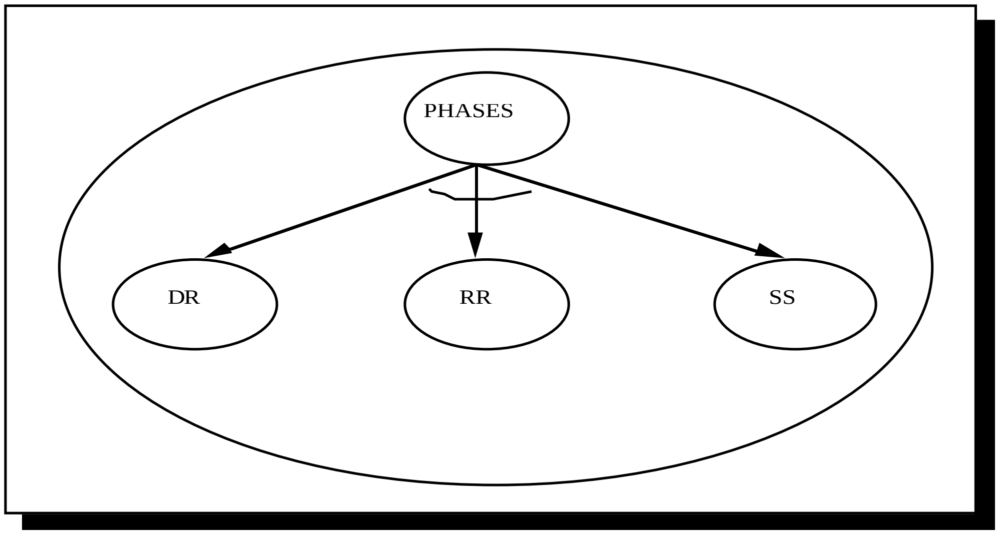


Figure 7.16 Goal Organization for the Unit Space

In the unit space the goals are organized as shown in Figure 7.16. Just like our tutors this space divides a CV problem into three phases. All of these phases need to be considered to satisfy the objectives of this space.

Our tutors use a tutoring protocol to exercise control over a tutoring session. One of the major responsibilities of the unit space is to plan the tutoring of the three phases according to some tutoring protocol. Although the planning mechanism used in this space is general enough to be used to plan using any tutoring protocol, for the purposes of CIRCSIM-Tutor (v.3) we have selected Tutoring Protocol 3 (see Chapter V). This protocol is carried out by the tutoring states of Figure 7.17. In this space, the function of the pedagogy level is the same as explained in the previous spaces (see Sections 7.5.1 and 7.5.2). The selection of a goal in this space is a straightforward function because here there are only three subgoals and each of these needs to be used in a fixed sequence.

The deliver state uses various strategies and tactics to accomplish its function. As mentioned, this space uses tutoring Protocol 3. This protocol here is achieved by considering following four strategies. First, the system introduces the selected phase to the student. Next it introduces various rules that the student can use to predict the qualitative values for the prediction table variables. Next, the system switches back and forth between the prediction collection phase (PCP) state and the tutoring phase (TP) state (see Figure 7.17).

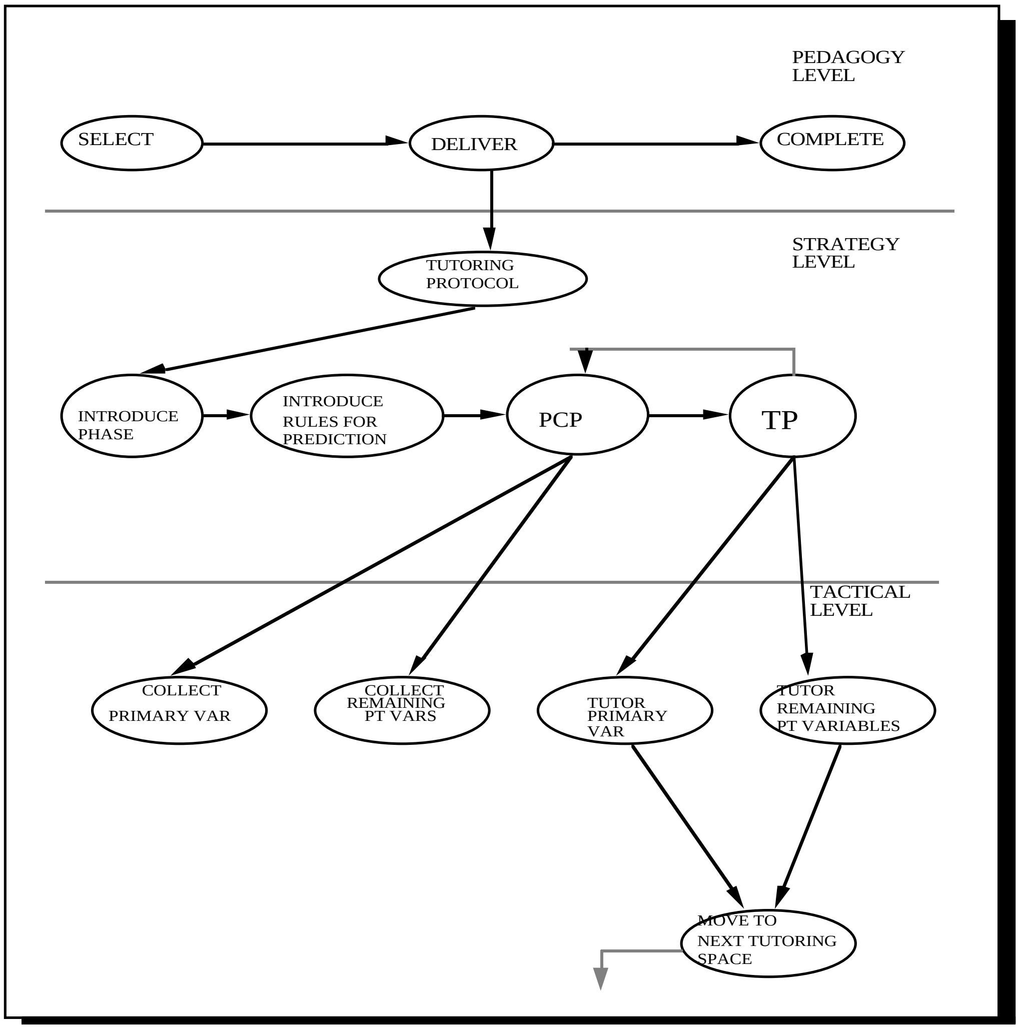


Figure 7.17 Planning States for the Unit Space

Each of these two strategies have associated tactics. The system first considers the “collect primary var” tactic and then jumps to the tutoring phase. If the student has correctly predicted the value for the primary variable then it jumps back to the prediction collection phase to “collect remaining prediction table variables.” On the other hand if the student has made an error in predicting primary variable then the system jumps to the “tutor primary variable” state. This tutoring is achieved by considering the next tutoring space. Once the tutoring of the primary variable is successfully accomplished the system again jumps back to the prediction collection phase state to “collect remaining prediction table variables.” Next, while in the “tutor remaining predictions table variable” state, it starts tutoring for the current CV phase. Each time the system successfully accomplishes its goal for a CV phase it invokes the next CV phase for the student.

7.5.4 Lesson Space: Initiating a Dialogue with the Student. One of the major purposes of the multilevel planning mechanism (see Figure 7.6) used in the system model is to systematically and hierarchically divide the decision making process so that it corresponds to the major decisions that our tutors perform while organizing and acting in a tutoring session. The decision making process used in the first two spaces of this system model is not directly observable in the in-session behavior of our tutors in the keyboard-to-keyboard sessions. It is the system view that makes explicit this very high level decision making of the tutor.

The decision making processes used in the spaces that are below the exercise space are observable in the in-session behavior of our tutors. Once a decision about a phase of CV procedure has been made, the system starts an interaction with the student. The decision making process of the pedagogy expert during this interaction is captured in the lesson space (see Figure 7.4). It is the conceptual model described in Chapter VI that is responsible for the behavior of CIRCSIM-Tutor (v.3) in the lesson space. Two major objectives of this space are: (1) tutor so that the major misconceptions of the student are remediated, (2) teach the major concepts of the domain. The first objective has the highest priority. The second objective is pursued only if time and opportunity is available to the tutor to convey such information. In order to mimic the behavior of our tutors, as captured in the conceptual model (see Chapter VI), the lesson space of the system model has been divided into the following subspaces: (1) tutoring episode, (2) tutoring hypothesis, and (3) tutoring issue. The tutoring episode space contains the required ingredients to make decisions about the error patterns (see Section 6.4) that are inferred from the student behavior. The tutoring hypothesis space on the other hand deals with the decision making that is required for the student difficulties (see Section 6.4). Once a student difficulty has been selected the system model invokes the next space, the tutoring issue space, that takes care of the actual interaction of the tutor with the student. This tutoring space is the lowest space of the system model. Here it is assumed that the system has done sufficient higher level planning to start a dialogue with the student. The next few sections describe each of these subspaces of the lesson space.

7.5.5 <u>Tutoring Episode Space: Handling Error Patterns and Domain Topics</u>. As we have seen in Section 5.5.11, the in-session behavior of our tutors is organized around episodes of tutoring in which, most of the time, the tutor tries to remediate the misconceptions of the student that cause errors in predictions. It is the responsibility of the tutoring episode space to plan for such an episode of tutoring. As we have seen in the previous chapters, two issues dictate the creation of such an episode. (1) Errors in the student's prediction. This causes sensitization of error patterns. It is the error pattern level around which the decision process of the tutor revolves during a tutoring session (see Section 6.6). Hence in this space error patterns usually invoke a tutoring episode. (2) CIRCSIM-Tutor (v.3) has an explicit curriculum. The system tries as much as possible to make sure that the student using the system covers most of this curriculum. It is not the major goal of the system that it explicitly cover each part of this curriculum, rather the major goal is that the student learn (via the discovery method) while performing problem-solving activities. The system silently tries to infer the coverage of this curriculum by the student from his/her behavior.

It is not always possible to infer the knowledge of the student for all parts of curriculum from his/her problem-solving activity. As a result, as the opportunity arises the system tries to invoke a generic topic of the domain for discussion.

Now we will describe the activities of the system model in the tutoring episode space. But first we will consider the goal organization used in the lesson space.

7.5.5.1 Topic Network. Just like every other space of the system model, the lesson space also has a goal organization that constitutes a portion of the curriculum dimension of the system model. This goal organization is common to all three sub-spaces of the lesson space (see Figure 7.18). I call this goal organization the topic network. As the name suggests, the topic network consists of a set of domain topics. I define a topic as a short description of a piece(s) of domain knowledge (e.g., regulated variable). It is an entity that the tutor can select for tutoring. These topics are connected to each other via generic didactic links. These links are different from the domain relations (e.g., causal relations). The purpose of these relationships is to help the system develop a personalized goal set for the student. Figure 7.19 illustrates various didactic links between domain topics in the topic network.

The core topics are the essential topics that the student must know about in order to understand fully the behavior of the baroreceptor reflex (e.g., the role of the baroreceptor reflex). The supporting topics help in the understanding of the core topics. These topics could be in the form of prerequisite knowledge (e.g., What is a physiology parameter?) or basic skills (e.g., How to propagate causal influence from one physiology variable to another?). The peripheral topics are not essential but help the student to generalize and extend their domain knowledge (e.g., Intracellular agonist-receptor transduction events).

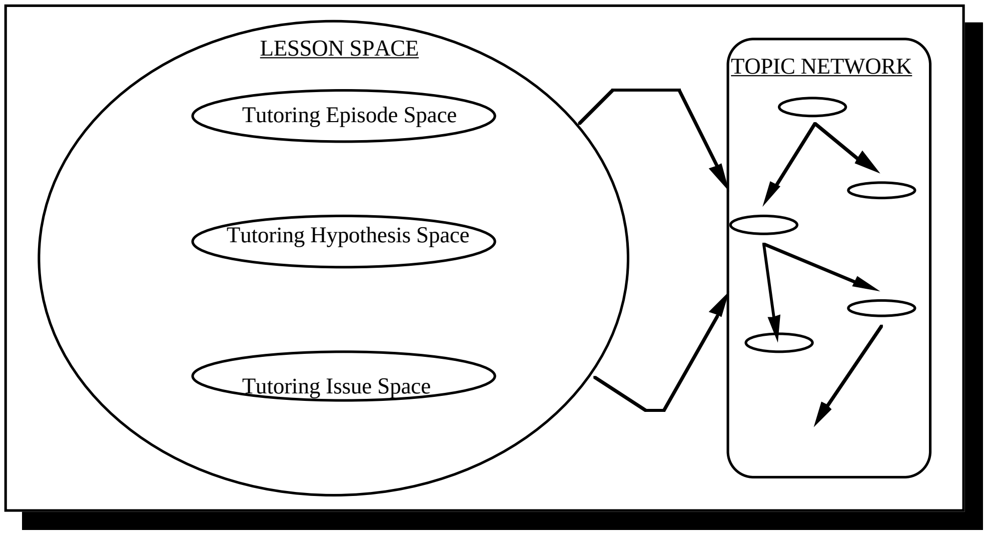


Figure 7.18 Lesson Space Accessing the Topic Network

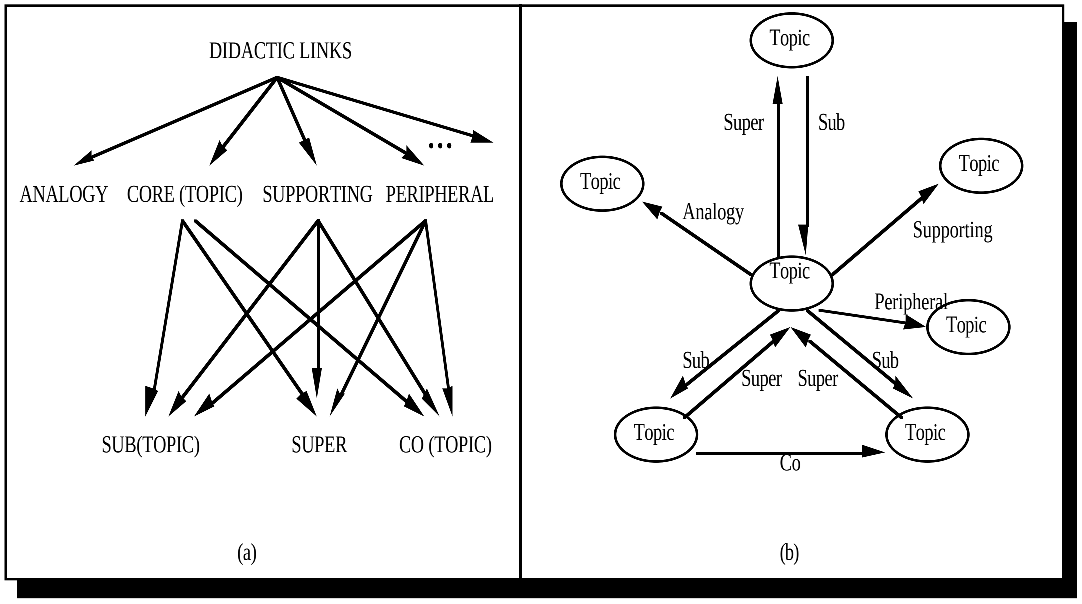


Figure 7.19 (a) A Partial Classification of Didactic Links,
(b) A Schematic View of Domain Topics Connected Via Didactic Links

All three types of topics are related to their respective types using subtopic, super topic, and cotopic relations. A topic, say C, is a cotopic to another topic, say T, when C is didactically related to T but C has no subtopic-super topic relationship with T (e.g., heart rate has a co-topic relationship with total peripheral resistance and inotropic state because all three are neurally controlled variables). The analogy relationship helps the tutor to develop an analogy for a core topic (e.g., the multiplicative relationship MAP = CO x HR has an analogy link to the Ohm's Law Equation V = I x R).

The development of the topic network is one of the complex tasks in developing the system model. This network is not yet complete. Further research is needed to fully understand the nature and utility of these didactic links in our domain.

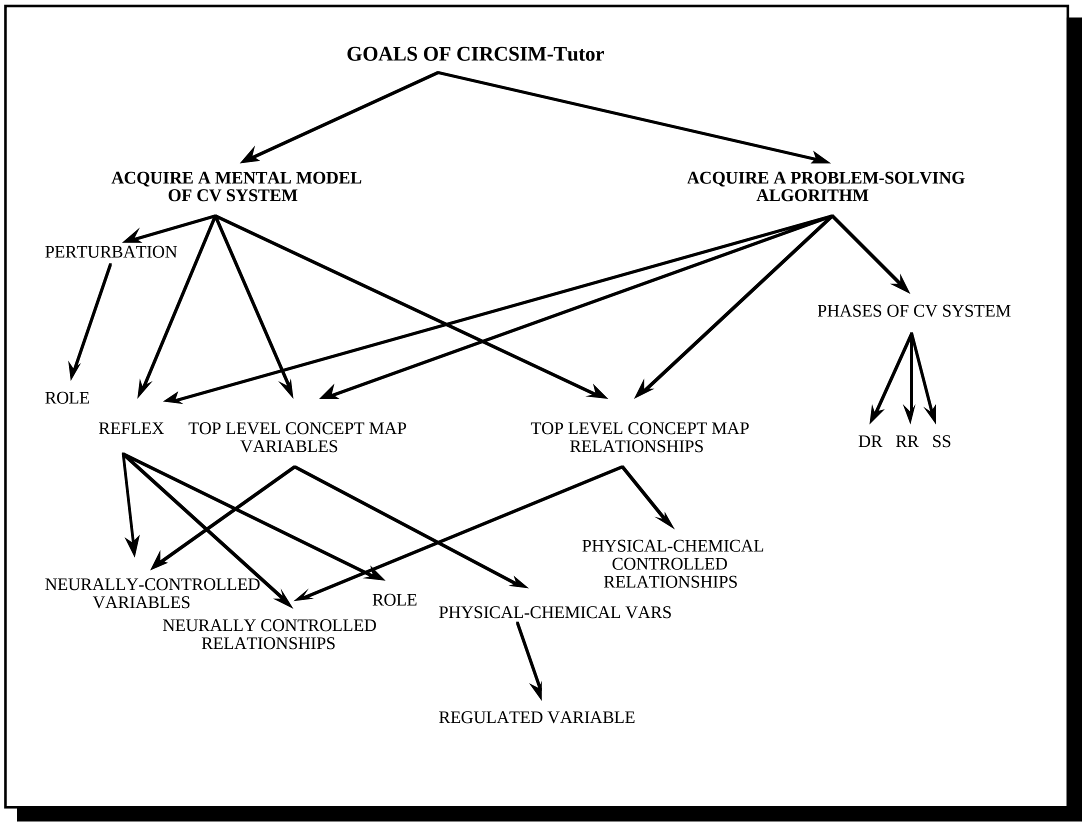

GOALS OF CIRCSIM-Tutor
ACQUIRE A MENTAL MODEL OF CV SYSTEM
ACQUIRE A PROBLEM-SOLVING ALGORITHM
PERTURBATION
PHASES OF CV SYSTEM
ROLE
REFLEX
TOP LEVEL CONCEPT MAP VARIABLES
TOP LEVEL CONCEPT MAP RELATIONSHIPS
DR
RR
SS
PHYSICAL-CHEMICAL CONTROLLED RELATIONSHIPS
NEURALLY-CONTROLLED VARIABLES
ROLE
PHYSICAL-CHEMICAL VARS
NEURALLY CONTROLLED RELATIONSHIPS
REGULATED VARIABLE

Figure 7.20 A Partial Network of Core Topics

Section 6.4 describes a view of the student as seen by the tutor. The last level of this view contains student difficulties. In order to consider a student difficulty, the tutor needs to use a set of topics to develop a plan for interaction with the student. In other words each student difficulty has pointers to various domain topics. While considering these topics, it is possible for the tutor to traverse the topic network and select some other topics that are related to the current situation. For example, while discussing an error in HR in DR the tutor can stretch the neural variable topic to discuss TPR and IS (this could be done by using the cotopic relationship). By doing this the tutor will be in a position to determine the student’s knowledge about the role of the controlled variables in baroreceptor reflex. Hence the topic network is a very useful store of structured goals that the tutor can use to get, dynamically, the ingredients for a discussion with the student. Figure 7.20 shows a partial network of core topics.

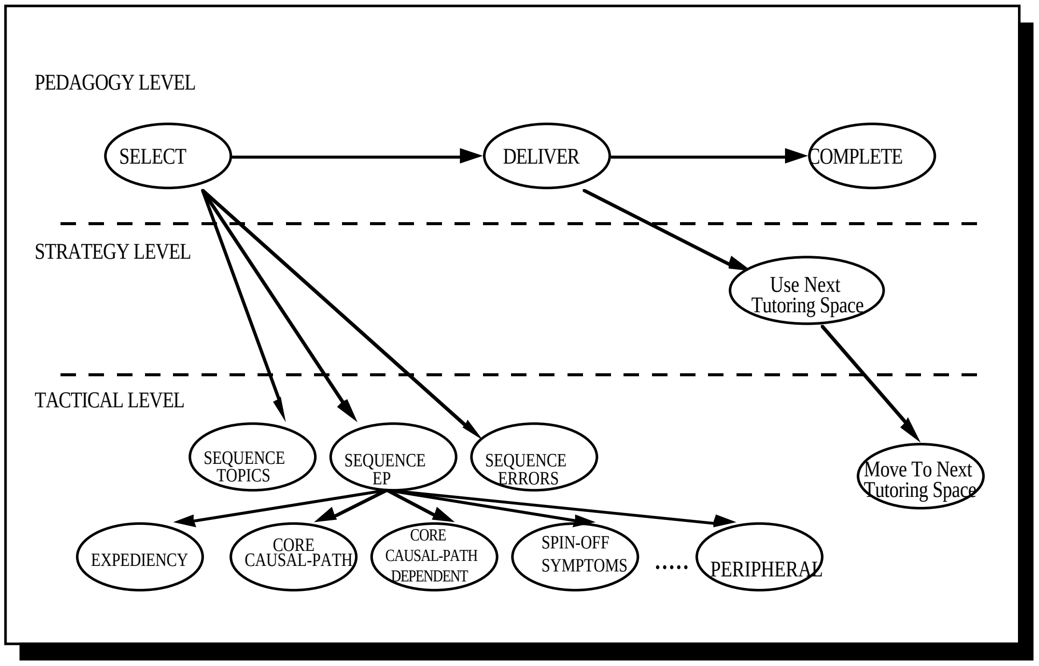


Figure 7.21 Tutoring States for the Tutoring Episode Space

7.5.5.2 <u>Generating tutoring Episodes</u>. In this section we will describe the behavior of the tutoring episode space. Figure 7.21 shows the tutoring states used in this space. This space assumes that modeler has created lists of errors, error patterns, and student difficulties. See Appendix C for a list of error patterns and student difficulties for CIRCSIM-Tutor (v.3). The first thing it does is to access these lists from the student model. Once again the pedagogy level performs the same set of functions as in other tutoring spaces. At the strategy level this space sequences error patterns, errors, and topics. First preference is given to error patterns. This sequencing operation is achieved by considering various tactics at the tactical level. Next, this space selects an error pattern from a list of sequenced error patterns. If this error pattern is pointed to by multiple errors then this space sequences and selects an error for this error pattern.

DR	RR	SS
1 Expediency	1 Expediency	1 Neural (Clamped)
2 Core Causal Path	2 Core Causal Path	2 Neural (Non-Clamped)
3 Core Causal Path Dependent	3 Core Causal Path Dependent	3 MAP-SS
4 Multiplicative Relationship	4 Multiplicative Relationship	4 Core Causal Path
5 Spin-Off Symptoms	5 Spin-Off Symptoms	5 Core Causal Path Dependent
6 Peripheral	6 Peripheral	6 Multiplicative Relationship
7 Prediction Sequence Violation	7 Prediction Sequence Violation	7 Algebraic
		8 Spin-Off Symptoms
		9 Peripheral
		10 Prediction Sequence Violation

Figure 7.22 Ordering Tactics Used in the Tutoring Episode Space (here these are classified according to CV phases and are arranged from highest to lowest default priority ranking)

Once the functions of the “select” state are finished the system makes a transition to the “deliver” state. Here this space invokes the next space to perform the next set of tasks for tutoring.

The tutoring episode space is invoked periodically until all errors are remediated. If time permits then this space tries to continue selecting error patterns until all are discussed/remediated. If time still permits then this space considers the topic network as a basis for discussing domain topics for tutoring. Next we describe in detail different tactics that this space uses to sequence error patterns. Figure 7.22 lists a summary of these tactics.

I <u>Expediency</u>. This tactic selects error patterns that are, from the tutor’s point of view, easy to get out of the way. This does not necessarily mean that error patterns selected by this tactic are not serious problems but rather these are relatively isolated from other problems and hence easy to remediate. Also tutoring on these first would help the remediation process for other misconceptions of the student. This tactic is often used in the DR and RR phases of the system. It has the highest priority, i.e., the error patterns selected by considering this tactic are planned first for tutoring with the student. An example error pattern that is considered by this tactic is “non primary neural vars.”

II <u>Core Causal Path</u>. This tactic is used in all three phases of CV system. It orders error patterns that are on the core causal path. In DR, a core causal path is the most direct path from the primary to the regulated variable. Figure 7.23 shows all possible core causal paths in the top level of concept map. An error pattern that is at the beginning of the core causal path has higher priority than an error pattern that is at the end of this path. For example, for a CV procedure in which CVP is the primary variable, the core causal path consists of following relationships: CVP -> SV, SV -> CO, CO -> MAP. Now if the student has made errors in SV and MAP then along with other error patterns, following two error patterns are sensitized: CVP -> SV, CO -> MAP. In this case CVP -> SV has higher priority than CO -> MAP, because CVP -> SV comes earlier than CO -> MAP on the core causal path.

