Reactive Instructional Planning to Support Interacting Teaching Strategies

JULITA VASSILEVA
Federal Armed Forces University - Munich
85577 Neubiberg, Germany
E-mail: jiv@informatik.unibw-muenchen.de

Abstract
We propose an architecture for reactive planning of contents in instruction. It is based on a framework for reactive planning which integrates opportunistic reactions with plan-based (plan-repairs and complete replanning). It is suggested how this framework can implement two radically different teaching styles, tutoring and coaching, as well as their interaction within one system. One additional advantage is the possibility to manage the way of system's reacting by means of different pedagogical rules. At this point the system has been implemented and experimented in the domain of integration of elementary functions.

1 INTRODUCTION

There have been many advances in the field of student modelling for Intelligent Tutoring Systems (ITS). However, there has been no equivalent development in the study of teaching strategies for ITS that could make use of elaborated student models. Most ITSs have been designed around an opportunistic paradigm where instructional interactions have been directly triggered by diagnostic actions. In fact, most of the currently existing ITSs are coaching systems (Self, 1994) which have little control over how knowledge gets presented to the student.

Teaching, as most of the human activities is aimed at achieving certain goals and is based on plans (Miller et. al, 1960). There have been numerous approaches implementing more or less generic pedagogical strategies. Most of them have been focused mainly on how to best present an already selected contents. For example, how to sequence explanations, tests, exercises, exploration (Van Marcke, 1992), how to manage the initiative in a tutorial dialogue (Woolf & McDonald 1984), (Woolf & Murray, 1987), (Major, 1993). This is called "delivery planning" by Wasson (1990). In contrast, she calls "content planning" the process of selecting the content for an instructional goal that places the student on an appropriate learning path, like in (Peachey & McCalla, 1986), (Murray 1989, 1990), and her own work (Wasson, 1990). Content and delivery planning are parts of instructional planning - the process of mapping out a global sequence of instructional goals and actions that provides consistency, coherence and continuity in the teaching process.

Most of the approaches to delivery planning have been concerned with representing some discourse strategies used by human tutors. They have been computationally represented with ATNs (Woolf & McDonald, 1984), procedural networks (Spensley et al., 1990), parametrized template networks (Woolf & Murray, 1987).

A pioneering approach to using "classical" (Wilkins, 1988) planning techniques in ITS was developed by Peachey & McCalla (1986) and Wasson (1990). They propose planning of content and delivery based on explicit representation of the target knowledge concept structure (curriculum) to be performed at different levels of granularity. The planning paradigm exploited is of least-commitment non-linear hierarchical planning (Chapman, 1987). In the context of planning instruction this means that the plan does not consist of a particular sequence of goals to be achieved, but rather of a set of plan fragments at a given level of granularity. The order of executing these plan fragments is decided at execution time. In this way unexpected instructional opportunities, like blocked learning paths, missing prerequisite knowledge or serendipitous gains in knowledge can be taken into account.

We propose a framework and architecture for reactive planning, integrating the concept of planning ahead with that of reaction to the environment. This framework differs from Wasson's planning paradigm in that an a-priori ordered plan is created. This allows a global evaluation of the plan and an informed selection

[1] Published in Proceedings AI-ED'95, 7-th World Conference on AI and Education, Washington, August 16-19 1995. AACE: Charlottesville, VA., 334-342.

according to optimality criteria. During the execution of this plan, the system tries to keep to the plan as long as possible, but it is able to react adequately to unforeseen situations at plan time. This is done by not only modifying the plan (avoiding blocked learning paths and making use of shortcuts) as in Wasson's classical planning approach, but also by reacting locally to arising situations and by opportunistically-triggered replanning. The ability to react without a plan is a highly desirable feature in the uncertain and dynamic instructional environments.

An architecture implementing the proposed reactive instructional planning framework has been developed. It is suggested that reactive planning can be used as an instrument for implementing interacting teaching strategies.