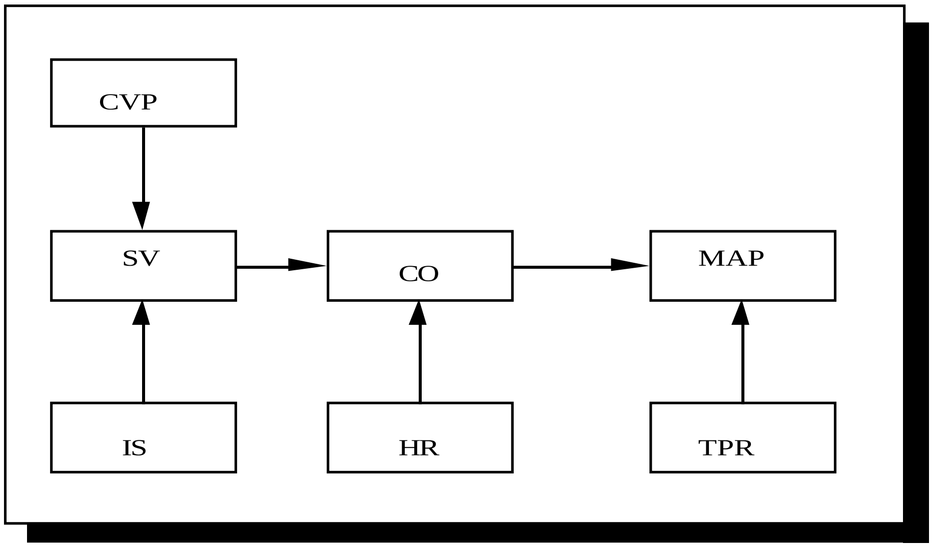


Figure 7.23 Core Causal Paths for DR at the Top Level of Concept Map

In RR, following rules apply to develop a core causal path at the top level of concept map.

If HR is not clamped
Then core causal path is: HR -> CO -> MAP

If HR is clamped
Then path is: IS -> SV -> CO -> MAP

If HR and CC are clamped
Then path is: TPR -> MAP

III <u>Core Causal Path Dependent</u>. This tactic is also used in all three phases of the CV system. Each error pattern representing a causal relationship consists of at least two CV parameters. If only one of the CV parameters of an error pattern falls on the core causal path then this tactic considers that error pattern for tutoring. For example, for a CV procedure in which HR is the primary variable, the core causal path consists of the following relationships: HR -> CO, CO -> MAP. If the student has made an error in CVP then along with other error patterns CO -> CVP is also sensitized. This error pattern is not considered by the core causal path tactic because CVP is not on the core causal path. But the core causal path dependent tactic will consider this error pattern for tutoring.

IV **<u>Multiplicative Relationships</u>.** This tactic considers error patterns that involve more than two CV variables, e.g., MAP = CO x TPR. This tactic has low priority compared to the tactics described above because three variable relationships are more difficult to understand compared to the two variable relationships. Hence two variable relationship error patterns are considered first. If there are several multiplicative relationships then these are ordered, again according to the core causal path tactic. For example if two relationships: CO = SV x HR and MAP = CO x TPR are sensitized then this tactic will first consider CO = SV x HR equation because here CO falls first on the core causal path before MAP.

V **<u>Spin-Off Symptoms</u>.** This tactic considers error patterns that surface as a result of some tutoring episode. It is the student model that flags these error patterns. These error patterns are considered for tutoring by this tactic only if the student's global assessment is high. For example if the student's predictions for CVP and SV are incorrect but the relationship between them is correct then the error pattern CVP -> SV is not sensitized. As soon as tutoring is done and the error in CVP is corrected, the error pattern CVP -> SV is sensitized by the student modeler (because this relationship is now incorrect). The spin-off symptoms tactic can select this error pattern for tutoring, depending upon the global assessment of the student.

VI **<u>Peripheral</u>.** All error patterns that are not considered after using the above five tactics are classified as peripheral. This tactic selects these error patterns only if time permits. These error patterns are not essential considering the current stage of tutoring or knowledge state of the student, hence they have been assigned a low priority rating for tutoring. For example, consider that this is the DR phase of the second procedure and the MAP = TPR x CO error pattern has been sensitized by the student modeler. In this case, if this error pattern was selected by the system in the first procedure and the current state of the student model indicates that the student has a good understanding of the multiplicative relationship between MAP, TPR, and CO then for the current procedure this error pattern will be classified as peripheral.

VII <u>Prediction Sequence Violation</u>. The domain expert of CIRCSIM-Tutor (v.3) solves a CV problem by predicting qualitative values for different CV variables in the concept map. These variables are predicted in a sequence. The student while solving a CV problem may or may not follow this sequence. It is the responsibility of the student modeler to keep track of the sequence violations committed by the student. A sequence violation by the student may or may not point to a problem in the student's understanding. It is for this reason that this tactic has the lowest priority in the tutoring episode space. For example in a CV problem where HR is the primary variable the student might after predicting the value for HR, predict IS. It is possible that here the student is trying to predict values of all neural variables first before proceeding in the problem. In this case the sequence violation does not convey any information to the tutor. On the other hand, it is quite possible that the student has a serious misconception about the functioning of neural variables. In this case the tutor may plan a tutoring episode to discuss this issue with the student. Our tutors rate the probability that the student is trying to predict all neural variables as considerably higher than the possibility of a serious underlying confusion, in the absence of other information.

VIII <u>Tactics to Order Error Patterns in SS</u>. A list of tactics that are used in the SS phase are shown in Figure 7.22. In this section we describe four tactics that order error patterns in SS. The first tactic selects all error patterns that involve clamped neural variables. This tactic has the highest priority, hence all error patterns selected by this tactic are tutored first. The second tactic collects all error patterns that involve nonclamped neural variables. Our tutors think that it is easier to understand the functioning of non-clamped neural variables than clamped neural variables. The third tactic selects error patterns involving MAP-SS. The last tactic orders error patterns that violate algebraic rules (see Section 5.5.1).

7.5.6 Tutoring Hypothesis Space: Handling Student Difficulties. This is the second subspace of the lesson space (see Figure 7.4). The major purpose of this space is to create hypotheses about the underlying problems that are responsible for the student's incorrect predictions. This space assumes that the tutoring episode space already has ordered and selected an error pattern.

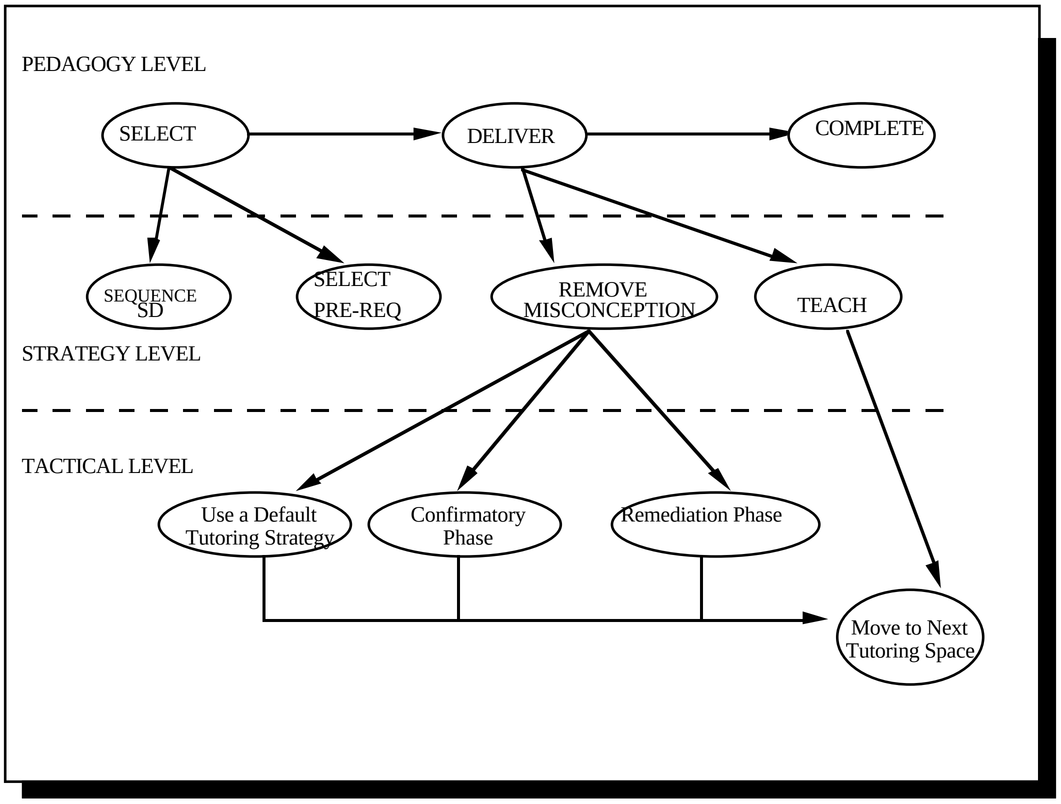


Figure 7.24 Tutoring States for the Tutoring Hypothesis Space

Figure 7.24 shows tutoring states for this space. The "select" state orders and selects student difficulties by looking at the current error pattern. Once a student difficulty has been selected, the "deliver" state is activated. At the strategy level this space considers the "remove misconception" state. This strategy is refined down to three major tactics. The first tactic engages the system in the exploratory phase (see Section 6.5). This tactic requires that the system model switch to the next tutoring space where it interacts with the student. If the selected student difficulty does not require this tactic then the next tactic "remediation phase" is invoked. If a single student difficulty is available for consideration then this space uses a default tutoring method to tutor the student. Instead of an error pattern, if the tutoring episode space has selected a domain topic on which to tutor the student, then at the "select" state, the tutoring hypothesis space tries to discover whether the student requires prerequisite knowledge before discussing the selected domain knowledge. The "deliver" state in this case considers the "teach" strategy, which in turn pushes the system to consider next tutoring space.

7.5.7 <u>Tutoring Issue Space: Handling Communication with the Student</u>. This is the third subspace of the lesson space (see Figure 7.4). This space assumes that the tutoring hypothesis space has already selected a student difficulty. The major purpose of this space is to organize a tutoring interaction with the student around the selected student difficulty. This space is more like the DMN of MENO-Tutor (Woolf, 1984) and the discourse planner of CIRCSIM-Tutor (v.2) (Woo, 1991). Figure 7.25 shows the tutoring states used in this space.

The functioning of the pedagogy level here is the same as in the other spaces of the system model. The "select" state, considering the selected student difficulty, collects a set of topics. These topics are then used by the "deliver" state to communicate with the student. The "complete" state here merely finishes the operations of this space.

The "deliver" state uses various strategies and tactics to plan an interaction with the student. At the strategy level this space uses a cycle. This cycle mimics the behavior of our tutors in the keyboard-to-keyboard sessions. In this cycle the system introduces a new topic or makes decision about continuing the same topic. Then it evaluates the student's response and finally it responds to student's input. This cycle is repeated as long as the system continues interacting with the student.

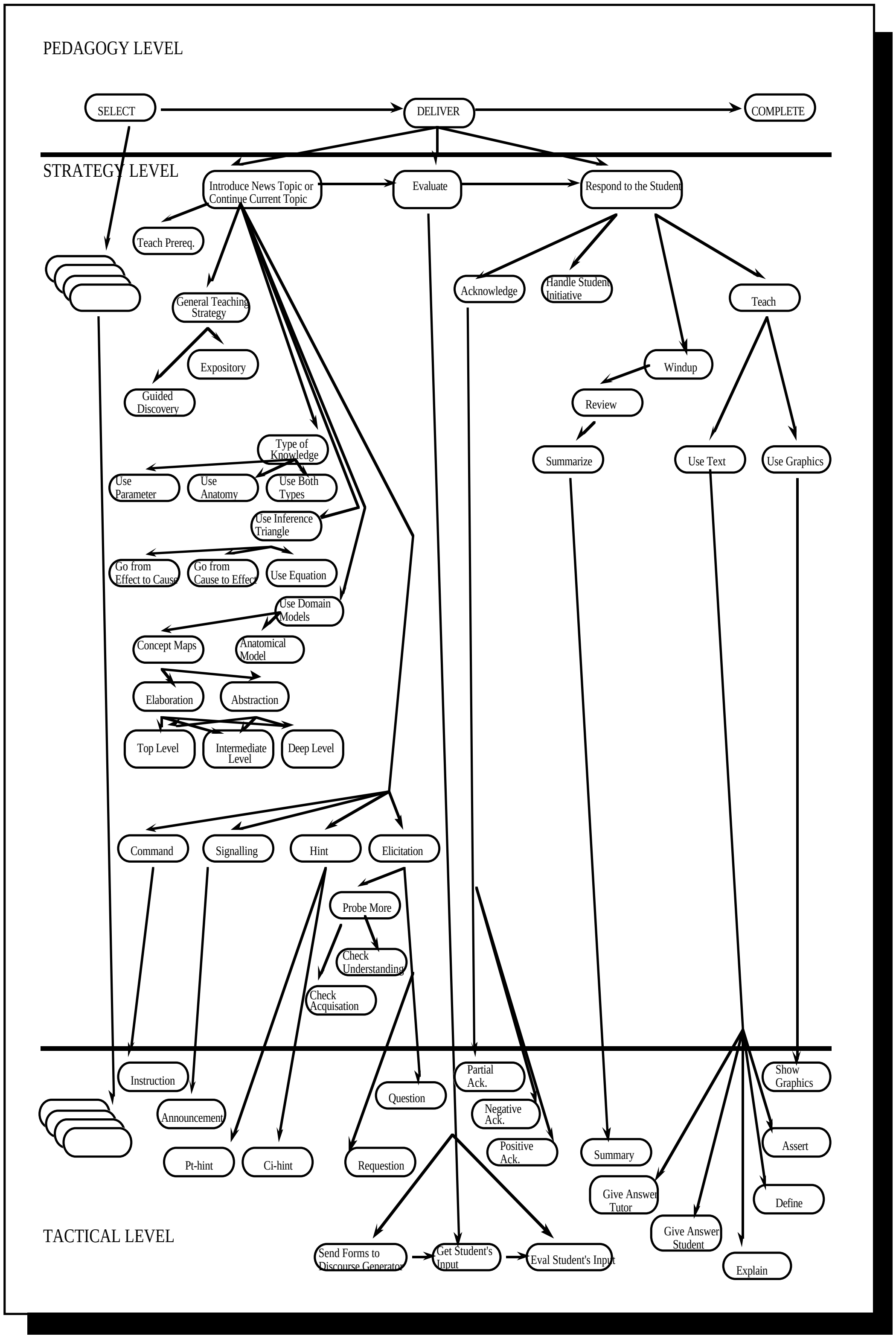


Figure 7.25 Tutoring States for the Tutoring Issue Space

One of the essential functions of this space is to make decisions about *how* to interact with the student. This function is achieved by considering a set of strategies and tactics. These strategies, for example, decide on the general teaching strategy, the type of knowledge that needs to be communicated, the method of use of the inference triangle, and the type and level of domain models that need to be used. In addition, this space also uses communication strategies to help generate utterances for the student in English.

7.6 System Model: A Step Towards a Generic ITS

In this chapter we have discussed the system model for CIRCSIM-Tutor (v.3). This model results out of the system phase of the ITS development framework described in Chapter IV. This model is based upon the system issues that need to be considered in order to develop an ITS. These issues can be related to context dependent or to context independent factors. It is these context independent factors that make our model a generic model that could be used for many domains.

This model is based on Lesgold's framework that distinguishes between curriculum and domain knowledge. But unlike this framework, the system model also makes the planning information explicit. The major theme behind this model is that it attempts to integrate curriculum and model-based ITS designs.

CIRCSIM-Tutor's domain knowledge is organized into models. The earlier versions of CIRCSIM-Tutor were purely model-based systems. These versions suffered from the common problem that there were no distinction between the domain and the curriculum in the system. As Lesgold (1988) noted, this is the most common problem in the majority of the ITS's developed to date. The system model described here has removed this problem by explicitly distinguishing between the planning knowledge, the curriculum knowledge, and the domain knowledge. Making explicit the curriculum knowledge has provided a vehicle for the system to explicitly reason about the goals of the system. Here the system has a mean of monitoring the progress of the student towards achieving the goals of the system. The planning knowledge allows this model to be driven by a number of different conceptual models of tutoring. I have generalized Woolf’s (1984) DMN to act as the planning mechanism for the system model. Due to the nature of the knowledge organization in the system model, it is perfectly feasible to use some other planning mechanism than that used for CIRCSIM-Tutor (v.3). The domain knowledge of the system model contains information in the form of the CV model that it uses to communicate with the student. Unlike the Lesgold’s framework, the system model does not restrict the system to use only the overlay model for the student. In CIRCSIM-Tutor (v.3) the bug library method of student modeling has been given priority to mimic as much as possible the behavior of tutors in the keyboard-to-keyboard sessions.

CHAPTER VIII

ARCHITECTURE OF CIRCSIM-TUTOR (V.3): IMPLEMENTING THE DOMAIN AND THE PEDAGOGY EXPERTS

8.1 Introduction

The system model (see Chapter VII) is still distant from the design needed to directly support the implementation (or physical) phase (see Chapter IV). The system model is at the same level of abstraction as the conceptual model. In the ITS development framework described in Section 4.1, the second subphase in the system phase emphasizes the software system point-of-view. This subphase brings the ITS development one step closer to its physical realization as a computer program. Here the developer views the ITS as a software system. In this subphase software engineering principles shape the system model into a coherent architecture. The key issue here is to keep this architecture independent of the formalism used to realize the system (e.g., a general purpose programming language).

In this chapter we will first describe a general architecture for CIRCSIM-Tutor (v.3) (Khuwaja et al., 1994a). After this we will describe the design and implementation of the components of this architecture that are related to my research work. These components implement the pedagogy and the domain experts of CIRCSIM-Tutor (v.3) (see Chapter VI).

8.2 Architecture of CIRCSIM-Tutor (v.3)

The architecture of CIRCSIM-Tutor (v.3) is shown in Figure 8.1. This architecture divides its components into two major classes: modules (or subsystems) and information stores. Modules are active processes that communicate and coordinate to create the required intelligent behavior for the system. For example, in Figure 8.1, the instructional planner is a module of CIRCSIM-Tutor (v.3). As the name implies, an information-store is a store of information/knowledge. Each information store has an interface through which the modules of the system access pieces of information. In

Figure 8.1, the Domain Knowledge Base (DKB) is an information store and the Domain Problem Solver (DPS) is an interface to it. In actuality, the DPS is more than an interface. It also provides the mechanisms for domain inferencing (see Section 8.6 for more details).

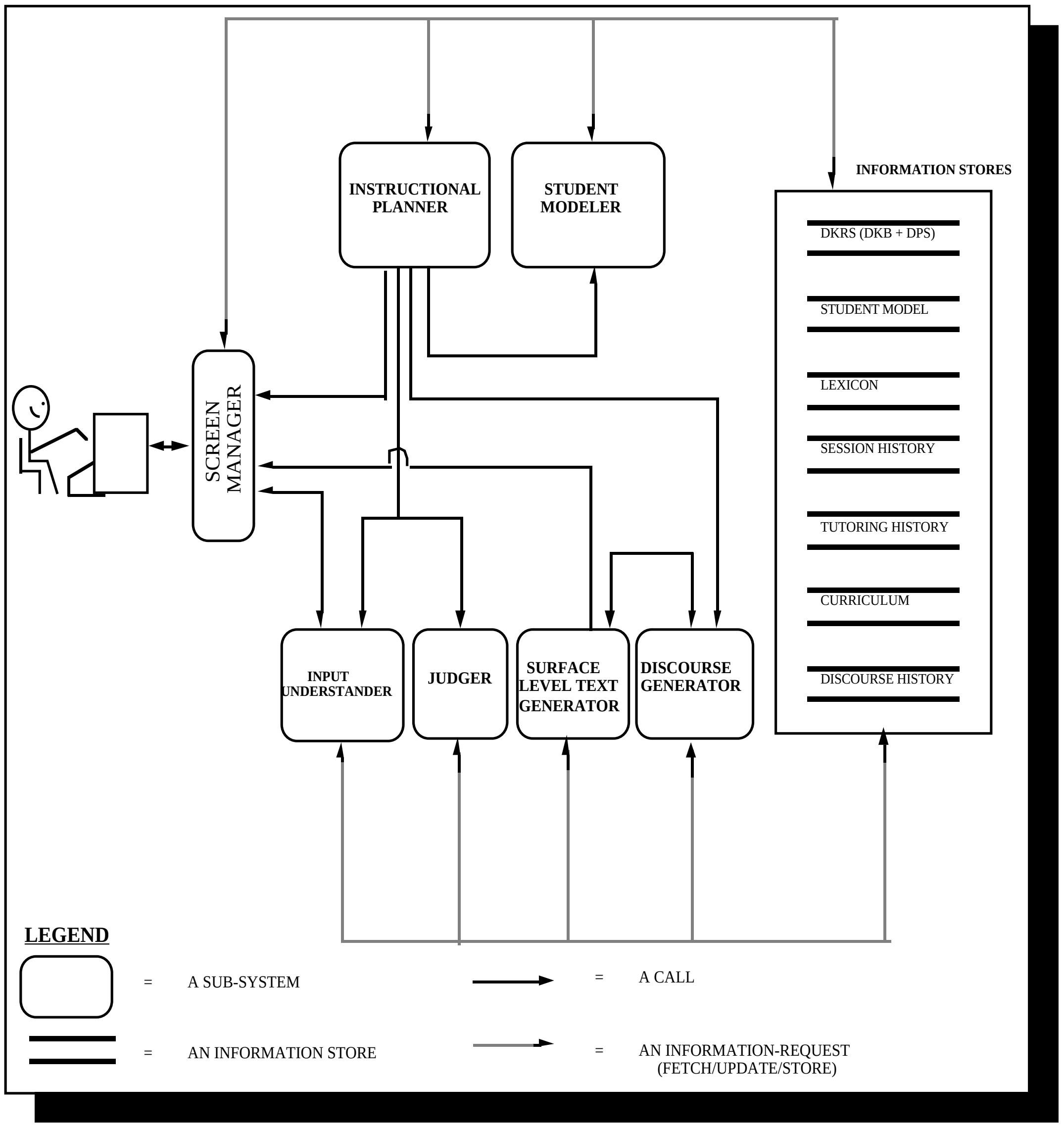


Figure 8.1 Architecture of CIRCSIM-Tutor (v.3)

This architecture does not constrain the communication protocol adopted for the system. For example, for a completely decentralized architecture, it allows communication between any two modules of the system. On the contrary, for a strictly centralized architecture it offers a central module that controls/monitors the communication traffic between the modules and information stores of the system.

Two types of messages are distinguished in this architecture. The first one is called a call. A call is a message that is generated by a module to communicate with another module in the system. The second type of message is called an information request. An information request is a message that is generated by a module to fetch/update/store a piece of information from/to an information store. Currently a decentralized approach has been selected for CIRCSIM-Tutor (v.3) so that it can be changed in future without much effort.

One of the attractive characteristics of this architecture is that it supports the notion of recursive architecture, i.e., each module of this architecture can be developed using the same architecture. In this case this architecture supports a layered design that greatly increases the system's modularity and flexibility.

8.3 Architectural Equivalence of the Domain and the Pedagogy Experts

In the ITS development framework, described in Chapter IV, each phase yields a different model (see Figure 4.1). The conceptual phase yields a conceptual model. This model must be transformed into a system model in the first subphase of the system phase. The system model must then be transformed into an architecture at the second subphase of the system phase. The resulting components of the architecture are then coded into pieces of software.

The conceptual models of the domain and the pedagogy experts are described in Chapter VI. These models have been transformed into a system model, as described in Chapter VII. In this chapter we will describe the transformation of the system model into architectural components and their implementation in CIRCSIM-Tutor (v.3).

The system model described in Chapter VII has been transformed into one subsystem and three information stores (see Figure 8.2). The DKRS stands for the

Domain Knowledge Representation System. It is an information store in the architecture of CIRCSIM-Tutor (v.3) (see Figure 8.1). This store consists of two parts, the DKB (Domain Knowledge Base) and the DPS (Domain Problem Solver). Sections 8.5 and 8.6 describe these two parts in detail, respectively. The DKRS is the system embodiment of the domain expert (see Figure 8.2).

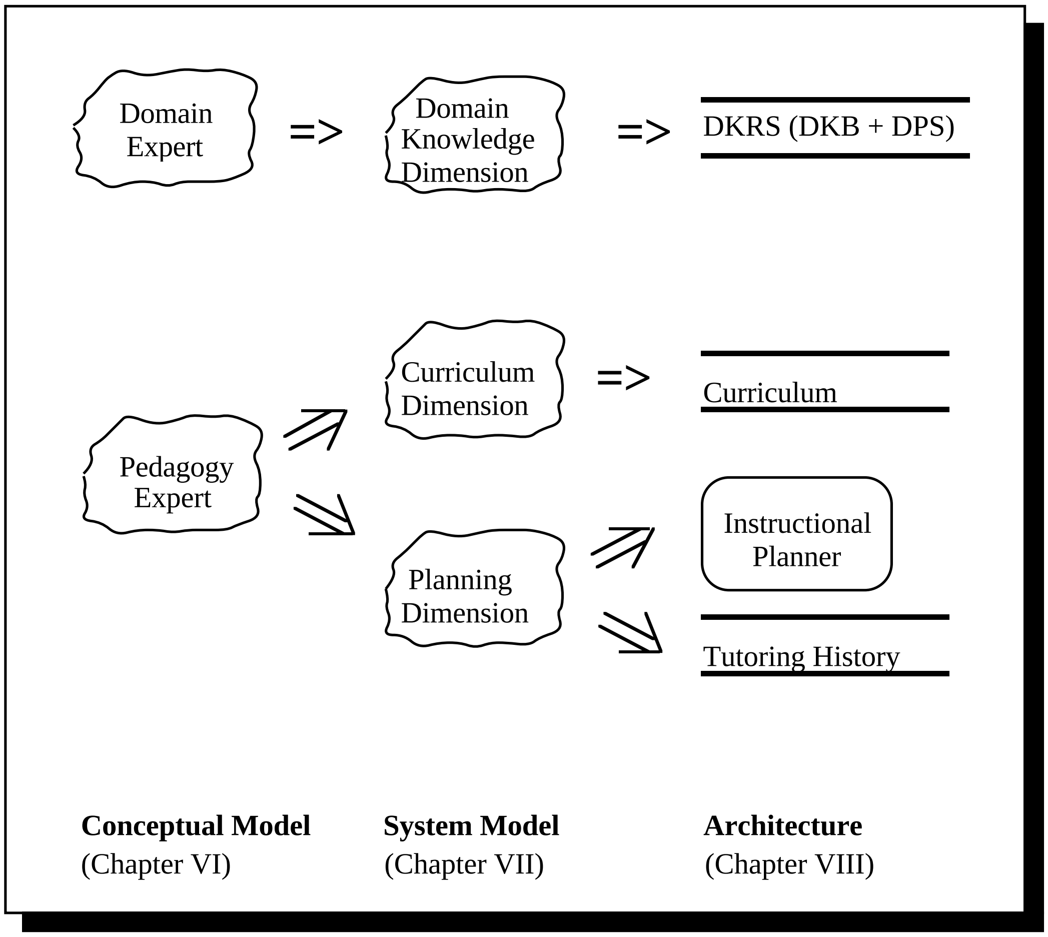


Figure 8.2 Outcome of Phases of the ITS Developmental Framework for the Research Described in this Thesis

The pedagogy expert is transformed into one subsystem, the Instructional Planner, (see section 8.7) and two information stores, curriculum and tutoring history (see Sections 8.8 and 8.9).