2 INSTRUCTIONAL PLANNING IN ITS

2.1 Content Planning: Existing Approaches

Planning is a problem solving technique that creates a sequence of actions (i.e. a plan) to achieve a goal and attempts to forecast the effects of executing the plan (Wasson, 1990). The classical planning problem assumes a state-based definition of the application domain by means of set of primitive actions and their preconditions and effects characterized as state predicates. A planning problem consists of this domain description plus an initial and final (goal) state. The main assumption in classical planning is that the domain description doesn't change while the planning is being carried out. This important limitation leads to a distinction between plan time and execution time.

The first approach to applying planning techniques in ITS (Peachey & McCalla, 1986) uses a domain knowledge base consisting of concepts corresponding to units of subject material which could be taught to the student. Its development (Wasson, 1990) results in a classical planner which is

- hierarchical (on different levels of concept generality),
- non-linear (the plan is a non-ordered set of goals),
- least-commitment (incremental creation of plan fragments, no complete plan).

The first feature reflects the possibility to use the hierarchical abstractions of domain concepts in order to organise planning better. This is a crucial feature for every planner that works in a complex domain.

The last two features provide a high run-time flexibility in adapting to the individual learning path of the student. Recent studies (Minton, Bresina & Drummond, 1994), however, conclude that the only significant difference between partial order and total order planning is *planning* efficiency (i.e. partially ordered planners save resources from creating global plans that will be never executed and from searching of all possible plan orderings). However, if the planning space is hierarchically organized, the eventual planning inefficiency of a total order planner is not a big problem, since it can be compensated by planning in smaller spaces. *Execution* flexibility can also be achieved with a total order planner and a post-processing step that removes unnecessary orderings from the total ordered plan (Veloso, Perez & Carbonell, 1990).

From a pedagogical viewpoint partial order, least-commitment planning of instruction is not necessarily optimal because of the following reasons. First, the dynamic process of incremental generation of local plans is hard to imagine and understand for a human. It is not possible to evaluate such a plan in advance. Second, the ordering of sibling-subgoals in the plan might be very important when there are goal-dependencies. This is often the case in teaching. Learning one concept can make it easier or harder for the student to understand another concept even thought there is no explicit strong relation of precedence between the two concepts. It would be an advantage, if the planner allowed informed ordering of goals that could be assigned by use of additional pedagogical knowledge source, or in an interactive regime during planning by the teacher or by the student himself. Even though we advocate a more rigorous way of planning, by generating an optimal plan and ordering the subgoals in advance and trying to keep to it as long as possible, we have to take into account the dynamic nature of teaching. We do this by accepting a reactive planning paradigm and providing the possibility for dynamic re-planning.

2.2 Reactive Planning

A system is called reactive, if it can react in an acceptable amount of time to any changes that occur in the world while the system is running (Wilkins, 1988). A reactive planning system can react to events which have not been foreseen at planning stage for different reasons (e.g. because they were not known or because it would have been too expensive to consider them at planning stage). Reactive planning is increasingly becoming an active area in AI research. There are a number of suggestions (Spector, Hendler, 1990), (Lyons, et. al, 1991), but no definitive answers yet, on an appropriate way to view long-term planning so as to integrate it with reaction. The area of ITS can be used as a domain for developing and testing ideas, architectures and applications because it has all the features of a domain, where reactive planning is necessary : 1) the agent (the

ITS) can never be certain of the effects of its actions (whether the student really possesses the knowledge suggested by the student model); 2) the agent cannot make the assumption that the world remains static and unchanging while carrying out the plan; 3) the agent cannot assume that it knows everything about the world.

Thus, implementing a reactive planning framework in an instructional planner of an ITS can bring valuable experience and results for both the field of planning in AI and for the field of ITS.

3 ARCHITECTURE FOR REACTIVE CONTENT PLANNING: TOBIE.

In order to implement realistic instructional planning, we need to have the possibility to represent and teach a larger diapason of knowledge of a given domain, e.g. curricular, conceptual and problem solving knowledge. TOBIE (Teaching Operators-Based Instructional Environment) provides an architecture for representing different levels of knowledge organisation at the same time and also a uniform and modular language for domain knowledge representation (see Figure 1).

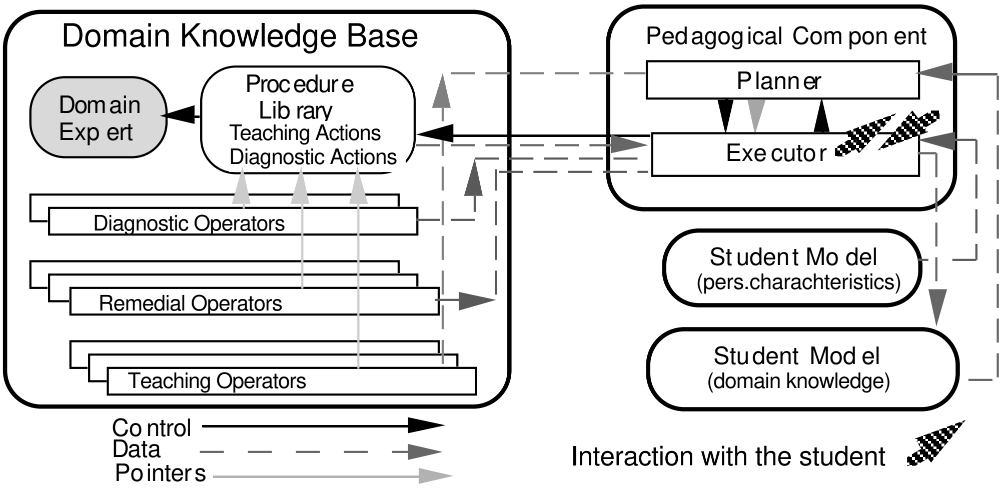

Figure 1. TOBIE Architecture

TOBIE (Vassileva, 1990, 1991) is an ITS-shell architecture based on content planning. It consists of Pedagogical Component (Planner and Executor), a Student Model, and a Domain Knowledge Base. The domain knowledge base contains the elementary objects of the domain corresponding to units of subject material that could be taught to the student. It can be considered as a directed AND/OR graph. Directed AND/OR graphs provide a mighty representation language in which curricular (concept) structures, goal (task) decompositions and problem-solving spaces can be expressed. For example, Figure 2 shows how AND/OR graphs represent domain knowledge on three different levels (curricular, problem-solving and performing single steps). AND/OR graphs can be represented implicitly by sets of production rules, encoded in TOBIE by means of Teaching Operators (TOs).

The TOs are STRIPS-like operators (Fikes & Nilsson, 1971) which consist of 6 parts (see Table 1): a name, a list of preconditions (concepts in the student model) under which the operator can be used, a list of expected effects (list of logical expressions to be added to the student model), an action which is a pointer to a teaching procedure in the Library of Procedures. These procedure present to the student teaching material in a specific way, e.g. present, explain, focus, remind, solve a problem, provide an exercise or test. The teaching procedure can also contain a recursive call of the system on a different level of organisation of domain knowledge, for example to show how to solve step by step a given type of problems (as mentioned before, in TOBIE it is possible to represent different levels of knowledge organisation). The TOs contain also a part called "Diagnosis". It contains a pointer to a Diagnostic Action — a procedure stored in the library of Teaching Actions which evaluates the success of the student and eventually analyses the student's answer and finds misconceptions. It adds to the student model either the effect-node(s) of the TO or the node corresponding to the diagnosed misconception. The teaching actions which request solving a problem of a give type are associated with the same diagnostic procedure. The last part of a TO is called "Type". It contains a list of parameters which describe the teaching action from a pedagogical point (e.g. if it is an explanation, example, exercise), the difficulty (if it is an exercise or test), and the type of media (text, graphics, picture with motion or sound).

Table 1. An example of a TO and a teaching action procedure (see the concept structure in Figure 2)

NAME	CONDITIONS	EFFECTS	ACTION	DIAGNOSIS	TYPE
teach_partfra	rat_t5, tr_partfra	part_fra	teach(partfra)	test(partfra)	linear

PROCEDURES LIBRARY:

teach (name):

pres_name, /* presents a text explaining how to solve integrals by method "name"

examp_name, /* presents an example of a solved problem with method "name"

demo(name), /* calls recursively planning in the problem-solving space with background goal "name" and default teaching action "demo" (tutoring strategy).

exerc(name) /* calls recursively planning in the problem-solving space with background goal "name" and default teaching action "exercise" (coaching strategy)

test(name): /* calls recursively planning in the problem-solving space with background goal "name" and default teaching action "test" (like coaching but no feedback).