8.4 Object-Oriented Methodology: Developing and Implementing the Components of Architecture of CIRCSIM-Tutor (v.3)

In order to design, develop, and implement some of the important components of the architecture of CIRCSIM-Tutor (v.3) (see Figure 8.2), I have used an object-oriented methodology. This methodology, nowadays, is very popular in the development of software systems. In this section we will briefly describe this methodology and the way it has been used in my research.

In the object-oriented methodology, a software system is developed using the class and object as basic building blocks (Booch, 1991). One of the major activities in this methodology is to develop an object model for the target software system. Usually three different stages have been used to develop this model. The first stage is called object-oriented analysis (OOA). According to Booch (1991):

> Object-oriented analysis is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain (p. 37).

The second stage is called object-oriented design (OOD). According to Booch (1991):

> Object-oriented design is a method of design encompassing the process of object-oriented decomposition and a notation for depicting both logical and physical models as well as static and dynamic models of the system under design (p. 37).

The final stage is called object-oriented programming (OOP). This stage is defined as:

> A method of implementation in which programs are organized as a collection of cooperating objects each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships (p. 36).

According to Booch (1991) these three stages are related as follows.

> Basically, the products of object-oriented analysis can serve as the models from which we may start an object-oriented design; the products of object-oriented design can then be used as blueprints for completely implementing a system using object-oriented programming methods (p. 37).

I have used a version of Booch’s (1991) methodology. The knowledge engineering methodology (see Chapter IV) used at the conceptual and the system phases of my research, has provided an alternative to the object-oriented analysis. I did object-oriented design to develop the architecture for the components of CIRCSIM-Tutor (v.3) shown in Figure 8.2. During the physical phase (see Section 4.1) I did object-oriented programming to implement these architectural components as a software program. I used CLOS (Common Lisp Object System) (Keene, 1989) of Procyon Common Lisp for this purpose. The hardware platform on which this programming activity was performed is the Apple Macintosh.

The following steps were used to design and implement each architectural component of Figure 8.2. In the first step a hierarchy of classes was developed. In the second step objects representing the instances of different classes are created. The third step deals with connecting various objects based upon the knowledge captured in the conceptual and the system models. In the fourth step each object is assigned its associated behavior(s) so that it can participate in the functioning of the software system being developed. In the following sections we will describe the object-oriented design and implementation for each of the architectural components shown in Figure 8.2.

8.5 <u>Domain Knowledge Base: Providing General Knowledge About the CV System to CIRCSIM-Tutor (v.3)</u>

The domain knowledge base (DKB) is a store of factual knowledge about the CV system. It is responsible for providing general domain knowledge (see Section 6.8) to CIRCSIM-Tutor (v.3). The DKB contains information about the three levels of concept maps (see Section 6.11) and the anatomical model of the CV system (see Section 6.12). This knowledge is accessible to the rest of the system via the domain problem solver (see section 8.6 for more details).

In the architecture of CIRCSIM-Tutor (v.3) the DKB is characterized as an information store (see Figure 8.1). The four steps described in Section 8.4 are used to develop this store. Figure 8.3 shows a partial hierarchy of classes that are used in the DKB. This figure also contain example instances (or objects) of these classes. Each object of the DKB has a current state, exhibits some well-defined behavior, and has a unique identity.

> The state of an object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties. . . Behavior is how an object acts and reacts, in terms of its state changes and message passing. . . Identity is that property of an object that distinguishes it from all other objects (Booch, 1991, pp. 78-84).

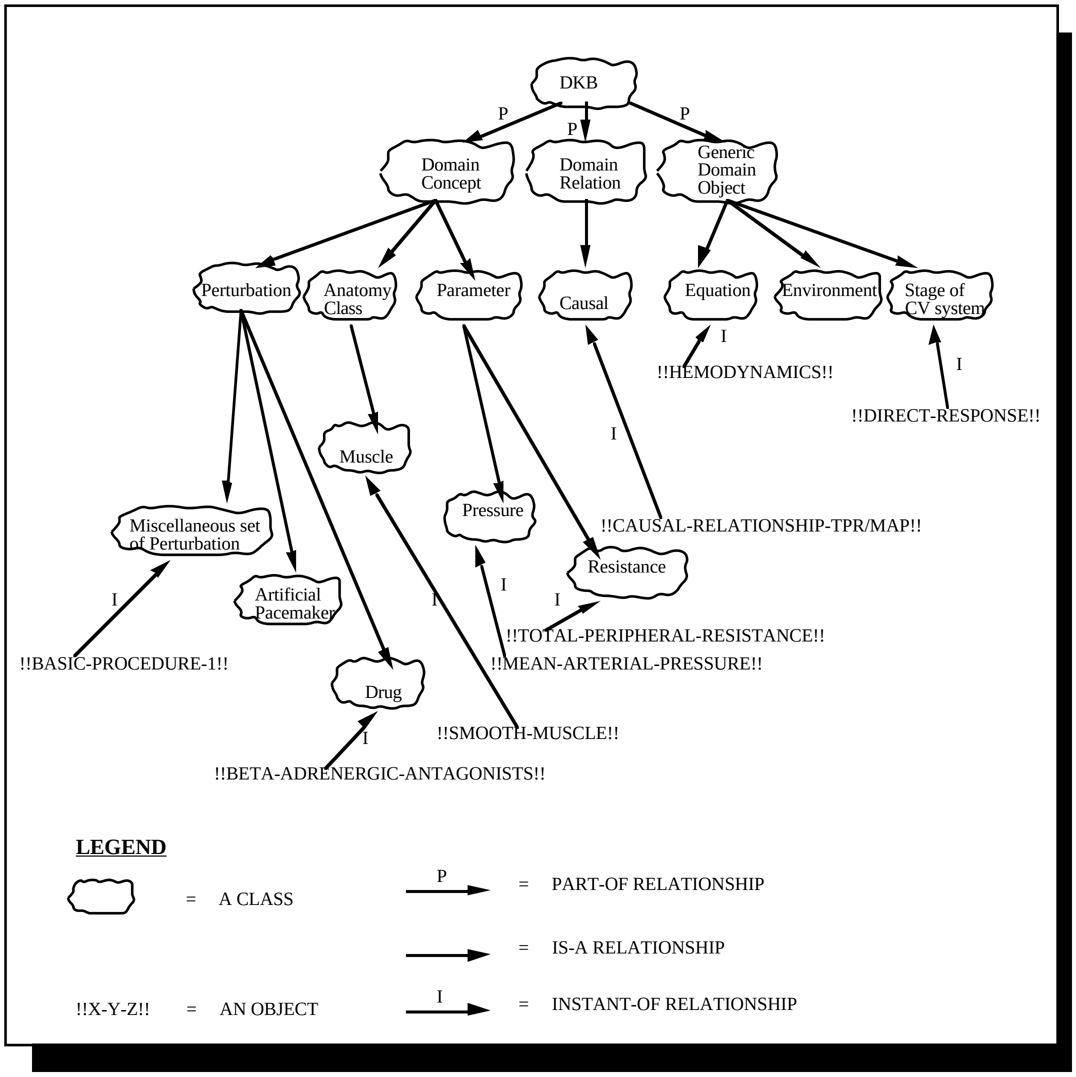


Figure 8.3 A Partial Hierarchy of Classes Used in the Domain Knowledge Base

Figure 8.4 contains a template showing the state of an example DKB object. When all objects of DKB are connected then they form general models of the domain. Figures 6.15 (a), 6.18, 6.19, and 6.22 show schematic views of these models.

```
OBJECT-IDENTITY
        !!MEAN-ARTERIAL-PRESSURE!!

TEMPLATE
NAME:               MEAN-ARTERIAL-PRESSURE
DEFINITION:         "the average pressure in the arteries over time"
SYNONYMS:           (AFTER-LOAD REGULATED-VARIABLE)
LEVEL-OF-CONCEPT-MAP:
                    (TOP INTERMEDIATE DEEP)
EQUATION:           (!!HEMODYNAMICS!!)
HAS-PART:           NIL
STATE:          #<STATE-OF-A-PARAMETER #x1948A0> ; A pointer to an object.
ASSOCIATED-WITH:
                    (!!ARTERIAL-SYSTEM!!)
PART-OF:            NIL
CAUSAL-RELATION:
                    (!!CAUSAL-RELATION-MAP/BRP!!
                    !!CAUSAL-RELATION-ABV/MAP!!
                    !!CAUSAL-RELATION-MAP/SV!!
                    !!CAUSAL-RELATION-CO/MAP!!
                    !!CAUSAL-RELATION-TPR/MAP!!)
ABBREVIATION:       MAP
NATURE-OF-REGULATION:
                    PHYSICAL-CHEMICAL
TONIC-ACTIVITY:     NIL
UNIT:                (MM-OF-HG)
```

Figure 8.4 A Domain Knowledge Base Object

8.6 Domain Problem Solver: Providing Access to the Domain Knowledge and Inferencing about it for CIRCSIM-Tutor (v.3)

The domain knowledge representation system (DKRS) is an information store in the architecture of CIRCSIM-Tutor (v.3) (see Figure 8.1). This store has two parts the domain knowledge base (see Section 8.5) and the domain problem solver (DPS) (Khuwaja et al., 1993). The DPS has three purposes in the architecture of CIRCSIM-Tutor (v.3). First, it provides an interface to the DKB through which other components of CIRCSIM-Tutor (v.3) can access pieces of the domain knowledge. Second, it provides an inferencing mechanism to perform reasoning in the domain. Third, with this inferencing capability, this component is capable of solving all the CV problems that CIRCSIM-Tutor (v.3) gives the student to solve. The relationship between the DKB and the DPS is shown schematically in Figure 8.5.

The DPS supports a querying process that helps other modules of CIRCSIM-Tutor (v.3) to communicate with the DKRS. Any piece of domain knowledge (a stored fact or an inferred one) can be accessed by other parts of CIRCSIM-tutor (v.3) by composing a query and posing it to the DPS, which, depending upon the form of query, accesses or infers appropriate piece(s) of domain knowledge. Before we discuss this query process, let us describe various entities in the DPS (see Figure 8.6).

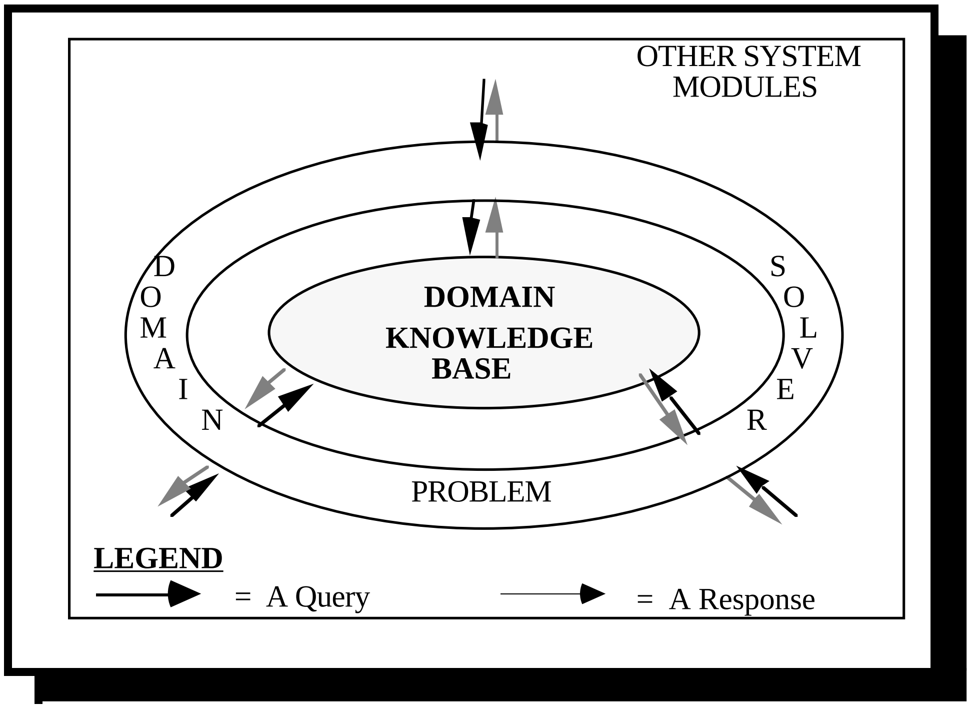


Figure 8.5 A Schematic View of the DKRS

The CLOS objects for the primitive inference functions (see Figure 8.6) of DPS represent the atomic inference functions in the domain, which determine what inferences can be made on the basis of domain relations (e.g., causal) in the DKB. The DPS also distinguishes between various inference forms. An inference form determines a type of inference on a piece of domain knowledge in the DKB. Four such inference form objects are identified in CIRCSIM-Tutor (v.3) (see Figure 8.6). The main function of the query analyzer is to extract the inference form and information about the type of knowledge source from the input query to the DPS. The inference engine uses the inference form information to perform the requested operation by invoking the appropriate inference form object.

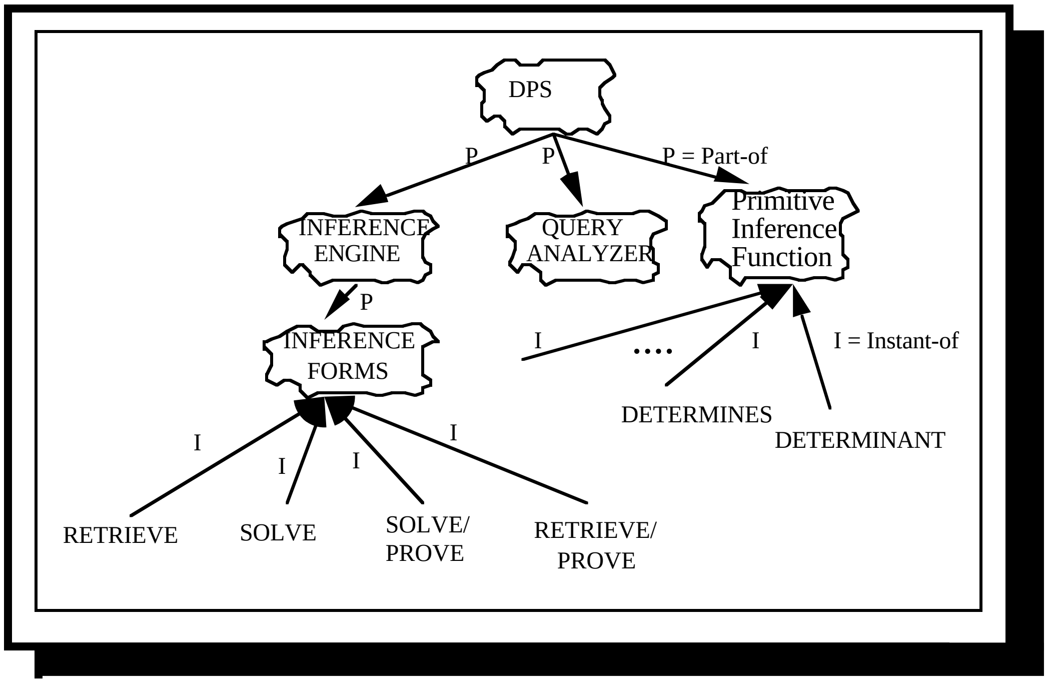


Figure 8.6 A Partial Hierarchy of Classes Used in the Domain Problem Solver

The sequence of events that takes place when the DPS is asked to perform a knowledge related task is shown in Figure 8.7. The input query to the DPS is first analyzed by the query analyzer, which extracts the inference form and information about the type of knowledge source. This information is then directed to the inference engine, which, as a response, invokes the appropriate inference form object. This object then invokes an appropriate knowledge source to perform a domain related inference operation. Finally, the response to the input query is returned to the caller. Appendix D explains different types of queries with examples that the DKRS handles in order to access/infer the domain knowledge.

The process of querying the DPS provides a unifying accessing protocol to other components of CIRCSIM-Tutor (v.3), with which they can access domain knowledge. Various inference forms allow the DPS to access both factual and inferred domain knowledge with equal ease. The knowledge in the DKRS is kept fully transparent for other components of CIRCSIM-Tutor (v.3). The unified access protocol, as a result of this transparent domain knowledge, provides full functional access to each chunk of domain knowledge in the DKRS. Also, due to the use of object-oriented methodology, the DKRS is highly modular, extensible, and maintainable.

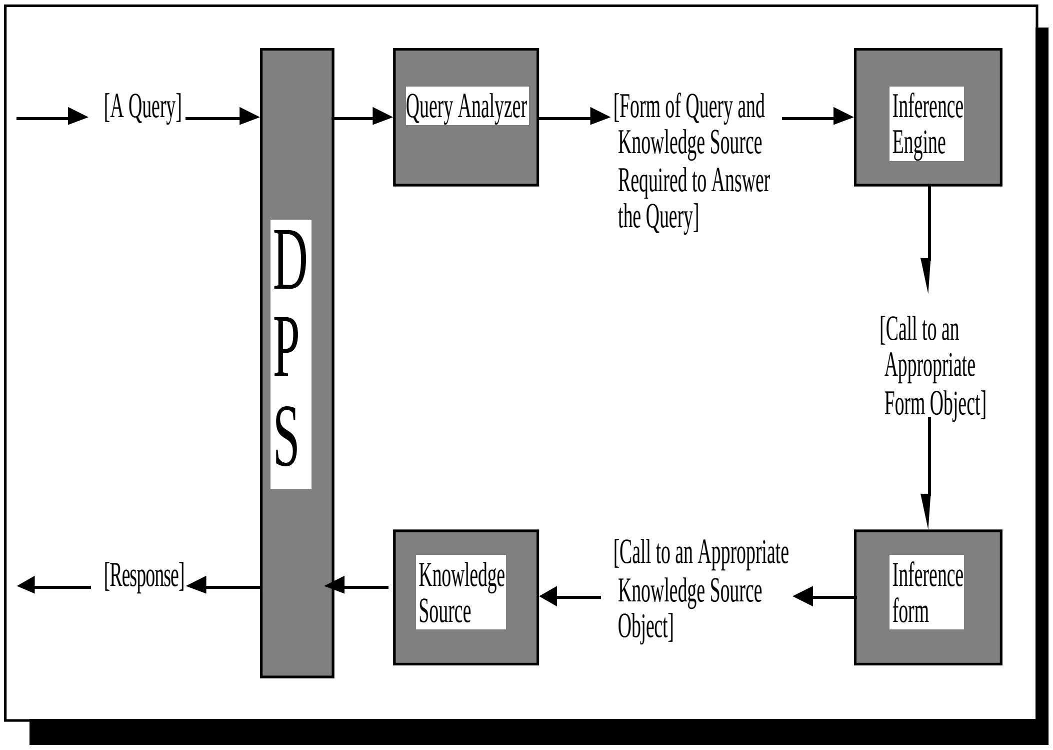

[A Query]
DPS
Query Analyzer
[Form of Query and Knowledge Source Required to Answer the Query]
Inference Engine
[Call to an Appropriate Form Object]
Inference form
[Call to an Appropriate Knowledge Source Object]
Knowledge Source
[Response]

Figure 8.7 A Flow of Events Representing the Querying Process in the DPS

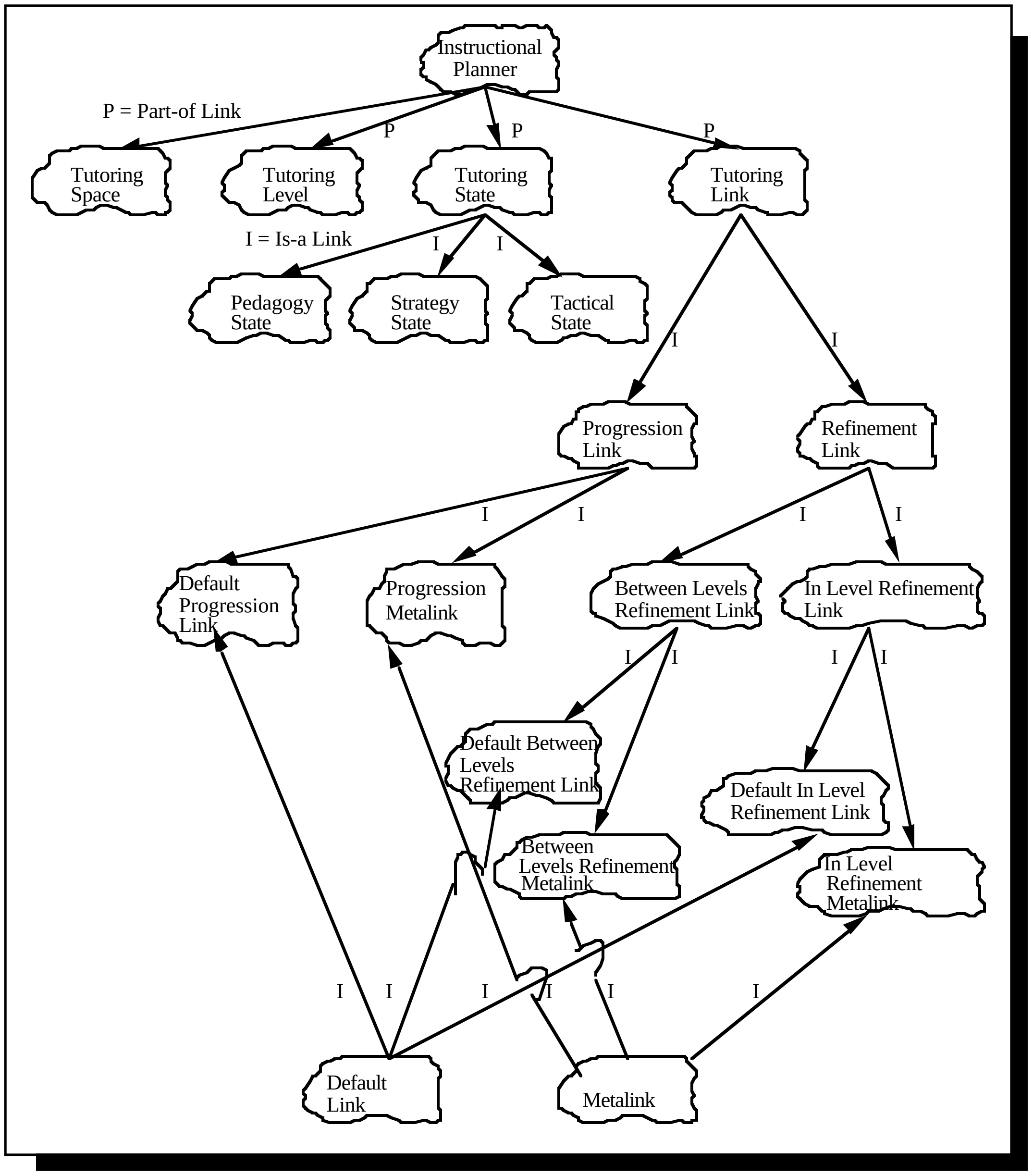


Figure 8.8 Hierarchy of Classes for the Instructional Planner

8.7 <u>Instructional Planner: Containing the Reasoning Mechanism for the Pedagogy Decision Making</u>

The instructional planner is an important module in the architecture of CIRCSIM-Tutor (v.3) (see Figure 8.1). It is an architectural equivalent of the pedagogy expert (see Figure 8.2), i.e., this module contains the reasoning mechanisms that perform the pedagogy decision making for the system. At the architectural level, the object model for the instructional planner has the same design as for the planning dimension described in Section 7.4.1. This design could be viewed as a set of tutoring spaces, each of which has tutoring levels. Each tutoring level has a set of tutoring states that are connected via a set of tutoring links. Figure 8.8 shows the class hierarchy used for the instructional planner.