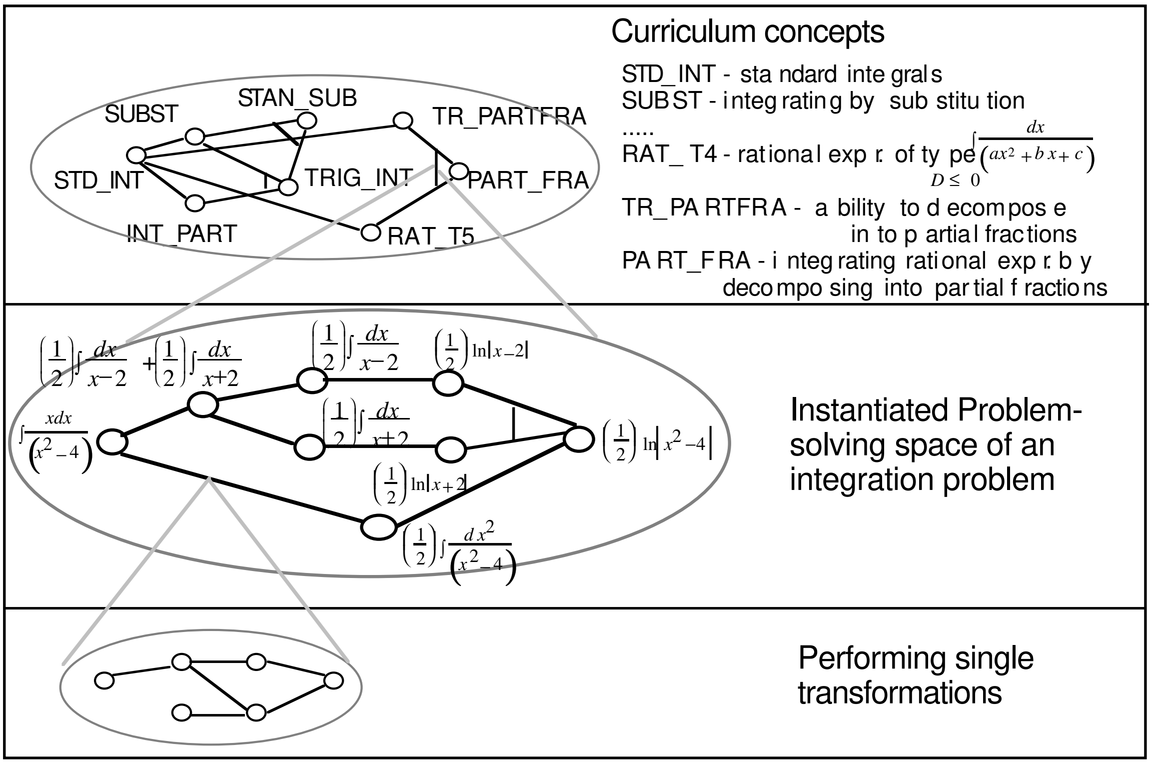

Figure 2. Representing different levels of knowledge organization

The model of the student's domain knowledge represents the current state of the student's knowledge in terms of the elementary objects that are believed to be learned (correct concepts and misconceptions). It is updated in two ways which are not discussed here, since it is of no direct link with the goals of this paper. The model of the student's personal characteristics represents certain preferences of the student to different types of teaching actions, psychological and motivational parameters, like field-dependence, concentration, confidence, persistence.

The Pedagogical Component contains two sub components - a Planner which dynamically generates plans in the knowledge structure to meet certain teaching or goals and an Executor which carries out the plan, re-invokes the Planner or reacts locally to arising opportunities. In the next section we shall discuss the Pedagogical Component and the way reactive planning is of carried out.

4 REACTIVE PLANNING IN TOBIE

The structure of the Pedagogical Component is shown in Figure 3.

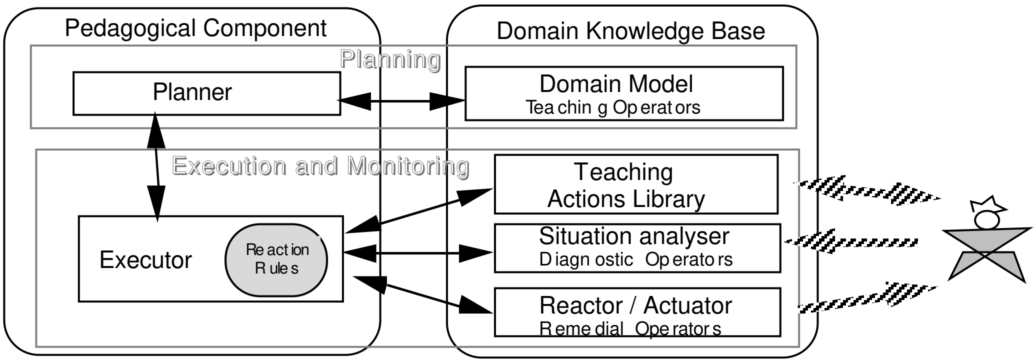

Figure 3. Reactive Planning in TOBIE

4.1 Planning

The Planner is activated within the domain concept structure at one given level of organization. The planning algorithm is a modification of the AO* (Nilsson, 1980). The optimisation function *h* can be selected so that different criteria for optimality can be implemented (e.g. the shortest, the plan avoiding certain concept, plan with a certain topology-type etc.). The solution graph of an AND/OR graph imposes only a partial ordering on the solution steps. If there are no subgoal interactions, the order of applying the operators is not important. In principle, goal dependencies can:

1) make the further execution of a plan impossible since there are TOs which influence negatively on the student's knowledge, i.e. which could delete concepts from the Student Model, or

2) make the plan no longer optimal because of unexpected acquiring of goals.

The first type of goal interaction is dangerous. In our architecture, however, only the Remedial Operators can delete concepts (corresponding to misconceptions) from the student model and they are not considered at planning stage. Misconceptions and their remediation are never planned in our system, but treated opportunistically. However, assuming that there is no explicit goal interaction, the plan can be partially reordered interactively or automatically according to certain pedagogical criteria (for example, more concrete concepts before abstract ones, simple concepts before more difficult ones etc.) and considering the pedagogical type of the TOs. If it happens that the student acquires unexpectedly concepts, no plan-shortcut is made. Unlike Wasson's planner our system doesn't take advantage of learning opportunities as they arise on the flow. A shortcut will be made only in case that the plan can't be continued and re-planning is needed.

4.2 Situations

After an initial plan is generated, it is passed to the Executor. It executes the TOs by invoking the procedure assigned by the "teaching action" part of the operator from the Procedure Library (see Table 1). This procedure can consist of other procedures which present explanation of the concept, give examples, start an exercise. The main teaching action prcedure of a TO contains always a procedure which tests whether the student has acquired knowledge on the concept. The diagnosed knowledge on the concept or misconception is included in the student model which provides a condition for the next Teaching or Remedial Operator to be applied.

In case that a misconception has been diagnosed and entered the student model, or an undiagnosable error has been made (or a call for help from the student), a not planned situation arises. Another type of unexpected situation can occur when the teaching action of a TO invokes instructional planning at another level of knowledge organization. This is not considered at planning time, since the actions of the TOs are stored separately in a library and TOs are selected only because of their conditions, effects and pedagogical type. In principle the calls to another level could be considered at planning, but this would make the planning much more complex and very often the resulting plan will not be feasible because of external factors, like time. Since switching to a different level means instruction in a comparatively independent part of the material, it won't be pedagogically justified to interrupt the process in the middle because of external factors, though sometimes it would be necessary to do so. That is why the decision whether to permit a switch and how to do it has to be taken at plan-execution time, when the need arises (and not long in advance).