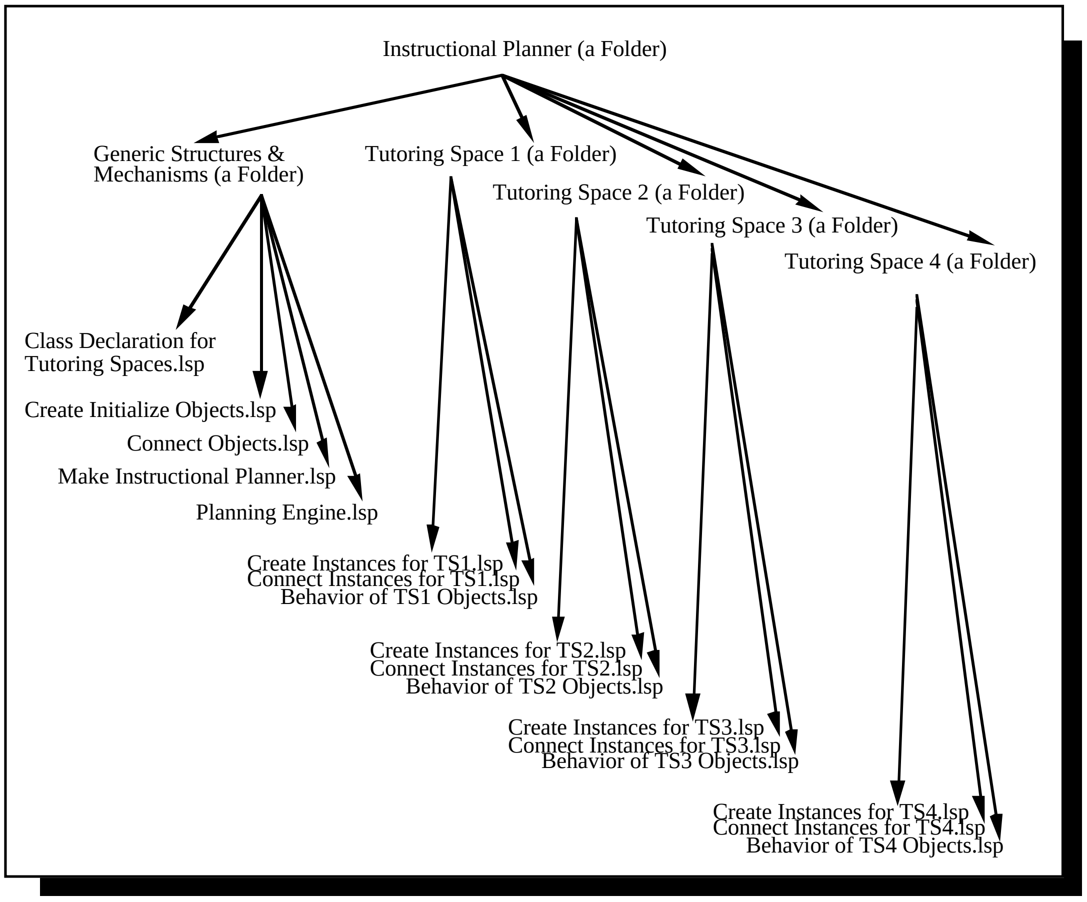

Instructional Planner (a Folder)
Generic Structures & Mechanisms (a Folder)
Tutoring Space 1 (a Folder)
Tutoring Space 2 (a Folder)
Tutoring Space 3 (a Folder)
Tutoring Space 4 (a Folder)
Class Declaration for Tutoring Spaces.lsp
Create Initialize Objects.lsp
Connect Objects.lsp
Make Instructional Planner.lsp
Planning Engine.lsp
Create Instances for TS1.lsp
Connect Instances for TS1.lsp
Behavior of TS1 Objects.lsp
Create Instances for TS2.lsp
Connect Instances for TS2.lsp
Behavior of TS2 Objects.lsp
Create Instances for TS3.lsp
Connect Instances for TS3.lsp
Behavior of TS3 Objects.lsp
Create Instances for TS4.lsp
Connect Instances for TS4.lsp
Behavior of TS4 Objects.lsp

Figure 8.9 File Organization for the Instructional Planner

The four steps described in Section 8.4 are used to develop the object model for this module. At the physical (or code) level the file format for this module is shown in Figure 8.9. The “Generic structures and mechanisms” folder (see Figure 8.9) contains files that help in creating and connecting objects for different tutoring spaces. This folder also contains code for the generic engine that provides a planning mechanism for different tutoring spaces of the instructional planner. Section 8.8 describes the functioning of this planning engine in detail.

The instructional planner is the central component of CIRCSIM-Tutor (v.3). It communicates and coordinates with other modules to create a tutoring behavior. The modules with which the instructional planner communicates are shown in Figure 8.1.

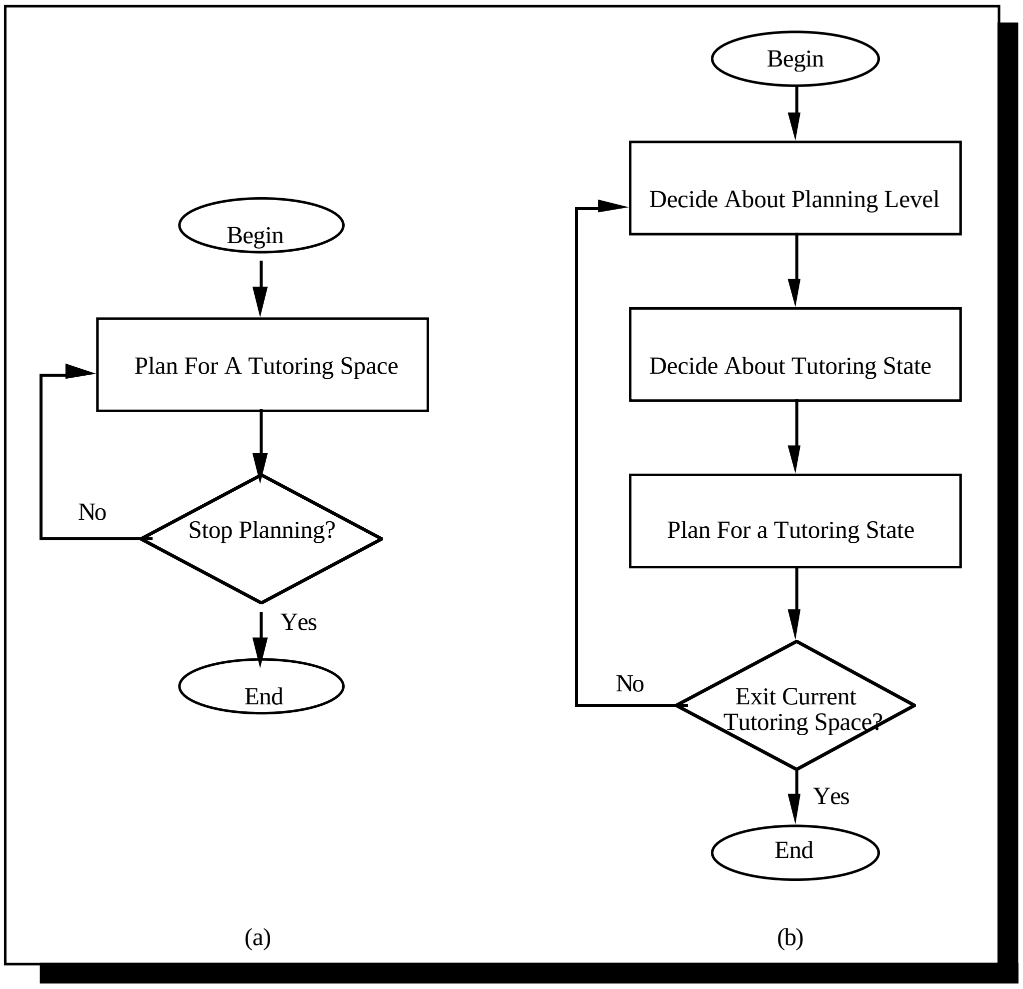

Begin
Plan For A Tutoring Space
No
Stop Planning?
Yes
End
(a)
Begin
Decide About Planning Level
Decide About Tutoring State
Plan For a Tutoring State
No
Exit Current Tutoring Space?
Yes
End
(b)

Figure 8.10 Flow Charts Representing the Higher Level Planning Behavior for the Instructional Planner

8.8 Planning Engine

The structures of the tutoring space used in the instructional planner are the same (see Section 7.5) although the contents are different. This means, technically, it is possible to use a generic planning mechanism for all tutoring spaces of the instructional planner. This is exactly the approach I have taken. This section describes the functioning of the generic planning engine that powers the decision making process of the instructional planner.

The functional behavior of this engine is shown in Figure 8.10 and 8.11 by a set of flow charts. At a very high level this engine plans for a tutoring space (see Figure 8.10 (a)). At this level it keeps on planning until some global condition forces it to stop functioning (e.g., the student quits without completing a CV problem).

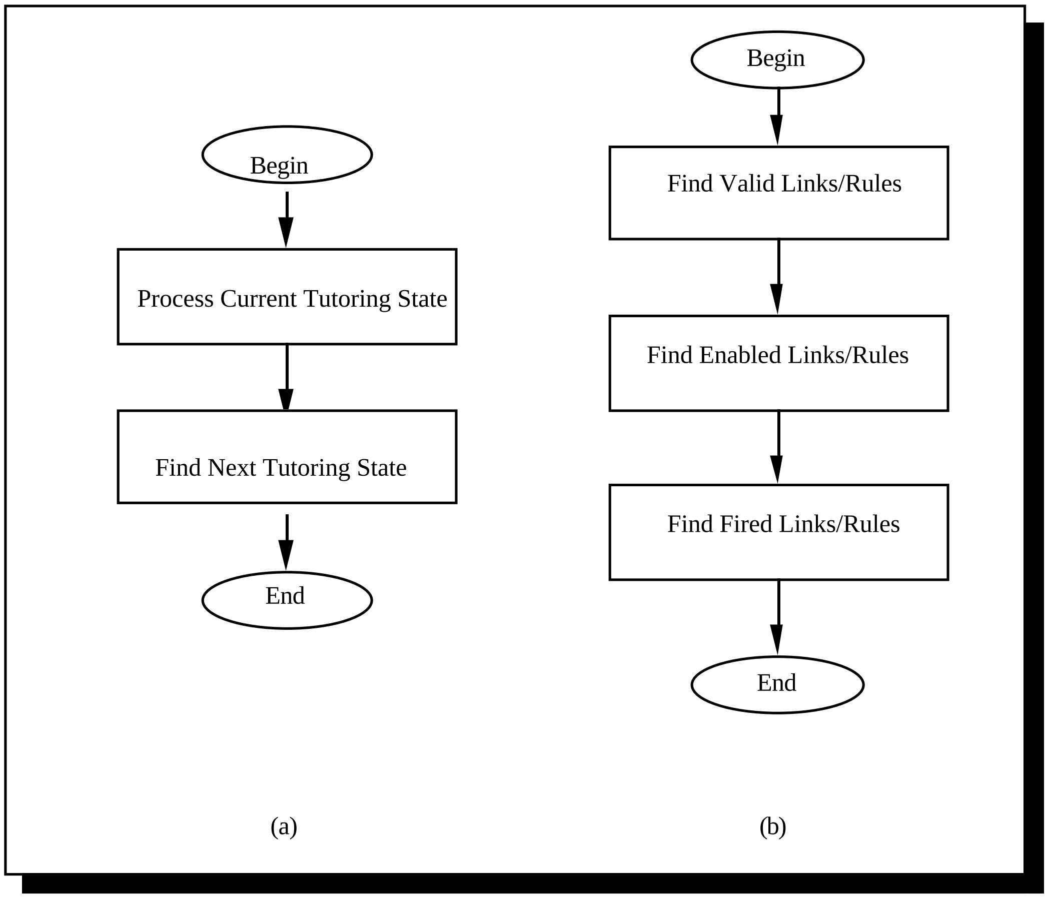


Figure 8.11 Detailed Behavior of the Planning Engine Shown in Figure 8.10

At the second level (see Figure 8.10 (b)), this engine enables the system to jump between tutoring states to perform pedagogy decision making. Basically Figure 8.10 (b) is an expansion of the box "plan for a tutoring space" in Figure 8.10 (a). At this second level of decision making the planning engine decides about the current planning level in the current tutoring space. Next it plans for the current tutoring state in the current planning level. This process is repeated until some condition forces this engine to switch the current tutoring space (e.g., as a result of the failure of the tutor's current hypothesis). In this case this engine jumps to a higher level planning loop of Figure 8.10 (a).

Figure 8.11 (a) is an expansion of the decision making process of "plan for a tutoring state" box in Figure 8.10 (b). Here the planning engine performs two major functions: it first processes the current tutoring state, and then finds the next tutoring state. In each tutoring state the instructional planner performs a tutoring action. Each tutoring action contributes to the overall behavior of the instructional planner. The first function of Figure 8.11 (a) enables the execution of the tutoring action for the current tutoring state. The second function deals with deciding about the next tutoring state that the instructional planner jumps into.

Figure 8.11 (b) is an expansion of this second function of Figure 8.11 (a). Here the planning engine performs three major functions. First, it finds a list of valid links (or rules) that could be traversed from the current tutoring state. Next, it creates a subset of these valid links containing only the links that are enabled due to the situation at hand. And finally, a link from this subset is selected that the instructional planner could use to traverse and jump into the next tutoring state. Two sets of these enabled links are created. The first set contains all enabled links that have lately been used by the instructional planner. The second set contains the enabled links that have not yet been used by the instructional planner. The selection rule used here is:

```
If the list of not lately used links is empty
Then select the first link from the recently used list
Else select the first link from the not recently used list
```

The consequence of this rule is that the instructional planner gives higher priority to a tutoring action that has not yet recently been used than to an alternative action that has recently been used. This will give the instructional planner a variety of alternative actions each time it interacts with the student.

This generic planning engine is used in conjunction with the tutoring spaces and two information stores - curriculum and tutoring history.

8.9 Tutoring Spaces

The instructional planner is composed of four major tutoring spaces: major objective space, exercise space, unit space, and lesson space (see Section 7.5). Each space consists of three tutoring levels, a set of tutoring states, and a set of tutoring links. The contents of an example tutoring state object is shown in Figure 8.12.

```
NAME:               TUTORING-PHASE/TS-3
TYPE-OF-STATE:      STRATEGICAL-STATE
PARENT-LEVEL:       !!STRATEGICAL-LEVEL/TS-3!!
DEFAULT-PROGRESSION-LINKS:
                    (!!PREDICTION-COLLECTION-PHASE/TS-3=>
                     TUTORING-PHASE/TS-3!!)
META-PROGRESSION-LINKS:
                    (!!TUTORING-PHASE/TS-3=>PREDICTION-
                     COLLECTION-PHASE/TS-3!!)
DEFAULT-IN-LEVEL-REFINEMENT-LINKS:
                    NIL
META-IN-LEVEL-REFINEMENT-LINKS:
                    NIL
DEFAULT-BETWEEN-LEVELS-REFINEMENT-LINKS:
                    (!!TUTORING-PHASE/TS-3=>TUTOR-PRIMARY-
                            VARIABLE/TS-3!!
                    !!TUTORING-PHASE/TS-3=>TUTOR-REMAINING
                            -PREDICTION-TABLE-VARIABLES/TS-3!!)
META-BETWEEN-LEVELS-REFINEMENT-LINKS:
                    NIL
HISTORY:            !!H/TUTORING-PHASE/TS-3!!
```

Figure 8.12 A Tutoring State Object

Each tutoring state object has a set of methods (or behaviors) attached to it. When a tutoring state object has been selected by the instructional planner, these methods determine its actions. An example link object is shown in Figure 8.13. Each link object also has its associated behavior. This behavior is responsible for the criteria that determine which link the planning engine traverses to select next tutoring state.

MIX-IN-NAME:	DEFAULT-LINK
TYPE-OF-LINK:	DEFAULT-PROGRESSION-LINK
ANTECEDENT-STATE:	!!PREDICTION-COLLECTION-PHASE/TS-3!!
CONSEQUENCE-STATE:	!!TUTORING-PHASE/TS-3!!
RULE-USAGE-HISTORY:	NOT-LATELY-FIRED

Figure 8.13 A Tutoring Link Object

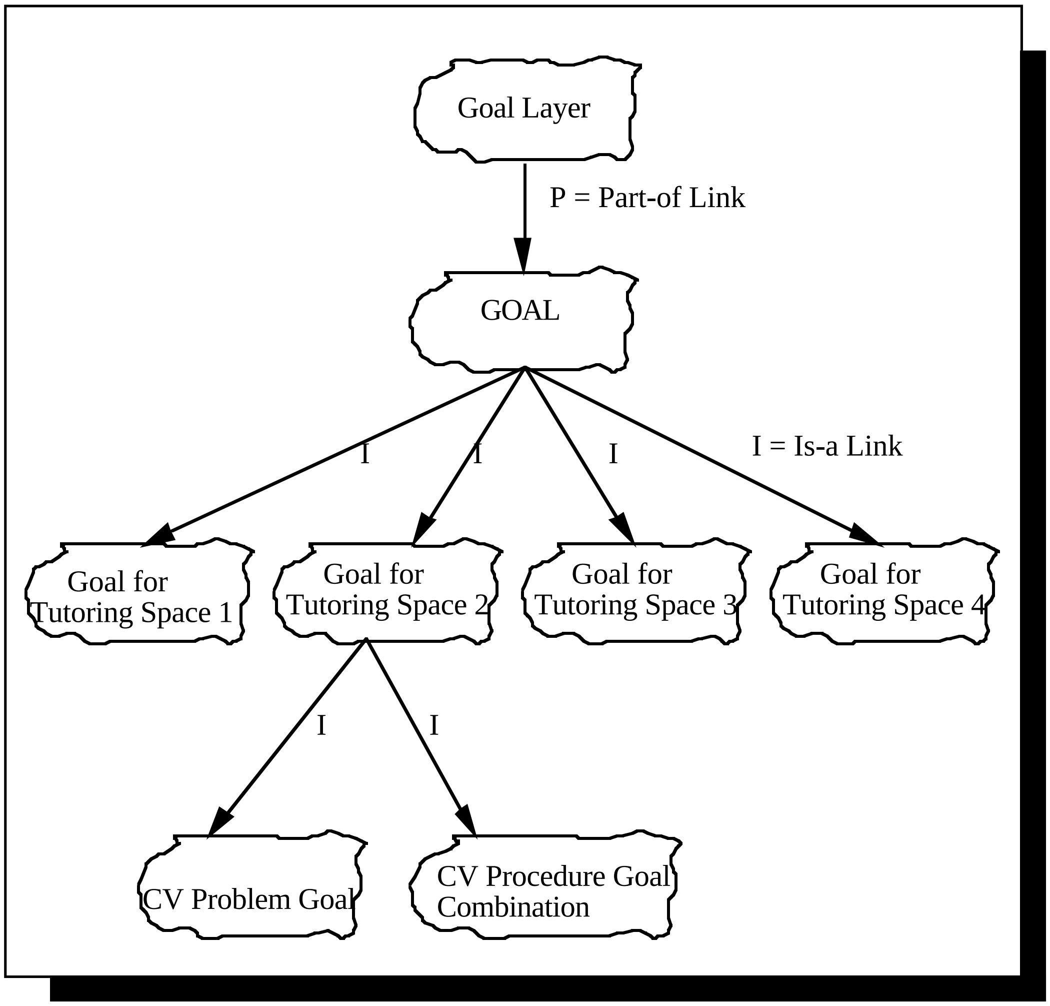

Goal Layer
P = Part-of Link
GOAL
I
I
I
I = Is-a Link
Goal for Tutoring Space 1
Goal for Tutoring Space 2
Goal for Tutoring Space 3
Goal for Tutoring Space 4
I
I
CV Problem Goal
CV Procedure Goal Combination

Figure 8.14 Hierarchy of Classes for the Curriculum

8.10 Curriculum: Containing Goals of CIRCSIM-Tutor (v.3)

The curriculum is an information store in the architecture of CIRCSIM-Tutor (v.3) (see Figure 8.1). This store contains the goals of CIRCSIM-Tutor (v.3). It is an represents the curriculum dimension of the system model (see Figure 8.2). Here goals are explicitly represented as objects. The hierarchy of classes used for this information store is shown in Figure 8.14. In this store goals are classified according to their location in various tutoring space. In other words, each tutoring space used in CIRCSIM-Tutor (v.3) has its own set of goals. It is because of this organization, it is possible to customize the representation and content of each goal object according to the needs of a tutoring space. Figure 8.15 shows the contents of a goal object used in the exercise space.

```
NAME:                           CV-PROCEDURE-COMBINATION-35
SYNONYM:                        NIL
DIRECTLY-TEACHABLE-P:           NO
TYPE-OF-GOAL:                   NON-CORE
LOCATION:                       !!TUTORING-SPACE-2!!
SUPER-GOALS:                    (!!GOAL-3/TS-2!! !!GOAL-4/TS-2!!
                                 !!GOAL-6/TS-2!!)
SUB-GOALS:                      (AND-STUDENT-HISTORY (!!GOAL-8/TS-2!!
                                 !!GOAL-16/TS-2!!))
ANALOGY:                        NIL
SUPPORTING-GOALS:               NIL
PERIPHERAL-GOALS:               NIL
GOAL-STATUS:                    NOT-COVERED
POINTERS-TO-DOMAIN-LAYER: NIL
RESULT-OF-TUTORING:             NIL
RE-TUTORABLE:                   NO
LAST-TUTORING-PLAN-USED:        NIL
LAST-SELECTION-PLAN-USED:       NIL
NO-OF-TIMES-TUTORED:            0
GOAL-CLASSIFICATION:            CV-PROCEDURE-COMBINATION
DEFAULT-DIFFICULTY-LEVEL:       CHALLENGING
FIRST-PROCEDURE-OF-THIS-COMBINATION:
                                !!GOAL-8/TS-2!!
SECOND-PROCEDURE-OF-THIS-COMBINATION:
                                !!GOAL-16/TS-2!!
```

Figure 8.15 A Goal Object

8.11 Tutoring History: Storing a Trace of Key Decisions of the Instructional Planner

The tutoring history is also an information store (see Figure 8.1). The major purpose of this store is to store the key elements of the instructional planner’s decisions so that these could be used while performing pedagogy decision making. Theoretically, this store can be used to store the complete history of pedagogical decision making performed by the instructional planner but currently it only stores the most recent key decisions of the instructional planner.

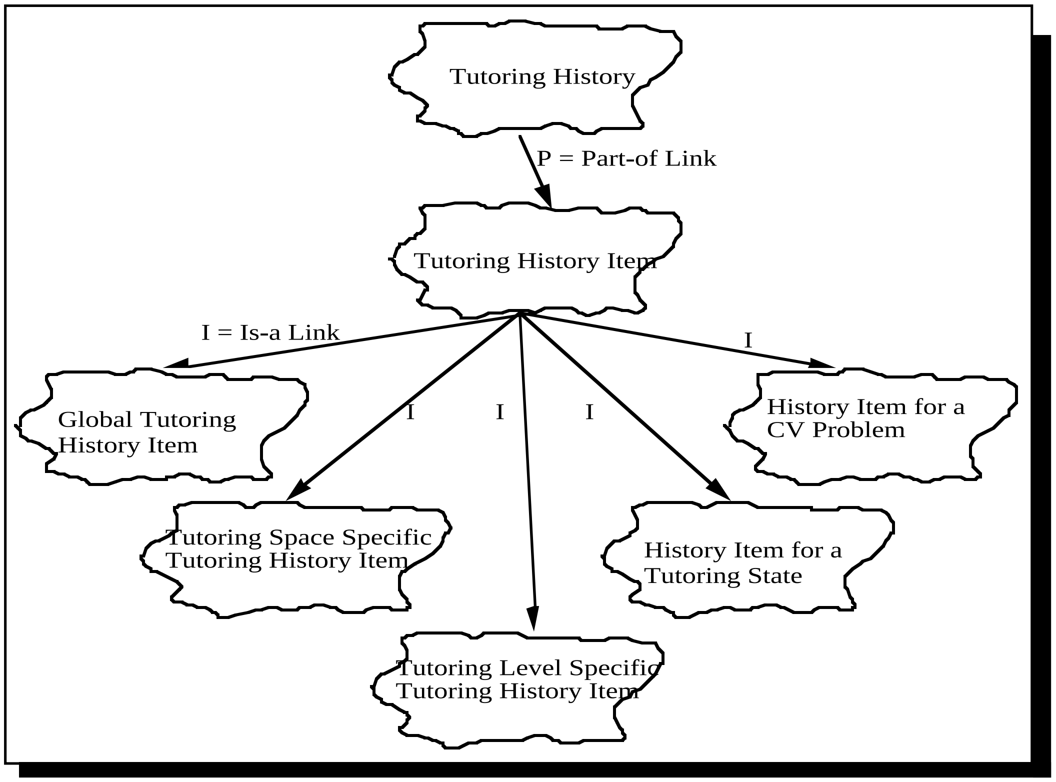

Tutoring History
P = Part-of Link
Tutoring History Item
I = Is-a Link
I
I
I
I
Global Tutoring History Item
History Item for a CV Problem
Tutoring Space Specific Tutoring History Item
History Item for a Tutoring State
Tutoring Level Specific Tutoring History Item

Figure 8.16 Hierarchy of Classes for the Tutoring History

A hierarchy of classes used in the tutoring history is shown in Figure 8.16. The objects of this store are classified according to the level at which the instructional planner makes decisions, e.g., this could be at the global level and the tutoring space level. The contents of a tutoring history object are shown in Figure 8.17. This object shows the activity status of the instructional planner at different planning levels.

```
NAME:          ACTIVITY-STATUS
CONTENT/SPACE-1/PEDAGOGICAL: BUSY
CONTENT/SPACE-1/STRATEGICAL:  BUSY
CONTENT/SPACE-1/TACTICAL:  BUSY
CONTENT/SPACE-2/PEDAGOGICAL: BUSY
CONTENT/SPACE-2/STRATEGICAL:  BUSY
CONTENT/SPACE-2/TACTICAL:  BUSY
CONTENT/SPACE-3/PEDAGOGICAL: BUSY
CONTENT/SPACE-3/STRATEGICAL:  BUSY
CONTENT/SPACE-3/TACTICAL:  BUSY
CONTENT/SPACE-4/PEDAGOGICAL: FREE
CONTENT/SPACE-4/STRATEGICAL:  FREE
CONTENT/SPACE-4/TACTICAL:  FREE
```

Figure 8.17 A Tutoring History Object

8.12 A Run of the Instructional Planner

Appendix E contains a partial trace of the decision making performed by the instructional planner while executing. This trace shows only the major decisions of the instructional planner.

CHAPTER IX

CONCLUSIONS, LIMITATIONS, AND FUTURE DIRECTIONS

9.1 <u>Introduction</u>

In this thesis a model of tutoring has been described. This model is intended for CIRCSIM-Tutor (v.3) - an Intelligent Tutoring System (ITS) - that tutors first year medical students on the functioning of the baroreceptor reflex, a negative feedback system. The fundamental assumption behind this research is that basing the development of an ITS on the study of the effective human tutors provides the best approach to developing effective machine tutors.

This model of tutoring is based on the behavior of our human tutors (AAR and JAM) in the keyboard-to-keyboard sessions. The effectiveness of the tutoring method of our human tutors has now been formally evaluated. This gives us confidence in the effectiveness of this model of tutoring.

My model defines the behaviors of the domain and the pedagogy experts (see Figure 4.2). The assumption here is that the domain expert provides the domain intelligence to the rest of CIRCSIM-Tutor (v.3), whereas the pedagogy expert is mainly responsible for two functions, the tutoring protocol used throughout the session and ongoing decisions during the session. The tutoring protocol provides a higher level plan for tutoring. During a session the pedagogy expert makes three major types of decisions: what to teach, when to teach, and how to teach. It is this last function that causes the pedagogy expert to interact heavily with the domain expert.

The domain expert basically uses a set of qualitative and causal models of the domain. These models in this thesis have been classified as: the parametric and the anatomical models of the CV system. To represent this domain knowledge a classification scheme is used. This scheme distinguishes six dimensions in which the domain knowledge can be represented. These modeling dimensions are: perspective, elaboration, scope, sufficiency, aggregation, and generalization.

This thesis also reports a study of the nature of integration between these two experts. This study showed that the integration between these experts can be explained in terms of a set of knowledge structures. One such structure is the inference triangle that has been used by these experts to perform the domain and the pedagogy reasoning.

For CIRCSIM-Tutor (v.3) a specific view of the student has been created that is used by the pedagogy expert to develop tutoring responses during a session. This view has three levels: the error level, the error pattern level, and the student difficulty level. The major objective of the pedagogy expert is to develop lessons that remediate the student's misconceptions.

The major goal of the model of tutoring described in this thesis is to help the student integrate his/her knowledge into a coherent qualitative causal model of the domain and solve problems in the domain. The key feature of this model is that the tutor uses multiple models of the domain in the process of facilitating knowledge integration. This model of tutoring is in the tradition of the Ohlsson's (1991) second-order theory of tutoring. I call this theory the integration theory of tutoring. In accordance with this theoretical orientation my model puts more emphasis on remedying the student's misconceptions.

The development of my model of tutoring has been approached using an ITS development framework. This framework has been developed by combining the key features of KADS, a popular knowledge based system development methodology with some design prescriptions from the field of instructional system design (ISD). This framework views the development of an ITS as a modeling activity. There are three major phases of this methodology. These are the conceptual phase, the system phase, and physical phase. At each phase a different model of an ITS results. This framework also agrees with the currently popular view of knowledge acquisition. According to this view the development of a knowledge base system is a collaborative activity that takes place between the knowledge engineer and the domain expert.

The description of the model of tutoring described above is more or less at the conceptual level. Once the conceptual model was ready I transformed it into the system model. This model is the outcome of the first subphase of the system phase of development. The major effort during the development of the system model was to make it as general as possible. This model is an improvement on Lesgold’s (1988) framework for machine tutors. The knowledge in an ITS, viewed through the system model, is organized around three dimensions: the planning dimension, the curriculum dimension, and the domain knowledge dimension. The key feature of this system model is that it attempts to combine the model-based and the curriculum-based themes of ITS. From another point of view, the system model consists of a set of tutoring spaces. Each space is responsible for performing one type of major decision of the tutor during interaction with the student. For CIRCSIM-Tutor (v.3) the following tutoring spaces are used: the major-objective space, the exercise space, the unit space, and the lesson space.

The outcome of the second subphase of the system phase is an architecture. This research has developed an architecture for CIRCSIM-Tutor (v.3). This architecture divides its components into two major classes: modules (or subsystems) and information stores. Modules are active processes that communicate and coordinate to create the required intelligent behavior for the system. As the name implies, an information store is a store of information/knowledge. I have used a version of Booch’s (1991) object-oriented methodology to develop an object model for four major architectural components of CIRCSIM-Tutor (v.3). These components are the instructional planner, the domain knowledge representation system (DKRS), the curriculum, and the tutoring history. The first one is a module and the remaining three are information stores. The DKRS is the system embodiment of the domain expert, whereas the remaining three components are the system embodiment of the pedagogy expert.

In the last phase of the ITS development methodology these architectural components are coded into software program. I have used the Common Lisp Object System for this purpose. The hardware platform on which this implementation was performed is the Apple Macintosh.

9.2 Significance and Limitations of This Research

There are many significant aspects of the research described in this thesis. The ITS development framework used for this research is a step towards a generic methodology to develop an ITS. There are two major advantages of this framework. Since it is based on a popular knowledge based system methodology, it provides a systematic methodology that can be used to develop large scale ITSs for real world application domains. Second, it is also based on the field of instructional system design, the development of these ITSs can be tailored so that it can be used in a real educational setting. The field of ITS needs such a development methodology (Khuwaja et al., 1994a). A successful use of this methodology for my research provides a feasibility study for a long term goal of developing a generic ITS methodology.

This research has developed a conceptual model of an ITS which is broken down into different levels (see Figure 4.2). The successive levels of this space define an increasing approximation to the behavior of human tutors.

The model of tutoring developed in this research uses the integration theory of tutoring to combine the Socratic method and the mental model theme into a single framework. Research on the WHY system (Collins, 1985) also had a similar agenda but it did not crystallize into a model of tutoring. Perhaps one reason for this is that the tutoring scenario Collins proposed puts heavy demands on the diagnostic phase of tutoring. We cannot meet these demands using current state-of-the-art AI research. Our model makes use of both an overlay and a bug library approach to the diagnostic problem. This method is quite commonly used in ITS research.

Using the Socratic method to teach problem-solving is itself a challenge because it raises an age old issue of immediate vs. delayed feedback. This model of tutoring uses a novel tutoring protocol, developed by our expert tutors. It merges the themes of immediate and delayed feedback.

Based on the elaboration hierarchy this model of tutoring uses multiple models of the domain to remedy the student's misconceptions. Only a few research efforts have been geared to use multiple models for tutoring. The method of use of multiple models of the domain in this model of tutoring is quite pragmatic and novel.

The integration between the domain and the pedagogy expert has been formally studied in this research. This study shows that these two roles of the tutor use common knowledge structures for their reasoning purposes. We believe that these knowledge types provide the "glue" that integrates different types of expertise in the skilled human tutor and makes the whole process of tutoring effective.

This model of tutoring is influenced by variables such as: the style and method of tutoring, the tutoring domain, the learning context, the teaching goals, and the nature of tutoring task (see Section 5.3). We hope that taking these variables into account will give us a more general model of tutoring.

The system model combines the curriculum and model-based themes of ITS. It advances Lesgold's knowledge representation framework for ITSs. The system model is generic and domain independent. It divides the tutoring expertise into three dimensions: the planning dimension, the curriculum dimension, and the domain knowledge dimension.

The system model forces the designer to concentrate on the "system" rather than "conceptual" issues of tutoring. This division of emphasis is important for different phases of ITS development. The separation of this model from the conceptual model is an important contribution in itself. Very few research efforts have made this explicit division. One advantage of this division is that the system model can use any conceptual model to develop an ITS. In other words the system model can provide a research tool for the ITS designer to test his/her theories of tutoring. In CIRCSIM-Tutor (v.3) the system model allowed us to use our conceptual model of tutoring to develop the system.

The architecture of CIRCSIM-Tutor (v.3) is very generic. It is domain and tutoring method independent. I am in favor of the use of the multiple expert metaphor for ITS development. But I think this metaphor should not be carried over to develop the architecture of an ITS. This architecture encourages the developer to break this metaphor into its components. This architecture uses software engineering principles to support modularity, portability, and extendibility. Historically the conceptual model is continuously changing in the CIRCSIM-Tutor project. One characteristic of this architecture is that it attempts to minimize, as much as possible, the effect of change of the conceptual model on the physical design of the system.

9.3 <u>Future Research Directions</u>

The research reported in this thesis can be continued along several directions. The purpose of this section is to identify these directions.

The ITS development framework described in this thesis needs further research to make it complete and general enough to be useful to develop all types of ITSs. At the conceptual phase this research assumes that the study of human tutors provides the best tutoring scenario. This assumption needs further investigation. It would be interesting to see how other methods of developing conceptual models (see Section 2.2.3) affect this ITS development methodology.

I have developed a knowledge based development methodology to develop conceptual models of the domain. It would be interesting to investigate the generality of this approach to develop qualitative models in other domains (such as respiratory physiology and electronic circuit design).

The domain expert uses multiple models of the domain. I believe that the models developed during this research are only a small set of the possible ones that our tutors use while performing reasoning in the domain. A definite research direction is to investigate all possible domain models and their usage in our tutoring context.

Only a subset of behaviors of our tutors in the keyboard-to-keyboard sessions have been considered for the pedagogy expert. It would be extremely advantageous to model other complex behaviors (e.g., behaviors responsible for the exploratory phase of the tutoring cycle - see Section 6.5) for CIRCSIM-Tutor (v.3). This indeed will put heavy burden on all components of CIRCSIM-Tutor (v.3) but it will make the behavior of the system much more like the behavior of our tutors in the keyboard-to-keyboard sessions. We have avoided exploring this area further because of the limitations of the input understander.

The anatomical model of the CV system is not yet fully developed, it is extremely important to make it fully functional so that the pedagogy expert could use both anatomical and physiological perspectives of the CV system for tutoring.

A study to investigate the nature of the integration between the domain and the pedagogy expert has already been initiated by this research. It would be advantageous to expand this study to explore all the types of glue that combine to allow the experts (e.g., the communication expert, the student modeler) to create an effective tutoring behavior.

Once the implementation of CIRCSIM-Tutor (v.3) is complete it would be extremely advantageous to compare its effectiveness with the effectiveness of our human tutors in the keyboard-to-keyboard sessions.

The system model is domain independent. It would be interesting to test its utility in developing ITSs in other domains.

My long-term goal is to generalize this model to develop a theory of tutoring that is general enough to be used in various tutoring situations and domains.

APPENDIX A

RESULTS OF AN EVALUATION STUDY FOR A KEYBOARD-TO-KEYBOARD TUTORING METHOD

TUTORING EXPERIMENT
APRIL 1993

Control Protocol

1. Pretest

a. CV Relationship exam #1 (10 minutes allotted)
b. Problem-half of group received Problem A, half received Problem B (30 minutes allotted)

2. Text

Segments of 3 chapters from Heller and Mohrman, Cardiovascular Physiology were provided. Students were given 1 hour to read the text.

3. Problem P

Students were given problem P (the same problem as the experimental group were tutored on) to solve (30 minutes allotted).

4. Posttest

a. CV Relationship exam #2 (same questions as #1 but rearranged - 10 minutes allotted).
b. Problem - students who had done Problem A as pretest were given problem B; and those that had done Problem B were given Problem A.

Experimental Protocol

1. Pretest - same as above.

2. Problem P
Students solved problem and were tutored by faculty. Students were required to get primary variable and its value correct. Otherwise tutoring took place only after each Prediction Table column was completed. No time limit.

3. Posttest - same as above.

Analysis

1. CV Relationship exams. Number of correct entries (#C) and number of wrong entries (#W) were tallied.

2. Problems. Number of incorrect predictions (#W) and number of relationship errors (#bugs) were tallied.

APPENDIX B

CV PROBLEMS: THEIR COMBINATIONS AND DESCRIPTIONS

BASIC PERTURBATION REPERTOIRE

BASIC PROCEDURES

PRA:	Reduce Arterial Resistance(RA) to 50% of normal.
PDB:	Denervate the Baroreceptors.
PBV:	Hemorrhage - Remove 1.0 L (Blood Volume = 4.0 L).
PIS:	Decrease Inotropic State (IS) to 50% of normal.
PRV:	Increase Venous Resistance(RV) to 200% of normal.
PIT:	Increase Intrathoracic Pressure(PIT) from -2 to 0 mm Hg.

DRUGS

DAB:	Administer a Beta-adrenergic agonist.
DAC:	Administer a Cholinergic agonist.
DAA:	Administer a Alpha-adrenergic agonist.
DBB:	Administer a Beta-adrenergic antagonist(blocker).
DBC:	Administer a Cholinergic(muscarinic) antagonist(blocker).
DBA:	Administer a Alpha-adrenergic antagonist(blocker).

ARTIFICIAL PACEMAKER

APU:	Install artificial pacemaker. Increase Heart Rate(HR) from 72 to 120.
APD:	Install artificial pacemaker. Decrease Heart Rate(HR) from 72 to 50.

POSSIBLE PERTURBATION COMBINATIONS

1) PRA After (DBB Or DBC Or AP(U Or D)).

2) PBV After (DBB Or DBC Or DBA Or AP(U Or D)).

3) PIS After (DBC Or DBA Or AP(U Or D)).

4) PRV After (DBB Or DBC Or DBA Or AP(U Or D)).

5) PIT After (DBB Or DBC Or DBA Or AP(U Or D)).

6) DAB After (DBC Or DBA Or AP(U Or D)).

7) DAC After (DBB Or DBA).

8) DAA After (DBB Or DBC Or AP(U Or D)).

9) AP(U Or D) After (DBB Or DBC Or DBA).

10) (DBB Or DBA) After AP(D).

11) (PRA Or PIS Or APD Or DBB) After PDB.

CIRCSIM-Tutor PROBLEM DESCRIPTIONS TYPES

1. DIRECT DEFINITION OF PRIMARY VARIABLE
2. INDIRECT DEFINITION OF PRIMARY VARIABLE
3. DIRECT DEFINITION OF PROCEDURE VARIABLE
4. INDIRECT DEFINITION OF PROCEDURE VARIABLE

REDUCE Ra TO 50% (PRA)

1. A patient was given a drug by his physician that reduced his total peripheral resistance by 50%.

2. A medical student injected an experimental animal with a drug that reduced the animal's arterial resistance to 50% of normal.

3. An unsupervised child was playing in the kitchen and drank some fluid that contained a chemical that significantly dilated the child's blood vessels.

4. A group of teenagers were experimenting with drugs. One of them swallowed some pills that contained a specific arteriolar smooth muscle relaxant.

DENERVATE THE BAROCEPTORS (PDB)

1. As part of an experiment in the physiology laboratory, a medical student cut the nerves from the baroreceptors. As a result, information about blood pressure can not reach CV centers in the central nervous system.

4. In the process of trying to remove some tumors growths in a patient's neck, a surgeon accidentally cut the patient's carotid sinus nerves.

HEMORRHAGE (PBV)

2. A medical student donated 1 liter of blood to a patient about to undergo surgery. Predict the effects of the student's blood donation.

4. AB played several vigorous games of tennis on a hot, humid summer day, without a break and without drinking anything.

DECREASE INOTROPIC STATE TO 50% (PIS)

1. Mr. HT has a condition that reduces the inotropic state of his heart.

2. Miss EM is given a drug that results in increased intracellular calcium in heart muscle cells.

3. Ms. BF has coronary artery disease. This condition reduces the blood flow to her myocardium, limiting the delivery of oxygen and nutrients to the heart muscle.

4. Mr. NS has a condition that reduces the synthesis of adrenergic receptors by his heart muscle cells.

INCREASED VENOUS RESISTANCE TO 200% (PRV)

1. Predict the effects of increasing venous resistance. Assume that no change in venous capacitance or venous compliance occurs.

2. A patient was admitted to the hospital after experiencing a fainting spell. After a series of tests her problem was determined to be an abdominal tumor that was compressing her vena cava, reducing her venous return.

3. Certain agents are known to cause veno-constriction, without affecting venous compliance or capacitance. What would be the effect of administering this agent to a patient?

4. An astronaut was placed in a human centrifuge. The centrifuge was rotated to provide a force of 3 gees (3 times the force of gravity) acting from his head toward his feet.

INCREASED INTRATHORACIC PRESSURE (PIT)

2. A medical student was testing her cardiovascular reflexes in the physiology lab. She performed a procedure that raised her intrathoracic pressure from -2 to 0 mm Hg.

4. A parent was preparing for her 5 year old's birthday by blowing up balloons. One very large balloon was particularly stiff. What would be the cardiovascular effect of her effort to inflate this balloon. Assume that she tried to blow it up in one very long, sustained expiratory effort.

BETA ADRENERGIC AGONIST (DAB)

1. Predict the effects of simultaneously increasing both heart rate and cardiac contractility (cardiac inotropic state) using the maintained infusion of a drug.

2. Predict the effects of continuously administering a long-acting, potent drug that produces the same effects as stimulating the sympathetic nerve supply to the heart.

3. What would be the effect of continuously infusing a subject with a potent, long-acting beta-adrenergic agonist?

<u>CHOLINERGIC AGONIST (DAC)</u>

1. An individual was continuously infused with a long-acting drug that reduces heart rate. Predict the consequences.

2. Predict the effects of continuously infusing an individual with a long-acting, potent drug that has the same effect as cholinergic stimulation of the SA node.

3. An individual was infused with a long-acting cholinergic agonist. Predict the effects.

<u>ALPHA-ADRENERGIC AGONIST (DAA)</u>

1. Predict the effects of a maintained infusion of an individual with a potent, long-acting drug that increased total peripheral resistance.

2. An individual was continuously infused with a potent, long-acting drug that has the same effects as stimulating the sympathetic nerve supply to the blood vessels.

3. Predict the effects of continually infusing an individual with a potent, long-acting alpha-adrenergic agonist.

<u>BETA-ADRENERGIC ANTAGONIST (DBB)</u>

1. Predict the effects of simultaneously reducing the inotropic state (cardiac contractility) of the heart and the heart rate.

2. An individual was continuously infused with a potent, long-acting drug that interferes with the tonic effects of the sympathetic nervous system on the SA node and the myocardium.

3. Predict the effects of continually infusing an individual with a potent, long-acting beta-adrenergic antagonist.

<u>CHOLINERGIC ANTAGONIST (DBC)</u>

1. Predict the effects of continually administering a potent, long-acting drug whose only effect is to increases the heart rate.

2. Predict the effects of a maintained infusion with a potent, long-acting drug that prevents the tonic para-sympathetic stimulation of the SA node.

3. What would be the effects of continually infusing an individual with a potent, long-acting cholinergic muscarinic antagonist (blocking agent)?

<u>ALPHA-ADRENERGIC ANTAGONIST (DBA)</u>

1. Predict the effects of continually administering an potent, long-acting drug that decreases the total peripheral resistance.

2. Predict the effect of continually infusing a subject with a potent, long-acting drug that prevents the effects of the tonic sympathetic action on blood vessels.

3. What would be the effects of a continuous infusion with a potent, long-acting alpha-adrenergic antagonist (blocking agent)?

PACEMAKER (APU / APD)

1. An individual with a non-functioning SA node has had an artificial pacemaker implanted that is the sole determiner of her heart rate. The pacemaker has been running at 72/minute for months. Suddenly, it misfunctioned and the rate changed to 120/minute (50/minute).

APPENDIX C

A LIST OF ERROR PATTERNS AND STUDENT DIFFICULTIES

LIST OF ERROR PATTERNS

Phase = DR

Wrong Primary Variable

Wrong Primary Variable Prediction

Any Neural Error DR

CVP -> SV

SV -> CO

CO -> MAP

TPR -> MAP

HR -> CO

IS -> SV

CO inv CVP

MAP inv SV

MAP = TPR x CO

CO = HR x SV

Phase = RR

Any Clamped Neural RR

Any Neural Error RR

Any Non Clamped 0 in Neural-RR

Any Non Clamped 0 in RR

RR-MAP Incorrect

MAP in DR inv TPR in RR

MAP in DR inv HR in RR

MAP in DR inv IS in RR

CVP -> SV

SV -> CO

CO -> MAP

TPR -> MAP

HR -> CO

IS -> SV

CO inv CVP

MAP inv SV

MAP = TPR x CO

CO = HR x SV

<u>Phase = SS</u>

MAP Incorrect SS

Any 0 in SS

HR Algebraic SS

CO Algebraic SS

SV Algebraic SS

IS Algebraic SS

TPR Algebraic SS

CVP Algebraic SS

CVP -> SV

SV -> CO

CO -> MAP

TPR -> MAP

HR -> CO

IS -> SV

CO inv CVP

MAP inv SV

MAP = TPR x CO

CO = HR x SV

LIST OF STUDENT DIFFICULTIES

Slip

Does Not Know

Mechanism

Definition of DR

Actual Neural Variables

IS Confusion

IS/Pre Load Confusion

Pre Load Confusion

In/Out Balance of Heart

After Load Confusion

Causality/Algebra

Approximate MAP

CO/SV Confusion

Effect of Clamping

Definition of RR

Regulated/Effector Variable Confusion

Compared to What

Sympathetic/Parasympathetic Confusion

Incorrect DR and RR Summation

Fully Compensated

APPENDIX D

HOW TO QUERY THE DKRS

I Introduction

The Domain Knowledge Representation System (DKRS) is an information store in the architecture of CIRCSIM-TUTOR (v.3). It contains two main parts: the Domain Knowledge Base (DKB), and the Domain Problem Solver (DPS). The DKB defines domain knowledge base objects and their behavior, whereas the DPS defines domain problem solver objects and their behavior. The DKRS is implemented in CLOS of Procyon Common Lisp. It is comprised of the following files:

1) class-declarations-for-DKB.lsp
2) class-declarations-for-DPS.lsp
3) create/init-DKB-instances.lsp
4) create/init-DPS-instances.lsp
5) connect-DKB-objects.lsp
6) connect-DPS-objects.lsp
7) declare/initialize-DKRS.lsp
8) accessory-operations.lsp
9) behavior-of-DKB-objects.lsp
10) behavior-of-DPS-objects.lsp

There are two types of knowledge available in the DKRS: factual and inferred. The factual knowledge is explicitly represented in the DKRS whereas the inferred knowledge can be obtained by manipulating the factual knowledge using inference procedures. Most of the knowledge in the DKRS is accessed by composing a query. A query has two main forms. In the first form, it can be used to obtain a specific information about a domain object. In the second form it can be used to confirm or reject any previously acquired information about a domain object. In this appendix Lisp expressions are shown in *italics.*

II Composing a Query to Access the Factual Domain Knowledge

There are two types of factual knowledge in the DKRS: static and dynamic. The static knowledge is the unchangeable information about a domain object whereas the dynamic knowledge about an object changes with the state of the system. Both of these types are obtained by accessing the slot contents of an object.

A Accessing the Static Domain Knowledge

The general template for a query to access the static knowledge of a domain object is as follows:

Template: (query '(<slot-name> <object-name> <?/value-of-slot>))

Where:
<slot-name> = The name of a slot of an object whose contents need to be analyzed, e.g. definition.

<object-name> = The global name of an object whose slot value need to be analyzed, e.g. !!HEART-RATE!!, !!BLOOD!!, !!CAUSAL-RELATION-MAP/SV!!, !!EQUATION-CO!!, !!ALPHA-ADRENERGIC-ANTAGONISTS!!

<?/value-of-slot> = This part of a query can be either "?" or a value of the specified slot. If it is "?" then the DKRS will fetch the content of the slot for the specified object. But if it is a value of the specified slot then the DKRS will assume that the caller wants to confirm this value by comparing it with the current value of the specified slot.

Result of a query: If the last section of a query contains a "?" then, provided the given slot exists for the specified object, the DKRS will return the current value of that slot. But if the last section of a query contains a value then, provided the given slot exists for the specified object, the DKRS will try to compare this value with the current value of the specified slot. If these two slot values are the same then a "True" is flagged otherwise a "nil" is returned.

The set of possible values for <slot-name> is: definition, synonyms, name, causal-relation, abbreviation, nature-of-regulation, tonic-activity, equation, unit, antecedent, consequence, medium, and nature-of-causal-relation.

Examples:

> *(query '(synonyms !!SMOOTH-MUSCLE!! ?))*
(CV-EFFECTOR)

> *(query '(synonyms !!SMOOTH-MUSCLE!! CV-EFFECTOR))*
T

> *(query '(synonyms !!SMOOTH-MUSCLE!! xyz))*
NIL

B Accessing the Dynamic Domain Knowledge

The general templates for queries to access the dynamic knowledge of a domain object are as follows:

1) A number of dynamic slots (<slot-name>) sharing the same template structure as is for static slots (given above) are: perturbation-in-action, current-stage-of-cv-system, mode-of-perturbation, level-of-concept-map, last-perturbation. These slots are only applicable for the "!!ENVIRONMENT!!" object.

Examples:

> *(query '(perturbation-in-action !!environment!! ?))*
BASIC-PROCEDURE1

> *(query '(perturbation-in-action !!environment!! BASIC-PROCEDURE1))*
T

> *(query '(current-stage-of-cv-system !!environment!! ?))*
STEADY-STATE

> *(query '(mode-of-perturbation !!environment!! ?))*
MULTIPLE

> *(query '(level-of-concept-map !!environment!! ?))*

DEEP

```
> (query '(last-perturbation !!environment!! ?))
!!ARTIFICIAL-PACEMAKER!!
```

2) **Template**: (query '(<head of query> <stage of cv system>
<level of concept map> ?))

Where:
<head of query> can be either "solution-path" or "first-variable-affected"
<stage of cv system> can be either "DR", "RR" or "SS" for Direct Response, Reflex Response and Steady State respectively.
<level of concept map> can be either “top”, “intermediate” or “deep.”