Another situation can arise from a combination of external factors which are not dependant on the plan execution. For example, a violation of time-restriction, evidence that the student is no longer concentrated, opportunity to fulfil a teaching goal that has been staying in the background (for example, on a different level of organization of material), evidence from history that the student has had difficulties with a concept before etc. This type of situation is recognised by the so-called "Diagnostic operators". They are rules encoding combinations factors (variables) with different values describing the current context. We distinguish among five types of factors: parameters of the environment (time, resources); history (how long did it take to study concepts with similar difficulty, did he ever learn the concept, did he ever show success on problems involving knowledge on this concept etc.); background teaching goals; the model of the student's domain knowledge and the model of the student's personal characteristics.

A search for matching diagnostic operators is done at every tact of executing the plan, i.e. after the execution of a elementary procedure included in the action-procedure of a TO (see Table 1.). The various situations matched by the diagnostic operators can't be treated in an equal way. That is why every diagnostic operator assigns also a specific reaction.

4.3 Reactions

Our system provides four principle type of reactions: ignoring the situation, an opportunistic reaction without changing the plan; local plan repair; global re-planning. They are shown in Figure 4.

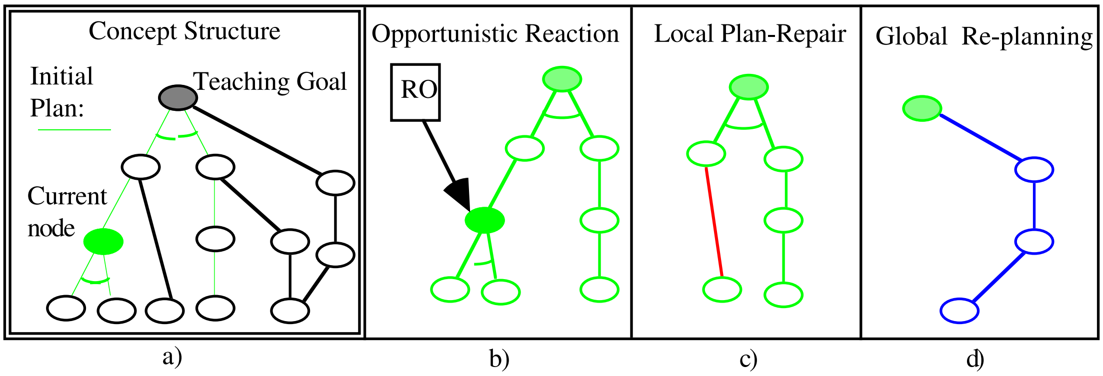

Figure 4: Reactions.

Figure 4, a) shows an AND/OR graph, representing a domain structure at some level of organisation, for example, any of the levels shown in Figure 2. The initial plan for achieving the teaching goal and the current node (concept or problem solving state) where an unexpected situation has arisen are presented. The first type of reaction (ignoring the situation) allows the system to follow a plan rigidly. An opportunistic local reaction (Figure 4, b) provides an immediate feedback while keeping the initial plan. A Remedial Operator (RO) will be executed when the student model contains the misconception matching the operator's preconditions. Special ROs are provided for unidentified errors (general hint, humorous remark, encouragement, etc.). Local Plan Repair (Figure 4, c) means that only the part of the plan related to the current node will be changed. In this way the system tries to find an alternative way to teach a difficult concept without changing the overall plan. A global replanning means finding an alternative plan for the main teaching goal (see Figure 4, d).

A summary of the possible reactions to the four situations is given in Table 2.