Examples:

```
> (query '(solution-path dr top ?))
((!!TOTAL-PERIPHERAL-RESISTANCE!! DECREASE)
  (!!MEAN-ARTERIAL-PRESSURE!! DECREASE)
  (!!STROKE-VOLUME!! INCREASE)
  (!!CARDIAC-OUTPUT!! INCREASE)
  (!!RIGHT-ATRIAL-PRESSURE!! DECREASE))

> (query '(first-variable-affected dr top ?))
((!!TOTAL-PERIPHERAL-RESISTANCE!! DECREASE))

> (query '(first-variable-affected DR deep ?))
((!!SMOOTH-MUSCLE-TONE!! DECREASE))
```

3) **Template**: (query '(<head of query> <object name> <stage of cv system>
<?/a-possible-answer>))

Where:
<head of query> is "value"
<object name> and <stage of cv system> are as defined above.
<?/a-possible-answer> = This part of query can be either "?" or a possible answer of this query. If it is "?" then the DKRS will fetch the content of the slot represented by the <head of query> for the specified object. But if it is a possible answer of the specified query then the DKRS will assume that the caller wants to confirm the specified answer.

Examples:

```
> (query '(value !!heart-rate!! dr ?))
NO-CHANGE

> (query '(value !!heart-rate!! rr ?))
INCREASE

> (query '(value !!heart-rate!! ss increase))
T
```

III Composing a Query to Infer the Domain Knowledge

Two types of knowledge structures are used in the DKRS. The first type enables us to infer about the anatomical knowledge about the CV system, whereas the second type holds the parametric qualitative and causal information about the CV system.

A Queries that Use Anatomical Knowledge of CV System

Template: (query '(<a-knowledge-source> <object-name> <?/a-possible-answer>
<an-optional-qualifier>))

Where:
<a-knowledge-source> = An information processing method. A possible set of knowledge sources in this case is: is-a, part-of, has-part, associated-with, and has-association.
<object-name> = The global name of an object, e.g. !!HEART-RATE!!, !!HEART!!
<?/a-possible-answer> = This part of query can be either "?", to find information, or a possible answer that need to be verified.
<an-optional-qualifier> = This is an optional part of this query and can be either "immediate" or "all". If it is "immediate" then inference procedure will only look for information one step above or below the current position in either is-a, part-whole or association hierarchy. But if it is "all" then the inference procedure searches the whole structure starting from the current position in the hierarchy.

Examples:

```
> (query '(is-a !!heart!! ?))
(ANATOMY-OBJECT)

> (query '(is-a !!heart!! ? immediate))
(ANATOMY-OBJECT)

> (query '(is-a !!heart-muscle!! ? all))
(HEART-MUSCLE MUSCLE ANATOMY-OBJECT DOMAIN-CONCEPT)

> (query '(is-a !!heart!! (ANATOMY-OBJECT DOMAIN-CONCEPT) all))
"Query has the wrong format!"

> (query '(is-a !!heart!! (ANATOMY-OBJECT DOMAIN-CONCEPT) ))
T

> (query '(part-of !!heart!! ?))
(!!CARDIOVASCULAR-SYSTEM!!)

> (query '(part-of !!heart!! ? immediate))
(!!CARDIOVASCULAR-SYSTEM!!)

> (query '(part-of !!systemic-arterioles!! ? all))
(!!ARTERIOLES!! !!ARTERIAL-SYSTEM!!
 !!SYSTEMIC-CIRCULATION!! !!CARDIOVASCULAR-SYSTEM!!)

> (query '(part-of !!heart!! !!CARDIOVASCULAR-SYSTEM!! ))
T
```

```
> (query '(has-part !!heart!! ?))
(!!PASSIVE-UNIT!! !!RIGHT-ATRIUM!! !!SA-NODE!!
 !!LEFT-VENTRICLE!!)

> (query '(has-part !!heart!! ? immediate))
(!!PASSIVE-UNIT!! !!RIGHT-ATRIUM!! !!SA-NODE!!
 !!LEFT-VENTRICLE!!)

> (query '(has-part !!heart!! ? all))
(!!PASSIVE-UNIT!! !!RIGHT-ATRIUM!! !!SA-NODE!!
 !!BETA-RECEPTOR!! !!C-M-RECEPTOR!! !!LEFT-VENTRICLE!!)

> (query '(has-part !!heart!! !!PASSIVE-UNIT!!))
T

> (query '(has-part !!heart!! asdf))
NIL

> (query '(associated-with !!heart!! ?))
"No such information is available!"

> (query '(associated-with !!heart-rate!! ?))
(!!SA-NODE!!)

> (query '(associated-with !!heart-rate!! ? immediate))
(!!SA-NODE!!)

> (query '(associated-with !!heart-rate!! ? all))
(!!SA-NODE!! !!HEART!! !!CARDIOVASCULAR-SYSTEM!!)

>  (query '(associated-with !!heart-rate!! !!HEART!! all))
"Query has the wrong format!"

> (query '(associated-with !!heart-rate!! !!HEART!! ))
T

>  (query '(has-association !!heart!! ? ))
NIL

> (query '(has-association !!heart!! ? all))
(!!STROKE-VOLUME!! !!FILLING-TIME!! !!HEART-RATE!!
 !!SA-NODE-RATE!! !!CARDIAC-OUTPUT!! !!RIGHT-ATRIAL-PRESSURE!!)

> (query '(has-association !!heart!! !!STROKE-VOLUME!!))
T

> (query '(has-association !!heart!!
         (!!STROKE-VOLUME!! !!FILLING-TIME!! !!HEART-RATE!!
        !SA-NODE-RATE!! !!CARDIAC-OUTPUT!! !!RIGHT-ATRIAL-PRESSURE!!)))
T

>  (query '(has-association !!heart!! srthrt))
NIL
```

B Queries that Perform Parametric Qualitative and Causal Reasoning in the Domain

1) **Template**: (query '(expand-causal-link <parameter-A> <parameter-B>
<level-of-concept-map> ?))

This query uses the knowledge source "expand-causal-link" to find all (causal) paths from parameter A to B in the specified level of the concept map.

Examples:

```
>  (query '(expand-causal-link !!heart-rate!!
     !!mean-arterial-pressure!! deep ?))
((!!HEART-RATE!! !!CARDIAC-OUTPUT!!
  !!ARTERIAL-BLOOD-VOLUME!! !!MEAN-ARTERIAL-PRESSURE!!)
 (!!HEART-RATE!! !!FILLING-TIME!!
  !!END-DIASTOLIC-VOLUME!! !!MUSCLE-FIBER-LENGTH!!
  !!ACTIN-MYOSIN-REACTION!!
  !!MYOCARD-CONTRACTILE-FORCE!! !!STROKE-VOLUME!!
  !!CARDIAC-OUTPUT!! !!ARTERIAL-BLOOD-VOLUME!!
  !!MEAN-ARTERIAL-PRESSURE!!)
 (!!HEART-RATE!! !!INTRACELLULAR-CA++!!
  !!ACTIN-MYOSIN-ASSOCIATION!!
  !!CARDIAC-CONTRACTILITY!!
  !!MYOCARD-CONTRACTILE-FORCE!! !!STROKE-VOLUME!!
  !!CARDIAC-OUTPUT!! !!ARTERIAL-BLOOD-VOLUME!!
  !!MEAN-ARTERIAL-PRESSURE!!))

> (query '(expand-causal-link !!heart-rate!!
     !!mean-arterial-pressure!! intermediate ?))
((!!HEART-RATE!! !!CARDIAC-OUTPUT!!
  !!ARTERIAL-BLOOD-VOLUME!! !!MEAN-ARTERIAL-PRESSURE!!))

> (query '(expand-causal-link !!heart-rate!!
      !!mean-arterial-pressure!! intermediasfgasgte ?))
"This link is not expandable in the given level!"

> (query '(expand-causal-link !!heart-rate!!
     !!mean-arterial-pressure!! top ?))
((!!HEART-RATE!! !!CARDIAC-OUTPUT!! !!MEAN-ARTERIAL-PRESSURE!!))
```

2) **Template**: (query '(<a-knowledge-source> <object-name>
<level-of-concept-map> <?/a-possible-answer>
<stage-of-cv-system>))

Where:
<a-knowledge-source> = "determinant" or "determines"
<object-name> and <level-of-concept-map> are as explained above. The last parameter <stage-of-cv-system> is an optional parameter. If it is present in the query then DKRS will interpret knowledge sources "determinant" and "determines" as "actual determinant" and "actually determines" respectively. If this query contains the knowledge source "determinant" then it will find the causal determinant of the given object. On the other hand, if this query contains the

knowledge source "determines" then it will find the parameters that are determined by the object given in the query.

Examples:

```
> (query '(determinant !!stroke-volume!! top ?))
(!!MEAN-ARTERIAL-PRESSURE!! !!RIGHT-ATRIAL-PRESSURE!!
  !!CARDIAC-CONTRACTILITY!!)

> (query '(determinant !!stroke-volume!! top ? rr))
(!!MEAN-ARTERIAL-PRESSURE!! !!RIGHT-ATRIAL-PRESSURE!!)

> (query '(determinant !!stroke-volume!! deep ?))
(!!MEAN-ARTERIAL-PRESSURE!! !!MYOCARD-CONTRACTILE-FORCE!!)

>  (query '(determinant !!blood-volume!! top ? ))
"The given parameter does not exists in the specified level of the concept map!"

> (query '(determinant !!sa-node-rate!! deep ? rr))
(!!EPINEPHRINE!! !!NOREPINEPHRINE!! !!ACETYLCHOLINE!!)

>  (query '(determines !!stroke-volume!! top ?))
(!!CARDIAC-OUTPUT!!)

>  (query '(determines !!stroke-volume!! top ? RR))
NIL

>  (query '(determines !!stroke-volume!! deep ?))
(!!CARDIAC-OUTPUT!!)

>  (query '(determines !!blood-volume!! top ? ))
"The given parameter does not exists in the specified level of the concept map!"

>  (query '(determines !!sa-node-rate!! deep ? rr))
(!!HEART-RATE!!)
```

3) **Template**: (query '(proportionality <parameter-A> <parameter-B>
<?/a-possible-answer> <stage-of-cv-system>))

This query using the knowledge source "proportionality" finds the nature of causal relationship between parameters A and B - it can be either direct, inverse or nil.

Examples:

```
> (query '(proportionality !!heart-rate!!
    !!mean-arterial-pressure!! ? dr))
NIL

>  (query '(proportionality !!heart-rate!!
    !!mean-arterial-pressure!! ? rr))
DIRECT

>  (query '(proportionality !!cardiac-output!!
    !!central-blood-volume!! inverse rr))
```

```
T

>  (query '(proportionality !!heart-rate!!
     !!mean-arterial-pressure!! invdfhdfherse rr))
NIL
```

4) **Template**: (query '(causes (<parameter-A> <?/a-possible-answer>)
(<parameter-B> <?/a-possible-answer>)
<stage-of-cv-system>))

This query using the knowledge source "causes" finds the value of parameter B/A given the value of parameter A/B. In other words this query finds the direction of change in parameter A/B as a result of the change in parameter B/A.

Examples:

```
> (query '(causes (!!heart-rate!! increase)
              (!!mean-arterial-pressure!! ?)  DR))
NIL

> (query '(causes (!!heart-rate!! increase)
              (!!mean-arterial-pressure!! ?)  rr))
INCREASE

>  (query '(causes (!!heart-rate!! decrease)
              (!!mean-arterial-pressure!! ?)  rr))
DECREASE

> (query '(causes (!!heart-rate!! decreadghfghse)
              (!!mean-arterial-pressure!! ?)  rr))
NIL

> (query '(causes (!!heart-rate!! ?)
              (!!mean-arterial-pressure!! decrease)  rr))
DECREASE

> (query '(causes (!!heart-rate!! ?)
              (!!mean-arterial-pressure!! increase)  rr))
INCREASE

> (query '(causes (!!heart-rate!! decrease)
              (!!mean-arterial-pressure!! increase)  rr))
NIL

>  (query '(causes (!!heart-rate!! increase)
              (!!mean-arterial-pressure!! increase)  rr))
T

>  (query '(causes (!!heart-rate!! incrghdgease)
              (!!mean-arterial-pressure!! increase)  rr))
NIL
```

IV <u>Generic Functions</u>

Besides the querying capability, DKRS also provides a set of generic functions that are available to all modules of the CIRCSIM-TUTOR (v.3). These functions perform various tasks in the domain to provide access to most frequently usable information.

A (global-object-name <object-name>)
This function fetches the internal name of the given object.

<u>Example</u>:

```
> (global-object-name 'heart)
(!!HEART!!)
```

B Following functions fetch a list of internal object names of a specific type.

<u>Examples</u>:

```
> (list-of-equations !!DKRS!!)
(!!COMPLIANCE!! !!HEMODYNAMICS!! !!EQUATION-CO!!)

> (list-of-causal-relations !!DKRS!!)
(!!CAUSAL-RELATION-BRP/CNSR!! !!CAUSAL-RELATION-CNSR/CC!!
  !!CAUSAL-RELATION-AMA/CC!! !!CAUSAL-RELATION-ICA/AMA!!
  !!CAUSAL-RELATION-EPI/ICA!!
  !!CAUSAL-RELATION-EPI/SANR!!
  !!CAUSAL-RELATION-NEPI/ICA!! !!CAUSAL-RELATION-VR/EDV!!
  !!CAUSAL-RELATION-CNSR/HR!! !!CAUSAL-RELATION-HR/ICA!!
  !!CAUSAL-RELATION-HR/FT!! !!CAUSAL-RELATION-HR/CO!!
  !!CAUSAL-RELATION-SFR/NEPI!!
  !!CAUSAL-RELATION-SFR/EPI!!
  !!CAUSAL-RELATION-CNSR/SFR!!
  !!CAUSAL-RELATION-PSFR/ACH!!
  !!CAUSAL-RELATION-CNSR/PSFR!!
  !!CAUSAL-RELATION-BRFR/CNSR!!
  !!CAUSAL-RELATION-CNSR/SANR!!
  !!CAUSAL-RELATION-NEPI/SANR!!
  !!CAUSAL-RELATION-ACH/SANR!!
  !!CAUSAL-RELATION-SANR/HR!! !!CAUSAL-RELATION-CC/MCF!!
  !!CAUSAL-RELATION-AMR/MCF!!
  !!CAUSAL-RELATION-NEPI/SMT!!
  !!CAUSAL-RELATION-EPI/SMT!! !!CAUSAL-RELATION-CNSR/AS!!
  !!CAUSAL-RELATION-SMT/AS!! !!CAUSAL-RELATION-BRP/BRS!!
  !!CAUSAL-RELATION-BRS/BRFR!!
  !!CAUSAL-RELATION-MFL/AMR!! !!CAUSAL-RELATION-CC/SV!!
  !!CAUSAL-RELATION-SV/CO!! !!CAUSAL-RELATION-MCF/SV!!
  !!CAUSAL-RELATION-EDV/MCF!! !!CAUSAL-RELATION-FT/EDV!!
  !!CAUSAL-RELATION-EDV/MFL!! !!CAUSAL-RELATION-VR/CBV!!
  !!CAUSAL-RELATION-CO/CBV!! !!CAUSAL-RELATION-CBV/EDV!!
  !!CAUSAL-RELATION-BV/CBV!! !!CAUSAL-RELATION-CO/ABV!!
  !!CAUSAL-RELATION-AS/RA!! !!CAUSAL-RELATION-GVS/RV!!
  !!CAUSAL-RELATION-SVS/RV!! !!CAUSAL-RELATION-RV/VR!!
  !!CAUSAL-RELATION-CNSR/TPR!!
  !!CAUSAL-RELATION-TPR/ABV!! !!CAUSAL-RELATION-RA/TPR!!
  !!CAUSAL-RELATION-VR/CVP!! !!CAUSAL-RELATION-BV/CVP!!
  !!CAUSAL-RELATION-CO/CVP!! !!CAUSAL-RELATION-CBV/CVP!!
  !!CAUSAL-RELATION-PVTM/GVS!!
  !!CAUSAL-RELATION-PIT/GVS!!
  !!CAUSAL-RELATION-PIT/PVTM!!
  !!CAUSAL-RELATION-PIT/CVP!! !!CAUSAL-RELATION-RAP/SV!!
  !!CAUSAL-RELATION-CO/RAP!! !!CAUSAL-RELATION-RAP/EDV!!
  !!CAUSAL-RELATION-CVP/RAP!! !!CAUSAL-RELATION-RAP/EDP!!
  !!CAUSAL-RELATION-EDP/EDV!! !!CAUSAL-RELATION-TPR/MAP!!
  !!CAUSAL-RELATION-CO/MAP!! !!CAUSAL-RELATION-MAP/SV!!
  !!CAUSAL-RELATION-ABV/MAP!! !!CAUSAL-RELATION-MAP/BRP!!
!!CAUSAL-RELATION-CHOLINERGIC-ANTAGONISTS/SA-NODE-RATE!!
!!CAUSAL-RELATION-BETA-ADRENERGIC-ANTAGONISTS/INTRACELLULAR-
CA++!!
!!CAUSAL-RELATION-BETA-ADRENERGIC-ANTAGONISTS/SA-NODE-RATE!!
!!CAUSAL-RELATION-ALPHA-ADRENERGIC-ANTAGONISTS/SMOOTH-
MUSCLE-TONE!!
!!CAUSAL-RELATION-CHOLINERGIC-AGONISTS/SA-NODE-RATE!!
!!CAUSAL-RELATION-BETA-ADRENERGIC-AGONISTS/INTRACELLULAR-
CA++!!
!!CAUSAL-RELATION-BETA-ADRENERGIC-AGONISTS/SA-NODE-RATE!!
!!CAUSAL-RELATION-ALPHA-ADRENERGIC-AGONISTS/SMOOTH-MUSCLE-
TONE!!
!!CAUSAL-RELATION-BASIC-PROCEDURE6/MEAN-INTRATHORACIC-
PRESSURE!!
!!CAUSAL-RELATION-BASIC-PROCEDURE5/SMALL-VEIN-SIZE!!
!!CAUSAL-RELATION-BASIC-PROCEDURE4/INTRACELLULAR-CA++!!
!!CAUSAL-RELATION-BASIC-PROCEDURE3/BLOOD-VOLUME!!
!!CAUSAL-RELATION-BASIC-PROCEDURE2/CENTRAL-NERVOUS-SYSTEM-
RESPONSE!!
!!CAUSAL-RELATION-BASIC-PROCEDURE1/SMOOTH-MUSCLE-TONE!!
!!CAUSAL-RELATION-ARTIFICIAL-PACEMAKER/HEART-RATE!!)

>  (list-of-anatomy-objects !!DKRS!!)
(!!C-M-RECEPTOR!! !!BETA-RECEPTOR!! !!ALPHA-RECEPTOR!!
  !!SMOOTH-MUSCLE!! !!EXPIRATORY-MUSCLE!!
  !!HEART-MUSCLE!! !!BLOOD!! !!PARASYMPATHETIC-NERVE!!
  !!SYMPATHETIC-NERVE!!
  !!PARASYMPATHETIC-NERVOUS-SYSTEM!!
  !!SYMPATHETIC-NERVOUS-SYSTEM!!
  !!AUTONOMIC-NERVOUS-SYSTEM!! !!BARORECEPTOR!!
  !!CENTRAL-NERVOUS-SYSTEM!! !!CENTRAL-VEIN!!
  !!SMALL-VEIN!! !!VENOUS-SYSTEM!! !!ORGAN-ARTERIOLES!!
  !!SYSTEMIC-ARTERIOLES!! !!ARTERIOLES!!
  !!ARTERIAL-SYSTEM!! !!SYSTEMIC-CIRCULATION!!
  !!LEFT-VENTRICLE!! !!SA-NODE!! !!RIGHT-ATRIUM!!
  !!PASSIVE-UNIT!! !!HEART!! !!BARORECEPTOR-REFLEX!!
  !!CARDIOVASCULAR-SYSTEM!!)

>  (list-of-parameters !!DKRS!!)
(!!ACETYLCHOLINE!! !!NOREPINEPHRINE!! !!EPINEPHRINE!!
  !!CENTRAL-NERVOUS-SYSTEM-RESPONSE!!
  !!CARDIAC-CONTRACTILITY!! !!ACTIN-MYOSIN-REACTION!!
  !!ACTIN-MYOSIN-ASSOCIATION!! !!INTRACELLULAR-CA++!!
  !!VENOUS-RETURN!! !!CARDIAC-OUTPUT!! !!FILLING-TIME!!
  !!HEART-RATE!! !!PARASYMPATHETIC-FIRING-RATE!!
  !!SYMPATHETIC-FIRING-RATE!!
  !!BARORECEPTOR-FIRING-RATE!! !!SA-NODE-RATE!!
  !!MYOCARD-CONTRACTILE-FORCE!! !!SMOOTH-MUSCLE-TONE!!
  !!VENTRICULAR-COMPLIANCE!! !!ARTERIAL-COMPLIANCE!!
  !!BARORECEPTOR-COMPLIANCE!! !!VENOUS-COMPLIANCE!!
  !!ARTERIOLE-SIZE!! !!BARORECEPTOR-SIZE!!
  !!MUSCLE-FIBER-LENGTH!! !!SMALL-VEIN-SIZE!!
  !!GREAT-VEIN-SIZE!! !!STROKE-VOLUME!!
  !!END-DIASTOLIC-VOLUME!! !!CENTRAL-BLOOD-VOLUME!!
  !!BLOOD-VOLUME!! !!ARTERIAL-BLOOD-VOLUME!!
  !!ARTERIAL-RESISTANCE!! !!VENOUS-RESISTANCE!!
  !!TOTAL-PERIPHERAL-RESISTANCE!!
  !!BARORECEPTOR-PRESSURE!! !!CENTRAL-VENOUS-PRESSURE!!
  !!VENOUS-TRANSMURAL-PRESSURE!!
  !!MEAN-INTRATHORACIC-PRESSURE!!
  !!RIGHT-ATRIAL-PRESSURE!! !!END-DIASTOLIC-PRESSURE!!
  !!MEAN-ARTERIAL-PRESSURE!!)

> (list-of-perturbations !!DKRS!!)
(!!CHOLINERGIC-ANTAGONISTS!! !!CHOLINERGIC-AGONISTS!!
  !!BETA-ADRENERGIC-ANTAGONISTS!!
  !!BETA-ADRENERGIC-AGONISTS!!
  !!ALPHA-ADRENERGIC-ANTAGONISTS!!
  !!ALPHA-ADRENERGIC-AGONISTS!! !!BASIC-PROCEDURE6!!
  !!BASIC-PROCEDURE5!! !!BASIC-PROCEDURE4!!
  !!BASIC-PROCEDURE3!! !!BASIC-PROCEDURE2!!
  !!BASIC-PROCEDURE1!! !!ARTIFICIAL-PACEMAKER!!)

> (complete-list-of-objects)
(!!C-M-RECEPTOR!! !!BETA-RECEPTOR!! !!ALPHA-RECEPTOR!!
  !!SMOOTH-MUSCLE!! !!EXPIRATORY-MUSCLE!!
  !!HEART-MUSCLE!! !!BLOOD!! !!PARASYMPATHETIC-NERVE!!
  !!SYMPATHETIC-NERVE!!
  !!PARASYMPATHETIC-NERVOUS-SYSTEM!!
  !!SYMPATHETIC-NERVOUS-SYSTEM!!
  !!AUTONOMIC-NERVOUS-SYSTEM!! !!BARORECEPTOR!!
  !!CENTRAL-NERVOUS-SYSTEM!! !!CENTRAL-VEIN!!
  !!SMALL-VEIN!! !!VENOUS-SYSTEM!! !!ORGAN-ARTERIOLES!!
  !!SYSTEMIC-ARTERIOLES!! !!ARTERIOLES!!
  !!ARTERIAL-SYSTEM!! !!SYSTEMIC-CIRCULATION!!
  !!LEFT-VENTRICLE!! !!SA-NODE!! !!RIGHT-ATRIUM!!
  !!PASSIVE-UNIT!! !!HEART!! !!BARORECEPTOR-REFLEX!!
  !!CARDIOVASCULAR-SYSTEM!! !!COMPLIANCE!!
  !!HEMODYNAMICS!! !!EQUATION-CO!!
  !!CHOLINERGIC-ANTAGONISTS!! !!CHOLINERGIC-AGONISTS!!
  !!BETA-ADRENERGIC-ANTAGONISTS!!
  !!BETA-ADRENERGIC-AGONISTS!!
  !!ALPHA-ADRENERGIC-ANTAGONISTS!!
  !!ALPHA-ADRENERGIC-AGONISTS!! !!BASIC-PROCEDURE6!!
  !!BASIC-PROCEDURE5!! !!BASIC-PROCEDURE4!!
```

```
!!BASIC-PROCEDURE3!! !!BASIC-PROCEDURE2!!
!!BASIC-PROCEDURE1!! !!ARTIFICIAL-PACEMAKER!!
!!ACETYLCHOLINE!! !!NOREPINEPHRINE!! !!EPINEPHRINE!!
!!CENTRAL-NERVOUS-SYSTEM-RESPONSE!!
!!CARDIAC-CONTRACTILITY!! !!ACTIN-MYOSIN-REACTION!!
!!ACTIN-MYOSIN-ASSOCIATION!! !!INTRACELLULAR-CA++!!
!!VENOUS-RETURN!! !!CARDIAC-OUTPUT!! !!FILLING-TIME!!
!!HEART-RATE!! !!PARASYMPATHETIC-FIRING-RATE!!
!!SYMPATHETIC-FIRING-RATE!!
!!BARORECEPTOR-FIRING-RATE!! !!SA-NODE-RATE!!
!!MYOCARD-CONTRACTILE-FORCE!! !!SMOOTH-MUSCLE-TONE!!
!!VENTRICULAR-COMPLIANCE!! !!ARTERIAL-COMPLIANCE!!
!!BARORECEPTOR-COMPLIANCE!! !!VENOUS-COMPLIANCE!!
!!ARTERIOLE-SIZE!! !!BARORECEPTOR-SIZE!!
!!MUSCLE-FIBER-LENGTH!! !!SMALL-VEIN-SIZE!!
!!GREAT-VEIN-SIZE!! !!STROKE-VOLUME!!
!!END-DIASTOLIC-VOLUME!! !!CENTRAL-BLOOD-VOLUME!!
!!BLOOD-VOLUME!! !!ARTERIAL-BLOOD-VOLUME!!
!!ARTERIAL-RESISTANCE!! !!VENOUS-RESISTANCE!!
!!TOTAL-PERIPHERAL-RESISTANCE!!
!!BARORECEPTOR-PRESSURE!! !!CENTRAL-VENOUS-PRESSURE!!
!!VENOUS-TRANSMURAL-PRESSURE!!
!!MEAN-INTRATHORACIC-PRESSURE!!
!!RIGHT-ATRIAL-PRESSURE!! !!END-DIASTOLIC-PRESSURE!!
!!MEAN-ARTERIAL-PRESSURE!! !!CAUSAL-RELATION-BRP/CNSR!!
!!CAUSAL-RELATION-CNSR/CC!! !!CAUSAL-RELATION-AMA/CC!!
!!CAUSAL-RELATION-ICA/AMA!! !!CAUSAL-RELATION-EPI/ICA!!
!!CAUSAL-RELATION-EPI/SANR!!
!!CAUSAL-RELATION-NEPI/ICA!! !!CAUSAL-RELATION-VR/EDV!!
!!CAUSAL-RELATION-CNSR/HR!! !!CAUSAL-RELATION-HR/ICA!!
!!CAUSAL-RELATION-HR/FT!! !!CAUSAL-RELATION-HR/CO!!
!!CAUSAL-RELATION-SFR/NEPI!! ...)
```

V <u>Solving a CV Problem</u>

The DKRS has the capability of utilizing its knowledge to solve a large number of CV problems. As soon as the tutor or the student selects a CV problem a set of messages are sent by the Instructional Planner to the DKRS to solve the selected problem. Each CV parameter in the DKB has capability to store its state information. When the DKRS solves a CV problem the value for each CV parameter in DR, RR, and SS are stored as a part of state information for each parameter object. This state information is accessable at any moment during which CIRCSIM-Tutor (v.3) communicates with the student. In the following example the DKRS has solved a CV problem. The value for each CV parameter in the deep level concept map for DR, RR, and SS is listed as follows.

Procedure: BASIC-PROCEDURE1
Description: A hypertensive patient was given a drug by his physician that reduced his total peripheral resistance by 50%.

Solution path for DR (deep level):

```
(!!SMOOTH-MUSCLE-TONE!! DECREASE)
```

(!!ARTERIOLE-SIZE!! INCREASE)
(!!ARTERIAL-RESISTANCE!! DECREASE)
(!!TOTAL-PERIPHERAL-RESISTANCE!! DECREASE)
(!!ARTERIAL-BLOOD-VOLUME!! DECREASE)
(!!MEAN-ARTERIAL-PRESSURE!! DECREASE)
(!!STROKE-VOLUME!! INCREASE)
(!!CARDIAC-OUTPUT!! INCREASE)
(!!CENTRAL-BLOOD-VOLUME!! DECREASE)
(!!CENTRAL-VENOUS-PRESSURE!! DECREASE)
(!!RIGHT-ATRIAL-PRESSURE!! DECREASE)
(!!END-DIASTOLIC-PRESSURE!! DECREASE)
(!!END-DIASTOLIC-VOLUME!! DECREASE)
(!!MUSCLE-FIBER-LENGTH!! DECREASE)
(!!ACTIN-MYOSIN-REACTION!! DECREASE)
(!!MYOCARD-CONTRACTILE-FORCE!! DECREASE)

Solution path for RR (deep level):

(!!BARORECEPTOR-PRESSURE!! DECREASE)
(!!BARORECEPTOR-SIZE!! DECREASE)
(!!BARORECEPTOR-FIRING-RATE!! DECREASE)
(!!CENTRAL-NERVOUS-SYSTEM-RESPONSE!! DECREASE)
(!!SYMPATHETIC-FIRING-RATE!! INCREASE)
(!!NOREPINEPHRINE!! INCREASE)
(!!SA-NODE-RATE!! INCREASE)
(!!HEART-RATE!! INCREASE)
(!!CARDIAC-OUTPUT!! INCREASE)
(!!ARTERIAL-BLOOD-VOLUME!! INCREASE)
(!!MEAN-ARTERIAL-PRESSURE!! INCREASE)
(!!STROKE-VOLUME!! DECREASE)
(!!CENTRAL-BLOOD-VOLUME!! DECREASE)
(!!CENTRAL-VENOUS-PRESSURE!! DECREASE)
(!!RIGHT-ATRIAL-PRESSURE!! DECREASE)
(!!END-DIASTOLIC-PRESSURE!! DECREASE)
(!!END-DIASTOLIC-VOLUME!! DECREASE)
(!!MUSCLE-FIBER-LENGTH!! DECREASE)
(!!ACTIN-MYOSIN-REACTION!! DECREASE)
(!!MYOCARD-CONTRACTILE-FORCE!! DECREASE)
(!!FILLING-TIME!! DECREASE)
(!!SMOOTH-MUSCLE-TONE!! INCREASE)
(!!ARTERIOLE-SIZE!! DECREASE)
(!!ARTERIAL-RESISTANCE!! INCREASE)
(!!TOTAL-PERIPHERAL-RESISTANCE!! INCREASE)
(!!INTRACELLULAR-CA++!! INCREASE)
(!!ACTIN-MYOSIN-ASSOCIATION!! INCREASE)
(!!CARDIAC-CONTRACTILITY!! INCREASE)
(!!EPINEPHRINE!! INCREASE)
(!!PARASYMPATHETIC-FIRING-RATE!! DECREASE)
(!!ACETYLCHOLINE!! DECREASE)

State of parameters in SS (deep level):

(!!ACETYLCHOLINE!! DECREASE)
(!!NOREPINEPHRINE!! INCREASE)

```
(!!EPINEPHRINE!! INCREASE)
(!!CENTRAL-NERVOUS-SYSTEM-RESPONSE!! DECREASE)
(!!CARDIAC-CONTRACTILITY!! INCREASE)
(!!ACTIN-MYOSIN-REACTION!! DECREASE)
(!!ACTIN-MYOSIN-ASSOCIATION!! INCREASE)
(!!INTRACELLULAR-CA++!! INCREASE)
(!!CARDIAC-OUTPUT!! INCREASE)
(!!FILLING-TIME!! DECREASE)
(!!HEART-RATE!! INCREASE)
(!!PARASYMPATHETIC-FIRING-RATE!! DECREASE)
(!!SYMPATHETIC-FIRING-RATE!! INCREASE)
(!!BARORECEPTOR-FIRING-RATE!! DECREASE)
(!!SA-NODE-RATE!! INCREASE)
(!!MYOCARD-CONTRACTILE-FORCE!! DECREASE)
(!!SMOOTH-MUSCLE-TONE!! DECREASE)
(!!ARTERIOLE-SIZE!! INCREASE)
(!!BARORECEPTOR-SIZE!! DECREASE)
(!!MUSCLE-FIBER-LENGTH!! DECREASE)
(!!STROKE-VOLUME!! INCREASE)
(!!END-DIASTOLIC-VOLUME!! DECREASE)
(!!CENTRAL-BLOOD-VOLUME!! DECREASE)
(!!ARTERIAL-BLOOD-VOLUME!! DECREASE)
(!!ARTERIAL-RESISTANCE!! DECREASE)
(!!TOTAL-PERIPHERAL-RESISTANCE!! DECREASE)
(!!BARORECEPTOR-PRESSURE!! DECREASE)
(!!CENTRAL-VENOUS-PRESSURE!! DECREASE)
(!!RIGHT-ATRIAL-PRESSURE!! DECREASE)
(!!END-DIASTOLIC-PRESSURE!! DECREASE)
(!!MEAN-ARTERIAL-PRESSURE!! DECREASE)
```

APPENDIX E

A PARTIAL TRACE OF THE FUNCTIONING OF THE INSTRUCTIONAL PLANNER

This appendix contains a partial trace of the instructional planner’s decision making process. During execution the instructional planner visits different tutoring states. In each tutoring state it performs various actions. This trace shows various activities performed by the instructional planner. The words “poping-up” in this trace indicates that the instructional planner is jumping from a lower planning level or tutoring space to a higher planning level or tutoring space. Refer Chapters VII and VIII for a detailed description of the design and implementation of the instructional planner of CIRCSIM-Tutor (v.3). This trace is here printed in *italics*.

;;
Visiting Tutoring Space: !!tutoring-space-1!!
;;

Visiting Tutoring Level: !!pedagogical-level/ts-1!!

Visiting Tutoring State: select/ts-1

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select/ts-1=>select-a-goal-selection-approach/ts-1!!)
enabled-rules: (!!select/ts-1=>select-a-goal-selection-approach/ts-1!!)
fired-rule: !!select/ts-1=>select-a-goal-selection-approach/ts-1!!

Visiting Tutoring Level: !!strategical-level/ts-1!!

Visiting Tutoring State: select-a-goal-selection-approach/ts-1

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select-a-goal-selection-approach/ts-1=>separate-approach/ts-1!!
!!select-a-goal-selection-approach/ts-1=>combined-approach/ts-1!!)
enabled-rules:
(!!select-a-goal-selection-approach/ts-1=>combined-approach/ts-1!!)
fired-rule: !!select-a-goal-selection-approach/ts-1=>combined-approach/ts-1!!

Visiting Tutoring Level: !!tactical-level/ts-1!!

Visiting Tutoring State: combined-approach/ts-1

```
Deciding about the next tutoring state.


Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: nil
enabled-rules: nil
fired-rule: nil


!!!!!!!!!!!!!!!!!!!!!!
!! POPING-UP !!
!!!!!!!!!!!!!!!!!!!!!


Visiting Tutoring Level: !!strategical-level/ts-1!!


Deciding about the next tutoring state.


Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select-a-goal-selection-approach/ts-1=>separate-approach/ts-1!!
!!select-a-goal-selection-approach/ts-1=>combined-approach/ts-1!!)
enabled-rules: nil
fired-rule: nil


!!!!!!!!!!!!!!!!!!!!!!
!! POPING-UP !!
!!!!!!!!!!!!!!!!!!!!!


Visiting Tutoring Level: !!pedagogical-level/ts-1!!


Deciding about the next tutoring state.


Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select/ts-1=>select-a-goal-selection-approach/ts-1!!)
enabled-rules: nil
fired-rule: nil


Type of rule/link under consideration: meta-progression-links
valid-rules: (!!select/ts-1=>complete/ts-1!!)
enabled-rules: nil
fired-rule: nil


Type of rule/link under consideration: default-progression-links
valid-rules: (!!select/ts-1=>tutor/ts-1!!)
enabled-rules: (!!select/ts-1=>tutor/ts-1!!)
```

```
fired-rule: !!select/ts-1=>tutor/ts-1!!

Visiting Tutoring Level: !!pedagogical-level/ts-1!!

Visiting Tutoring State: tutor/ts-1

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!tutor/ts-1=>select-a-goal-tutoring-approach/ts-1!!)
enabled-rules: (!!tutor/ts-1=>select-a-goal-tutoring-approach/ts-1!!)
fired-rule: !!tutor/ts-1=>select-a-goal-tutoring-approach/ts-1!!

Visiting Tutoring Level: !!strategical-level/ts-1!!

Visiting Tutoring State: select-a-goal-tutoring-approach/ts-1

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select-a-goal-tutoring-approach/ts-1=>one-shot-act/ts-1!!
!!select-a-goal-tutoring-approach/ts-1=>pre-act-post/ts-1!!)
enabled-rules: (!!select-a-goal-tutoring-approach/ts-1=>one-shot-act/ts-1!!)
fired-rule: !!select-a-goal-tutoring-approach/ts-1=>one-shot-act/ts-1!!

Visiting Tutoring Level: !!tactical-level/ts-1!!

Visiting Tutoring State: one-shot-act/ts-1

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: nil
enabled-rules: nil
fired-rule: nil

Type of rule/link under consideration: default-progression-links
valid-rules: (!!one-shot-act/ts-1=>introduce-system/ts-1!!)
enabled-rules: (!!one-shot-act/ts-1=>introduce-system/ts-1!!)
fired-rule: !!one-shot-act/ts-1=>introduce-system/ts-1!!
```

```
Visiting Tutoring Level: !!tactical-level/ts-1!!


Visiting Tutoring State: introduce-system/ts-1


Deciding about the next tutoring state.


Type of rule/link under consideration: default-progression-links
valid-rules: (!!introduce-system/ts-1=>move-to-next-tutoring-space/ts-1!!)
enabled-rules: (!!introduce-system/ts-1=>move-to-next-tutoring-space/ts-1!!)
fired-rule: !!introduce-system/ts-1=>move-to-next-tutoring-space/ts-1!!


Visiting Tutoring Level: !!tactical-level/ts-1!!


Visiting Tutoring State: move-to-next-tutoring-space/ts-1


Deciding about the next tutoring state.


Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!move-to-next-tutoring-space/ts-1=>next-tutoring-space!!)
enabled-rules: (!!move-to-next-tutoring-space/ts-1=>next-tutoring-space!!)
fired-rule: !!move-to-next-tutoring-space/ts-1=>next-tutoring-space!!


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Visiting Tutoring Space: !!tutoring-space-2!!
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


Visiting Tutoring Level: !!pedagogical-level/ts-2!!


Visiting Tutoring State: select/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!select/ts-2=>who-should-choose-next-procedure/ts-2!!)
enabled-rules: (!!select/ts-2=>who-should-choose-next-procedure/ts-2!!)
fired-rule: !!select/ts-2=>who-should-choose-next-procedure/ts-2!!


Visiting Tutoring Level: !!strategical-level/ts-2!!
Visiting Tutoring State: who-should-choose-next-procedure/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-in-level-refinement-links
valid-rules: (!!who-should-choose-next-procedure/ts-2=>students-choice/ts-2!!
!!who-should-choose-next-procedure/ts-2=>tutors-choice/ts-2!!)
enabled-rules: (!!who-should-choose-next-procedure/ts-2=>tutors-choice/ts-2!!)
fired-rule: !!who-should-choose-next-procedure/ts-2=>tutors-choice/ts-2!!


Visiting Tutoring Level: !!strategical-level/ts-2!!


Visiting Tutoring State: tutors-choice/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-in-level-refinement-links
valid-rules: nil
enabled-rules: nil
fired-rule: nil


Type of rule/link under consideration: default-progression-links
valid-rules: (!!tutors-choice/ts-2=>select-procedure-category/ts-2!!)
enabled-rules: (!!tutors-choice/ts-2=>select-procedure-category/ts-2!!)
fired-rule: !!tutors-choice/ts-2=>select-procedure-category/ts-2!!


Visiting Tutoring Level: !!strategical-level/ts-2!!


List of candidate CV procedures:
(!!goal-9/ts-2!! !!goal-11/ts-2!! !!goal-12/ts-2!!)


Visiting Tutoring State: select-procedure-category/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-progression-links
valid-rules:
(!!select-procedure-category/ts-2=>select-procedure-difficulty-level/ts-2!!)
enabled-rules:
(!!select-procedure-category/ts-2=>select-procedure-difficulty-level/ts-2!!)
fired-rule:
!!select-procedure-category/ts-2=>select-procedure-difficulty-level/ts-2!!
```

```
Visiting Tutoring Level: !!strategical-level/ts-2!!


List of candidate procedures after deciding about the difficulty level:
(!!goal-9/ts-2!! !!goal-11/ts-2!!)


Visiting Tutoring State: select-procedure-difficulty-level/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-progression-links
valid-rules:
(!!select-procedure-difficulty-level/ts-2=>select-procedure-description/ts-2!!)
enabled-rules:
(!!select-procedure-difficulty-level/ts-2=>select-procedure-description/ts-2!!)
fired-rule:
!!select-procedure-difficulty-level/ts-2=>select-procedure-description/ts-2!!


Visiting Tutoring Level: !!strategical-level/ts-2!!


List of candidate procedures after deciding about the description level:
(!!goal-11/ts-2!!)


Visiting Tutoring State: select-procedure-description/ts-2


Deciding about the next tutoring state.


Type of rule/link under consideration: default-progression-links
valid-rules:
(!!select-procedure-description/ts-2=>present-selected-procedure/ts-2!!)
enabled-rules:
(!!select-procedure-description/ts-2=>present-selected-procedure/ts-2!!)
fired-rule:
!!select-procedure-description/ts-2=>present-selected-procedure/ts-2!!


Visiting Tutoring Level: !!strategical-level/ts-2!!


Visiting Tutoring State: present-selected-procedure/ts-2


Deciding about the next tutoring state.
```

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: (!!present-selected-procedure/ts-2=>describe-procedure/ts-2!!
!!present-selected-procedure/ts-2=>give-menu/ts-2!!)
enabled-rules: (!!present-selected-procedure/ts-2=>describe-procedure/ts-2!!)
fired-rule: !!present-selected-procedure/ts-2=>describe-procedure/ts-2!!

Visiting Tutoring Level: !!tactical-level/ts-2!!

Visiting Tutoring State: describe-procedure/ts-2

Deciding about the next tutoring state.

Type of rule/link under consideration: default-between-levels-refinement-links
valid-rules: nil
enabled-rules: nil
fired-rule: nil

Type of rule/link under consideration: default-progression-links
valid-rules: (!!describe-procedure/ts-2=>setup-tutoring-environment/ts-2!!)
enabled-rules: (!!describe-procedure/ts-2=>setup-tutoring-environment/ts-2!!)
fired-rule: !!describe-procedure/ts-2=>setup-tutoring-environment/ts-2!!

.
.
.
.

REFERENCES

Anania, J. (1983). The influence of instructional conditions on student learning and achievement. *Evaluation in Education*, 7, 1 - 92.

Anderson, J. R. (1980). *Cognitive psychology and its implications*. San Francisco, CA: W. H. Freeman and Company.

Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J. R. (1988). The expert module. In Polson, M. C. & Richardson, J. J. (Eds.). *Foundations of intelligent tutoring systems* (pp. 21-53). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Anderson, J. R. (1992). Intelligent tutoring and high school mathematics. In Frasson, C., Gauthier, G. & McCalla, G. I. (Eds.), *Intelligent Tutoring Systems: 2nd International Conference* (pp. 1-10). Berlin: Springer-Verlag.

Anderson, J. R., Boyle, C. F., Corbett, A. T. & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. In Clancey, W. J. & Soloway, E. (Eds.). *Artificial intelligence and learning environment* (pp. 7-49). Cambridge, MA: The MIT Press.

Anderson, J. R., Boyle, C. & Reiser, B. J. (1985). Intelligent tutoring systems. *Science,* 228, 456 - 462.

Anderson, J. R., Conrad, F. G. & Corbett, A. T. (1989). Skill acquisition and the Lisp tutor. *Cognitive Science*, 13, 467 - 505.

Anderson, R. C. (1984). Some reflections on the acquisition of knowledge. *Educational Researcher*, 5 - 10.

Aronson, D. T. & Briggs, L. J. (1983). Contributions of Gagne and Briggs to a prescriptive model of instruction. In Reigeluth, C. M. (Ed.), *Instructional design theories and models: An overview of their current status* (pp. 75-100). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Ausubel, D. P. (1968). *Educational psychology: A cognitive view*. New York: Holt, Rinehart & Winston.

Barr, A., Beard, M. & Alkinson, R. C. (1976). The computer as a tutorial laboratory: The Stanford BIP project. *International Journal of Man-Machine Studies*, 8, 567 - 596.

Barr, A. & Feigenbaum, E. A. (1982). (Eds.). Applications-oriented AI research: Education. *Handbook of artificial intelligence*, vol. II. Reading, MA: Addison-Wesley.

Barrows, H. S. & Tamblyn, R. M. (1980). *Problem-based learning: An approach to medical education*. New York: Springer Publishing Company.

Berliner, D. (1986). In pursuit of the expert pedagogy, *Educational Researcher*, 15, 5 - 13.

Berne, R. M. & Levy, M. N. (1993). *Physiology*, 3rd edition. St. Louis: C. V. Mosby.

Bloom, B. S. (1956). (Ed.). *Taxonomy of educational objectives: The classification of educational goals*. Handbook 1, cognitive domain. New York: McKay.

Bloom, B. S. (1984). The 2-sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 13, 4 - 16.

Bonar, J. G. (1984). Cognition based intelligent tutoring systems. In Salvendy, G. (Ed.), *Human-computer interaction.* Elsevier Science Publishers.

Booch, G. (1991). *Object-oriented design with applications.* Redwood City, CA: The Benjamin/Cummings Publishing Company, Inc.

Brachman, R. J. (1985). On the epistemological status of semantic networks. In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings in knowledge representation* (pp. 191-215). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Brachman, R. J. & Levesque, H. J. (1985). *Readings in knowledge representation*. Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Breuker, J. (1988). Coaching in help systems. In Self, J. (Ed.), *Artificial intelligence and human learning* (pp. 310-337). London: Chapman and Hall Computing.

Breuker, J. (1990). Conceptual model of intelligent help system. In Breuker, J. (ed.), *EUROHELP: developing intelligent help systems* (pp. 41-67). Copenhagen: EC.

Brown, J. S. (1983). Learning by doing revisited for electronic learning environments. In White, M. A. (Ed.), *The future of electronic learning*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Brown, J. S. (1990). Towards a new epistemology of learning. In Frasson, C. & Gauthier, G. (Eds.), *Intelligent tutoring systems: At the crossroads of artificial intelligence and education* (pp. 266-282). Norwood, NJ: Ablex Publishing Corporation.

Brown, J. S. & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2, 155 - 191.

Brown, J. S., Burton, R. R. & deKleer, J. (1982). Pedagogical, natural language, and knowledge engineering techniques in SOPHIE-I, II, and III. In Sleeman, D. & Brown, J. S. (Eds.), *Intelligent tutoring systems* (pp. 227-282). London: Academic Press, Inc.

Brown, J. S., Collins, A. & Duguid, P. (1989a). Situated cognition and the culture of learning, *Educational Researcher*, 18, 1, 32 - 42.

Brown, J. S., Collins, A. & Duguid, P. (1989b). Debating the situation: A rejoinder to Wineburg and Palincsar, *Educational Researcher*, 18, 4, 10 - 12.

Brown, J. S. & deKleer, J. (1984). A framework for a qualitative physics. *Proceedings of the sixth cognitive science society conference*. Boulder, CO.

Bruner, J. S. (1966). *Towards a theory of instruction*. New York: W. W. Norton & Co.

Burns, H. & Parlett, J. W. (1991). The evolution of intelligent tutoring systems: Dimensions of design. In Burns, H., Parlett, J. W. & Redfield, C. L. (Eds.), *Intelligent tutoring systems: Evolutions in design* (pp. 1-11). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Burton, R. R. (1982). Diagnosing bugs in a simple procedural skill. In Sleeman, D. & Brown, J. S. (Eds.), *Intelligent tutoring systems* (pp. 157-183). London: Academic Press, Inc.

Burton, R. R. & Brown, J. S. (1979). An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, 11, 5 - 24.

Calderhead, J. (1988). *Teachers' professional learning*. London: The Falmer Press.

Carbonell, J. R. (1970a). *Mixed-initiative man-computer instructional dialogues*. Doctoral Dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Carbonell, J. R. (1970b). AI in CAI: An artificial intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems*, 11, 4, 190 - 202.

Chi, M., Glaser, R. & Farr, M. (1988). (Eds.). *The nature of expertise*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Clancey, W. J. (1985). Heuristic classification. *Artificial Intelligence*, 27, 3, 289 - 350.

Clancey, W. J. (1986). Qualitative student models. *Annual review of computer science*, 1, 391 - 450.

Clancey, W. J. (1987a). *Intelligent tutoring systems: A tutorial survey*. Report No. STAN-CS-87-1174.

Clancey, W. J. (1987b). *Knowledge-based tutoring: The GUIDON program*. Cambridge, MA: The MIT Press.

Clancey, W. J. (1992). Guidon-Manage Revisited: A socio-technical systems approach. In Frasson, C., Gauthier, G. & McCalla, G. I. (Eds.), *Intelligent Tutoring Systems: 2nd International Conference* (pp. 21-36). Berlin: Springer-Verlag.

Clancey, W. J. (1993). The knowledge level reinterpreted: Modeling socio-technical systems. In Ford, K. M. & Bradshaw, J. M. (Eds.), *Knowledge acquisition as modeling.* New York: John Wiley and Sons, Inc.

Cohen, P. A., Kulik, J. A. & Kulik, C. (1982). Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal*, 19, 237 - 248.

Collins, A. (1985). Component models of physical systems. In *Proceedings of the seventh cognitive science society conference*. Irvine, CA.

Collins, A., Brown, J. S. & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the craft of reading writing and mathematics. In Resnick, L. B. (Ed.), *Knowing, learning and instruction* (pp. 453-494). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Collins, A. & Gentner, D. (1983). Multiple models of evaporation processes. In *Proceedings of the fifth cognitive science society conference*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Collins, A. & Stevens, A. L. (1982). Goals and strategies of inquiry teachers. In Glaser, R. (Ed.). *Advances in instructional psychology*, vol. II, (pp. 65-119). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Collins, A. & Stevens, A. L. (1991). A cognitive theory of inquiry teaching. In Goodyear, P. (Ed.), *Teaching knowledge and intelligent tutoring* (pp. 203-230). Norwood, NJ: Ablex Publishing Corporation.

Collins, A., Warnock, E. H. & Passafiume, J. J. (1975). Analysis and synthesis of tutorial dialogues. In Bower, G. H. (Ed.), *The psychology of learning and motivation*, 9, (pp. 49-87). Academic Press.