4.4 Matching The Situations With Reactions

A set of rules is responsible to select a reaction to situations that occur during plan-execution. These rules we call "reaction rules" or "pedagogical rules", since we believe this is more a pedagogical decision. The conditions of these rules are based on the same groups of factors that are matched by the diagnostic operators, but they have one additional factor - the type of situation. That is why we provide a tool for creating pedagogical rules and diagnostic operators that define new possible situations as combinations of factors. Four approaches are possible:

- to define at hoc the reactions to the possible situations (the current solution);
- to interview teachers with the goal to extract rules and implement them using the rule-editing tool or to ask them to implement the rules directly themselves (however, this requires that the teachers are able to articulate the factors influencing their decisions, which is not often the case);
- to observe what human teachers do in real situations and try to extract some knowledge out of their behavior, i.e. analysing protocols of teaching sessions, and applying machine learning techniques in order to define cases and, eventually, to generate decision trees;
- to try to define some guidelines from existing didactic theories, or to try to model within this framework the teaching strategies of existing ITS, like Wasson (1990), COCA (Major, 1993).

Table 2. Situations and Reactions

Reaction **Situation**	Local Opport. Reaction	Local Plan- Repair	Global Replanning
unidentified error, misconceotion	execute remedial operator	make a plan for the current parent goal and replace the corres-ponding part of the old plan	start the planner anew
combination of factors	execute remedial operator		
call to another level	keep old plan in stack and revise it after returning	revise old plan from stack using "shortcuts" from obtained goals at the other level	after coming back, re-plan to make use of the changed environment

5 IMPLEMENTING TWO TEACHING STYLES AND THEIR COMBINATIONS

The TOBIE architecture and the reactive planning framework described above allow implementing two radically opposite teaching styles and their interaction in one system.

Tutoring is a teaching style aimed at communicating new material, presentation-oriented, straightforward, the initiative is in the tutor, the student is guided and prompted to reply, solve a problem, etc.

Coaching is a teaching style aimed at developing the skills of applying existing knowledge to new situations, to re-organize knowledge structures and develop meta-cognitive skills. The initiative is in the hands of the student, the coach can comment and interfere to give advice, or provide help when requested or when this is necessary to overcome misconceptions and difficulties.

Both strategies can be modelled as a planning process. The only difference is who takes the active role. In the first case, it is the system who creates a plan for teaching a given concept and leads the student step by step to achieving the goal. In all previous sections we have been focusing implicitly on a tutoring style, that is why now we shall only concentrate on the implementation of a coaching style and how one system can interactively use both styles.

A typical case when coaching style is used is to support the student in exercises for solving problems (represented as a problem state space, a goal and a initial state). The system can generate a solution by creating a plan which leads from the initial to the goal state (if several plans are possible, it selects an optimal one), but instead of executing it, it only observes the student's actions and tries to match them with the system's plan. In case that the student makes the same steps and goes along the system's plan, no reaction from the system is needed. In case of a difference, however, a situation occurs which requires a reaction. The possible reactions are defined in exactly the same way as discussed in the previous section. They are: ignoring the difference (keeping silent); local reaction aimed at bringing the student back on the system's plan (like in the model-tracing style of the Lisp-tutor); local plan-repair (trying to find a way to accommodate the student's solution within the system's plan) and complete replanning (trying to find another plan for solving the problem that fits with the student's solution).

Other situations that need reaction can be defined in analogy with those described in the previous section: a combination of factors (e.g. environment, the history, the student model, the structure of the problem solving space, opportunities to discuss background teaching goals, personal characteristics of the student). For example, if the time is nearly over, and the plan which the student is following is too long or involves complicated (expensive) steps, this creates an opportunity to interfere.

The pedagogical rules for selecting a reaction for the specific situation involve the factors mentioned above and depend on the type of situation. For example, if there is enough time and the student is confident, to choose a re-planning reaction to accommodate his way of solving instead of bringing him back with a remedial operator to the corresponding state in system's plan.

Here we have to state that there is no difference in the planning and executing mechanisms used to implement both teaching styles. The difference in the interaction and initiative is completely due to the different teaching actions, the procedures that carry out the dialogue with the student and the presentation of material.