Davis, R., Buchanan, B. & Shortliffe, E. (1985). Production rules as a representation for a knowledge-based consultation program. In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings in knowledge representation* (pp. 371-387). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

deKleer, J. & Brown, J. S. (1983). Assumptions and ambiguities in mechanistic mental models. In Gentner, D. & Stevens, A. L. (Eds.). *Mental models* (pp. 155-190). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

deKleer, J. Doyle, J., Steele, G. L. Jr. & Sussman, G. J. (1985). AMORD: Explicit control of reasoning. In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings on knowledge representation* (pp. 345-355). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Derry, S. J., Hawkes, L. W. & Ziegler, U. (1988). A plan-based opportunistic architecture for intelligent tutoring. In *Proceedings of the second international conference on intelligent tutoring systems* (pp. 116-123). Montreal, Canada.

Dickinson, C. J., Goldsmith, C.H. & Sackett, D. L. (1973). MACMAN: A digital computer model for teaching some basic principles of hemodynamics. *Journal of Clinical Computing*, 2, 42 - 50.

Elbaz, F. (1981). The teacher's "practical knowledge": Report of a case study. *Curriculum Inquiry*, 11, 43 - 71.

Ellson, D. G. (1976). Tutoring. In Gage, N. L. (Ed.). *The psychology of teaching methods*. Chicago, IL: University of Chicago Press.

Ericsson, K. A. & Simon, H. A. (1993). *Protocol Analysis: Verbal Reports as Data.* Cambridge, MA: The MIT Press.

Evens, M. W., Spitkovsky, J., Boyle, P., Michael, J. A. & Rovick, A. A. (1993). Synthesizing tutorial dialogues. *Proceedings of the Cognitive Science Society* (pp. 137-142). Los Alamitos, CA: IEEE Computer Society.

Feigenbaum, E. A. & McCorduck, P. (1983). *The fifth generation.* New York: Addison-Wesley.

Fletcher, J. D. (1984). Intelligent instructional systems in training. In Andriole, S. J. (Ed.). *Applications in artificial intelligence*. Petrocelli Books.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24, 85 - 168.

Ford, K. M. & Bradshaw, J. M. (1993a). (Eds.). *Knowledge acquisition as modeling.* New York: John Wiley and Sons, Inc.

Ford, K. M. & Bradshaw, J. M. (1993b). Introduction: Knowledge acquisition as modeling. In Ford, K. M. & Bradshaw, J. M. (Eds.), *Knowledge acquisition as modeling.* New York: John Wiley and Sons, Inc.

Ford, K. M., Bradshaw, J. M., Adams-Webber, J. R. & Agnew, N. M. (1993). Knowledge acquisition as a constructive modeling activity. In Ford, K. M. & Bradshaw, J. M. (Eds.), *Knowledge acquisition as modeling.* New York: John Wiley and Sons, Inc.

Fox, B. A. (1993). *The human tutorial dialogue project*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Fuller, R. B. (1962). *Education automation: Freeing the scholar to return to his studies*. Carbondale, Ill: Southern Illinois University Press.

Gagne, R. M. & Briggs, L. J. (1979). *Principles of instructional design*, 2nd edition. New York: Holt, Rinehart & Winston.

Galdes, D. K. (1990). *An empirical study of human tutors: The implications for intelligent tutoring systems*. Unpublished Doctoral Dissertation, Department of Industrial and Systems Engineering, Ohio State University, Ohio.

Gentner, D. & Stevens, A. L. (1983). (Eds.). *Mental models*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Goldstein, I. P. (1982). The genetic graph: A representation for the evolution of procedural knowledge. In Sleeman, D. & Brown, J. S. (Eds.), *Intelligent tutoring systems* (pp. 51-77). London: Academic Press, Inc.

Goodyear, P. (1991). Research on teaching and the design of intelligent tutoring systems. In Goodyear, P. (Ed.), *Teaching knowledge and intelligent tutoring* (pp. 3-23). Norwood, NJ: Ablex Publishing Corporation.

Halff, H. M. (1988). Curriculum and instruction in automated tutors. In Polson, M. C. & Richardson, J. J. (Eds.). *Foundations of intelligent tutoring systems* (pp. 79-108). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Hayman, R. T. (1974). *Ways of teaching*, 2nd edition. J. B. Lippincott Company.

Hume, G. D. (in preparation). *Tutoring to the capability of the student: Using student modeling to decide when and how to generate hints.* Ph.D. Thesis, Computer Science Department, Illinois Institute of Technology, Chicago, Illinois.

Hume, G. D., Michael, J. A., Rovick, A. A. & Evens, M. W. (1993). The use of hints as a tutorial tactic. In *Proceedings of the Cognitive Science Society*. Los Alamitos, CA: IEEE Computer Society.

Imbeau, G., Frasson, C. & Gauthier, G. (1988). A multiple-expert system for large scale intelligent tutoring. In *Proceedings of the first instructional conference on intelligent tutoring systems* (pp. 272-278), Montreal, Canada.

Johnson, W. L. & Soloway, E. M. (1984a). PROUST: knowledge-based program debugging. *Proceedings of the seventh international software engineering conference*. Orlando, Florida.

Johnson, W. L. & Soloway, E. (1984b). Intention-based diagnosis of programming errors. *Proceedings of the national conference on artificial intelligence*. Austin, Texas.

Jonas, S. (1978). *Medical mystery*. New York: W. W. Norton & Company.

Jones, A. A., Bagford, L. W. & Wallen, E. A. (1979). *Strategies for teaching*. The Scarecrow Press.

Jordan, J. A. Jr. (1963). *Socratic teaching*. Harvard Educational Review, 33, 1, 96 - 104.

Kearsley, G. (1987). Overview. In Kearsley, G. (Ed.). *Artificial intelligence and instruction: Applications and methods* (pp. 3-10). Reading, MA: Addison-Wesley Publishing Company.

Keene, S. E. (1989). *Object-oriented programming in Common Lisp: A programmer's guide to CLOS.* Reading, MA: Addison-Wesley Publishing Company.

Khuwaja, R. A., Evens, M. W., Rovick, A. A. & Michael, J. A. (1992). Knowledge representation for an intelligent tutoring system based on a multilevel causal model. *Proceedings of the ITS'92* (pp. 217-224). New York: Springer-Verlag.

Khuwaja, R. A., Evens, M. W., Rovick, A. A. & Michael, J. A. (1993). Building the domain expert for a cardiovascular physiology tutor. *Proceedings of the 6th Annual IEEE Computer-Based Medical Systems Symposium* (pp. 165-170). Ann Arbor, MI: IEEE Computer Society.

Khuwaja, R. A., Evens, M. W., Rovick, A. A. & Michael, J. A. (1994a). Architecture of CIRCSIM-Tutor (v.3): A smart cardiovascular physiology tutor. *Proceedings of the 7th Annual IEEE Computer-Based Medical Systems Symposium*. (pp. 158-163). NC: IEEE Computer Society.

Khuwaja, R. A., Rovick, A. A., Michael, J. A. & Evens, M. W. (1994b). A tale of three protocols: The implications for intelligent tutoring systems. *Proceedings of the GW International Conference on Intelligent Systems.* Las Vegas, NV.

Khuwaja, R. A., Rovick, A. A., Michael, J. A. & Evens, M. W. (in preparation (a)). *The inference triangle: A qualitative causal reasoning tool used by the effective tutor*.

Khuwaja, R. A., Michael, J. A., Rovick, A. A. & Evens, M. W. (in preparation (b)). *Modeling domain expertise for an intelligent tutoring system: using a multi-dimensional knowledge representation technique.*

Kim, N. (1989). *CIRCSIM-TUTOR: An intelligent tutoring system for circulatory physiology.* Ph.D. Thesis, Computer Science Department, Illinois Institute of Technology, Chicago, Illinois.

Kim, N., Evens, M., Michael, J. A. & Rovick, A. A. (1989). CIRCSIM-TUTOR: An intelligent tutoring system for circulatory physiology. In *Proceedings of the International Conference on Computer-Assisted Learning* (pp. 254-266). Berlin: Springer-Verlag.

Lampert, M. (1984). Teaching about thinking and thinking about teaching. *Journal of Curriculum Studies*, 6, 1 - 18.

Landa, L. N. (1983). Descriptive and prescriptive theories of learning and instruction: An analysis of their relationship and interactions. In Reigeluth, C. M. (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 55-73). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Lee, Y-H, Evens, M. (1992). Ill-formed input handling system for an intelligent tutoring system. In *Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence* (pp. 354-360). Seoul, Korea.

Lee, Y-H, Evens, M., Michael, J., and Rovick, A. (1990). IFIHS: Ill-Formed Input Handling System. In *Proceedings of the Second Midwest Artificial Intelligence and Cognitive Science Conference* (pp. 93-97). Carbondale, Illinois.

Lee, Y-H, Evens, M., Michael, J., and Rovick, A. (1991). Spelling correction for an intelligent tutoring system. In *Proceedings of the First Great Lakes Computer Science Conference* (pp. 18-20). New York: Springer.

Leinhardt, G. & Greeno, J. G. (1986). The cognitive skill of teaching. *Journal of Educational Psychology*, 78, 2, 75 - 95.

Leinhardt, G. & Greeno, J. G. (1991). The cognitive skill of teaching. In Goodyear, P. (Ed.), *Teaching knowledge and intelligent tutoring* (pp. 233-268). Norwood, NJ: Ablex Publishing Corporation.

Lesgold, A. (1988). Towards a theory of curriculum for use in designing intelligent instructional systems. In Mandl, H. & Lesgold, A. (Eds.), *Learning issues for intelligent tutoring systems* (pp. 114-137). New York: Springer-Verlag.

Lesgold, A., Ivill-Friel, J. & Bonar, J. (1989). Toward intelligent tutoring systems for testing. In Resnick, L. B. (Ed.). *Knowing, learning and instruction: Essays in honor of Robert Glaser* (pp. 337-360). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Li, J., Rovick, A. & Michael, J. (1992). ABASE: A Hypermedia-Based Tutoring and Authoring System. In *Proceedings of the 4th International Conference on Computer Assisted Learning*. New York: Springer-Verlag.

Li, J., Seu, J., Evens, M., Michael, J. & Rovick, A. (1992). Computer dialogue system (CDS): A system for capturing computer-mediated dialogue. *Behavior Research Methods, Instruments, & Computers*, 24, 535 - 540.

Littman, D. C. (1990). *Strategies for tutoring multiple bugs.* Ph.D. Dissertation, Yale University, Pittsburgh, Pennsylvania.

Littman, D. (1991). Tutorial planning schemas. In Goodyear, P. (Ed.), *Teaching knowledge and intelligent tutoring* (pp. 107-122). Norwood, NJ: Ablex Publishing Corporation.

Littman, D., Pinto, J. & Soloway, E. (1985). Observations on tutorial expertise. In *Proceedings of IEEE conference on expert systems in government.* Washington, DC: IEEE.

Macmillan, S. A., Emme, D. & Berkowitz, M. (1988). Instructional planners: Lessons learned. In Psotka, J., Massey, L. D. & Mutter, S. A. (Eds.), *Intelligent tutoring systems: Lessons learned* (pp. 229-256). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

McArthur, D., Stasz, C. & Zmuidzinas, M. (1990). Tutoring techniques in algebra. *Cognition and Instruction*, 7, 197 - 244.

McGuire, C. H., Foley, R. P., Gorr, A., Richards, R. W. & Associates (1983). *Handbook of health professions education*. San Francisco, CA: Jossey-Bass Publishers.

McKendree, J., Reiser, B. J. & Anderson, J. R. (1984). Tutorial goals and strategies in the instruction of programming skills. *Proceedings of the sixth annual conference of the cognitive science society*. Boulder, CO.

Merrill, D. C., Reiser, B. J., Ranney, M. & Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *Journal of the Learning Sciences,* 2, 277 - 305.

Merrill, M. D. (1983). Component display theory. In Reigeluth, C. M. (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 279-333). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Michael, J. A. (1989). An agenda for research on teaching of physiology. *American Journal of Physiology*, 256 (*Advances in Physiology Education,* 1), S14 - S17.

Michael, J. A. (1993). Teaching problem solving in small groups. *Ann. NY Acad. Sci.*, 701, 37-48.

Michael, J. A. & Modell, H. (1993). Life science education: Reflections on some of the challenges facing us. *Ann. NY Acad. Sci.*, 701, 83-90.

Michael, J. A., Rovick, A. A., Evens, M. W., Shim, L., Woo, C., & Kim, N. (1992). The uses of multiple student inputs in modeling and lesson planning in CAI and ICAI programs. *Proceedings of the 4th ICCAL conference* (pp. 441-452). Berlin: Springer-Verlag.

Minsky, M. (1985). A framework for representing knowledge. In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings on knowledge representation* (pp. 245-262). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Molnar, A. (1986). An unpublished talk presented on the panel “AI in Education,” Soloway, E., chair, National meeting of the American Association on Artificial Intelligence. Philadelphia, PA.

Moore, R. C. (1985). The role of logic in knowledge representation and commonsense reasoning. In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings on knowledge representation* (pp. 335-341). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Murray, W. R. (1988a). *Dynamic instructional planning in the BB1 blackboard architecture*. Technical report, R-6168, FMC corporation.

Murray, W. R. (1988b). *Control for intelligent tutoring systems: A comparison of blackboard architectures and discourse management networks.* Technical Report R-6267, FMC corporation.

Naisbitt, J. (1984). *Megatrends: Ten new directions transforming our lives.* New York: Warner Books.

National Science Foundation (1983). *Educating America for the 21th century*. Washington, D. C.

Norman, D. A. (1973). Memory, knowledge, and answering of questions. In Solso, R. L. (Ed.), *Contemporary issues in cognitive psychology: The Loyola symposium*. Washington, DC: Winston.

Norman, D. A. (1983). Some observations on mental models. In Gentner, D. & Stevens, A. L. (Eds.), *Mental models* (pp. 7-14). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Novak, J. D. & Gowin, D. B. (1984). *Learning How to Learn*. Cambridge, MA: Cambridge University Press.

Ohlsson, S. (1987). Some principles of intelligent tutoring. In Lawler, R. & Yazdani, M. (Eds.), *AI and education: Learning environments and intelligent tutoring systems* (pp. 203-237). Norwood, NJ: Ablex Publishing.

Ohlsson, S. (1991). Knowledge requirements for teaching: The case of fractions. In Goodyear, P. (Ed.), *Teaching knowledge and intelligent tutoring* (pp. 25-59). Norwood, NJ: Ablex Publishing Corporation.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, New York.

Park, O., Perez, R. S. & Seidel, R. J. (1987). Intelligent CAI: Old wine in new bottles, or a new vintage? In Kearsley, G. (Ed.). *Artificial intelligence and instruction: Applications and methods* (pp. 11-45). Reading, MA: Addison-Wesley Publishing Company.

Patel, R. S. (1988). Artificial intelligence techniques for diagnostic reasoning in medicine. In Shrobe, H. E. & AAAI (Eds.), *Exploring artificial intelligence* (pp. 347-379). San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Patel, R. S., Szolovits, P. & Schwartz, W. B. (1984). Causal understanding of patient illness in medical diagnosis. In Clancey, W. J. & Shortliffe, E. H. (Eds.). *Readings in medical artificial intelligence: The first decade* (pp. 339-360). Reading, MA: Addison-Wesley.

Peachey, D. R. & McCalla, G. I. (1986). Using planning techniques in intelligent tutoring systems. *International Journal of Man-Machine Studies*, 24, 77 - 98.

Phillips, E. M. & Pugh, D. S. (1987). *How to get a PhD: Managing the peaks and troughs of research.* Milton Keynes, England: Open University Press.

Pirolli, P. L. & Greeno, J. G. (1988). The problem space of instructional design. In Psotka, J., Massey, L. D. & Mutter, S. A. (Eds.), *Intelligent tutoring systems: Lessons learned* (pp. 181-201). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Plato (1974). The MENO. In Hayman, R. T. (Ed.), *Teaching: Vantage points for study*, 2nd edition. J. B. Lippincott Company.

Polson, M. C. & Richardson, J. J. (1988). (Eds.). *Foundation of intelligent tutoring systems*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Popham, W. J. & Baker, E. L. (1970a). *Establishing instructional goals*. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Popham, W. J. & Baker, E. L. (1970b). *Planning an instructional sequence.* Englewood Cliffs, NJ: Prentice-Hall, Inc.

Putnam, R. T. (1987). Structuring and adjusting contents for students: A study of live and simulated tutoring of addition. *American Educational Research Journal*, 24, 1, 13 - 48.

Reichgelt, H. (1991). *Knowledge representation: An AI perspective*. Norwood, NJ: Ablex Publishing Corporation.

Reigeluth, C. M. (1983a). (Ed.). *Instructional-design theories and models: An overview of their current status*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Reigeluth, C. M. (1983b). Instructional design: What is it and why is it? In Reigeluth, C. M. (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 3-36). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Reigeluth, C. M. (1987). (Ed.). *Instructional theories in action: Lessons illustrating selected theories and models.* Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Reigeluth, C. M. & Stein, F. S. (1983). The elaboration theory of instruction. In Reigeluth, C. M. (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 335-381). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Reigeluth, C. M. & Merrill, M. D. (1979). Classes of instructional variables. *Educational Technology*, 5 - 24.

Reigeluth, C. M., Merrill, M. D., Wilson, B. G. & Spiller, R. T. (1980). The elaboration theory of instruction: A model for sequencing and synthesizing instruction. *Instructional Science*, 9, 195 - 219.

Reiser, B. J., Beekelaar, R., Tyle, A. & Merrill, D. C. (1991). GIL: Scaffolding learning to program with reasoning-congruent representations. *Proceedings of the international conference of the learning sciences*. Evanston, Ill: Association for the advancement of computing in education.

Resnick, L. B. (1977). On holding an instructional conversation: Comments on chapter 10 by Collins. In Anderson, R. C., Spiro, R. J. & Montague, W. E. (Eds.), *Schooling and the acquisition of knowledge* (pp. 365-370). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Resnick, L. B. (1983). Mathematics and science learning: A new conception. *Science*, 220, 477 - 478.

Romiszowski, A. J. (1981). *Designing instructional systems*. London: Kogan Page Ltd.

Romiszowski, A. J. (1984). *Producing instructional systems*. London: Kogan Page Ltd.

Rothstein, W. G. (1987). *American medical schools and the practice of medicine*. New York: Oxford University Press.

Rovick, A. A. & Brenner, L. (1983). HEARTSIM: A cardiovascular simulation with didactic feedback. *The Physiologist*, 26, 236 - 239.

Rovick, A. A. & Michael, J. A. (1986). CIRCSIM: An IBM PC computer teaching exercise on blood pressure regulation. In *Proceedings of XXX IUPS Congress*. Vancouver, Canada.

Rovick, A. A. & Michael, J. A. (1991). GASP-PC: A computer teaching program on the chemical control of breathing. *Federation of American Societies for Experimental Biology Annual Meeting* (p. 474). Atlanta, GA.

Rovick, A. A. & Michael, J. A. (1992). The prediction table: A tool for assessing students' knowledge. *American Journal of Physiology*, 263 (*Advances in Physiology Education*, 8), S33 - S36.

Rumelhurt, J. McClelland & The PDP research group (1986). (Eds.). *Parallel distributed processing: Explorations on the microstructure of cognition*, 1. Cambridge, MA: The MIT Press.

Russell, D. M. (1988). IDE: The interpreter. In Psotka, J., Massey, L. D. & Mutter, S. A. (Eds.), *Intelligent tutoring systems: Lessons learned* (pp. 323-349). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Sanders, G., Evens, M., Hume, G., Rovick, A. A. & Michael, J. A. (1992). An analysis of how students take the initiative in keyboard-to-keyboard dialogues in a fixed domain. In *Proceedings of the Cognitive Science Society*. Los Alamitos, CA: IEEE Computer Society.

Self, J. (1988). (Ed.). *Artificial intelligence and human learning: Intelligent computer-aided instruction*. London: Chapman & Hall Computing.

Seu, J. H., Evens, M., Michael, J. & Rovick, A. (1991). Understanding ill-formed input to an intelligent tutoring system in an LFG framework. *Proceedings of the Third Midwest Artificial Intelligence and Cognitive Science Society Conference* (pp. 36-40). Carbondale, IL.

Shim, L., Evens, M. W., Michael, J. A. & Rovick, A. A. (1991). Effective cognitive modelling in an intelligent tutoring system for cardiovascular physiology. *Proceedings of the 4th Annual IEEE Symposium on Computer-Based Medical Systems* (pp. 338-345). Baltimore, MD: IEEE Computer Society.

Sime, J. & Leitch, R. (1992). A learning environment based on multiple qualitative models. In Frasson, C., Gauthier, G. & McCalla, G. I. (Eds.), *Intelligent Tutoring Systems: 2nd International Conference* (pp. 116-123). Berlin: Springer-Verlag.

Smith, B. C. (1985). Prologue to “Reflection and semantics in a procedural language.” In Brachman, R. J. & Levesque, H. J. (Eds.), *Readings on knowledge representation* (pp. 31-40). Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Spohrer, J. C. & Soloway, E. (1986). Novice mistakes: Are the folk wisdoms correct? *Communications of the ACM*, 29, 7, 624- 632.

Stevens, A. L. & Collins, A. (1980). Multiple conceptual models of a complex system. In Snow, R., Frederico, P. & Montague, W. (Eds.), *Aptitude, learning and instruction*, vol. II, (pp. 177-197). Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

Stevens, A., Collins, A. & Goldin, S. E. (1982). Misconceptions in student’s understanding. In Sleeman, D. & Brown, J. S. (Eds.), *Intelligent tutoring systems* (pp. 13-24). London: Academic Press, Inc.

Stevens, A. L. & Roberts, B. (1983). Quantitative and qualitative simulations in computer-based training. *Journal of Computer-Based Instruction*, 10, 1, 16- 19.

Sukthankar, S., Ramachandran, K., Evens, M., Rovick, A., and Michael, J. (1993). Graphical user interface with domain visualization for an intelligent medical tutoring system. In *Proceedings of the 6th Annual IEEE Computer-Based Medical Systems Symposium* (pp. 13-16). Ann Arbor, MI: IEEE Computer Society.

VanLehn, K. (1988). Student modeling. In Polson, M. C. & Richardson, J. J. (Eds.). *Foundations of intelligent tutoring systems*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers.

VanLehn, K. (1993). How people learn from a tutor. Unplished research proposal, Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA.

Wenger, E. (1987). *Artificial intelligence and tutoring systems*. Los Altos, CA: Morgan Kaufman.

Wescourt, K., Beard, M. & Gould, L. (1977). Knowledge-based adaptive curriculum sequencing for CAI: Application of a network representation. In *Proceedings of the national ACM conference*. Seattle, Washington.

Wheatley, G. H. (1991). Constructivist perspectives on science and mathematical learning. *Science Education*, 75, 9 - 21.

White, B. Y. & Frederiksen, J. R. (1986). Qualitative models and intelligent learning environments. In Lawler, R. & Yazdani, M. (Eds.), *AI and education: Learning environments and intelligent tutoring systems* (pp. 281-305). Norwood, NJ: Ablex Publishing.

White, B. Y. & Frederiksen, J. R. (1990). Causal model progressions as a foundation for intelligent learning environments. In Clancey, W. J. & Soloway, E. (Eds.), *Artificial intelligence and learning environments* (pp. 99-157). Cambridge, MA: The MIT press.

Wielinga, B. J., Bredeweg, B. & Breuker, J. A. (1987). Knowledge acquisition for expert systems. In Nossum, R. T. (Ed.), *Advanced topics in artificial intelligence*. Springer-Verlag.

Wielinga, B. J. & Breuker, J. A. (1990). Models of expertise. *International Journal of Intelligent Systems*, 5, 497 - 509.

Winston, P. H. (1984). *Artificial intelligence*, 2nd edition. Reading, MA: Addison-Wesley Publishing Company.

Woo, C. W. (1991). *Instructional planning in an intelligent tutoring system: Combining global lesson plans with local discourse control.* Ph.D. Thesis, Computer Science Department, Illinois Institute of Technology, Chicago, Illinois.

Woo, C., Evens, M., Michael, J. A. & Rovick, A. A. (1991). Dynamic instructional planning for an intelligent physiology tutoring system. In *Proceedings of the fourth IEEE symposium on computer-based medical systems* (pp. 226-233). Baltimore, MD: IEEE Computer Society Press.

Woolf, B. P. (1984). *Context dependent planning in a machine tutor*. Ph.D. Dissertation, University of Massachusetts, Amherst, Massachusetts.

Woolf, B. (1986). Intelligent tutoring systems: A survey. In Shrobe, H. E. & American Association for Artificial Intelligence (Eds.), *Exploring artificial intelligence* (pp. 1-43). San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Woolf, B. (1988a). 20 years in the trenches: What have we learned? *Proceedings of the 1st international conference on intelligent tutoring systems* (pp. 33-39). Montreal, Canada.

Woolf, B. P. (1988b). Representing complex knowledge in an intelligent machine tutors. In Self, J. (Ed.), *Artificial intelligence and human learning* (pp. 3-27). London: Chapman and Hall Computing.

Woolf, B. & Cunningham, P. A. (1987). Multiple knowledge sources in intelligent tutoring systems. *IEEE Expert*, 2, 2, 41 - 54.

Woolf, B. & McDonald, D. D. (1985). Building a computer tutor: Design issues. *AEDS Monitor*, 23, 10 - 18.

Zhang, Y., Evens, M., Michael, J. A. & Rovick, A. A. (1987). Knowledge compiler for an expert physiology tutor. In Proceedings of the ESD/SMI Conference on Expert Systems (pp. 153-169). Dearborn.

Zhang, Y., Evens, M., Michael, J. A. & Rovick, A. A. (1990). Extending a knowledge base to support explanations. In *Proceedings of Third Annual IEEE Symposium on Computer-Based Medical Systems* (pp. 259-266). Chapel Hill, NC: IEEE Press.

Zhang, Y. (1991). *Knowledge-based discourse generation for an intelligent tutoring system*. Ph.D. Thesis, Computer Science Department, Illinois Institute of Technology, Chicago, Illinois.