A combination of the two strategies will be shown with example. Let's suppose that the system is teaching in the domain of symbolic integration and has planned instruction according to the curriculum from Figure 2. The current goal is to teach the concept (method in our case) PART_FRA - "integration by decomposing into partial fractions". The action of the TO for this goal is a sequence of five procedures (see Table 1), the first two presenting a textual explanation and example, the third one demonstrating a step-by-step solution of an example problem, the fourth one - coaching the student's solving of another problem and the fifth one - testing how the student copes with a problem alone. The demo-procedure invokes planning on a different level of knowledge organization, where the TOs represent admissible transformations between different types of expressions (states) and heuristics for selection of transformations for different states (Slagle, 1963). A problem solving space is generated (instanciated for the example problem) by executing TOs which perform transformations over the instanciated integrand-types (the problem-states). There are two main types of teaching-action procedures at this level, which correspond to the two teaching styles. A procedure for tutoring style ("demo") performs the transformation encoded with the TO and displays the result. By executing the actions of the TOs from the plan, the solution of the problem is shown step by step.

The exercise("exerc") procedure invokes the planner again on the problem-solving level for a different problem. This time a coaching style procedure is activated. Such a procedure visualises the initial state, asks the student to select a transformation (a TO), performs it (only if it is applicable; if not - a situation arises) and builds increasingly another problem-solving space the student model.

Let's suppose that the student has to solve the problem: $\int \frac{x}{x^2-4}dx$. The student chooses to apply a standard transformation $xdx \Rightarrow \left(\frac{1}{2}\right)dx^2$, resulting in $\left(\frac{1}{2}\right)\int \frac{dx^2}{x^2-4}$ and then selects a substitution to transform it to $\left(\frac{1}{2}\right)\ln\left|x^2-4\right|$ which is the right solution. However, the plan of the system was to teach him solve with the method of partial fractions (background goal inherited from the upper level), so the system's solution would be to transform the initial integrand into $\frac{A}{x-2}+\frac{B}{x+2}$, then to find out the coefficients A, B by solving a system of linear equations and finally to obtain $\left(\frac{1}{2}\right)\int \frac{dx}{x-2}+\left(\frac{1}{2}\right)\int \frac{dx}{x+2}$ which transforms into a sum of standard integrals $\left(\frac{1}{2}\right)\ln|x-2|+\left(\frac{1}{2}\right)\ln|x+2|$, equivalent to $\left(\frac{1}{2}\right)\ln\left|x^2-4\right|$. Because of the background teaching goal, a reaction will be given at the situation which arises at the first step (where the student's solution differs from the system's). The reaction will be a remedial action to bring him to the system's plan. If there is not a specific background goal or if the pedagogical rules do not assign so high priority to background goals, the system would let the student proceed in his own way. As mentioned before, the pedagogical rules can define a completely different behavior of the system.

6 CONCLUSIONS AND FURTHER WORK

In this paper we propose an architecture for reactive planning of contents in instruction. In contrast with classical non-linear, hierarchical, least commitment planning, our approach allows global evaluation of the plan and selection of optimal one, coping with interactive goals and hierarchical planning on different levels of organising the material. The architecture is based on a framework for reactive planning integrating opportunistic reactions with plan-based (plan-repairs and complete replanning). It is suggested how this framework can implement two radically different teaching styles, teaching and coaching, as well as their interaction within one system. One additional advantage is the possibility to manage the way of the system's reacting by means of different pedagogical rules. At this point the system has been implemented in the domain of integration of elementary functions at three levels or domain knowledge organisation: curricular - methods for integration, solving integration problems and performing single transformations. A tool for editing diagnostic operators and pedagogical rules has been developed and three rule-sets have been created ad-hoc. Our current work is aimed at identifying such rules by interviewing teachers. A tool for induction of pedagogical rules from protocols is being implemented now, applying machine learning techniques. We intend to compare the rules derived from protocols with those which follow from some general frameworks and theories known in didactics.

REFERENCES

Chapman, D. (1987) Planning for Conjunctive Goals, *Artificial Intelligence* 32, 333-377.

Fikes, R., Nilsson, N. (1971) STRIPS: a new approach to the application of theorem proving to problem solving.. *Artificial Intelligence*, 2(3/4), 189-208.

Lyons, D., Hendriks, A., Mehta, S. (1991) Achieving Robustness by Casting Planning as Adaptation of a Reactive System, *Proceedings of IEEE International Conference on Robotics and Automation*, April, 1991, IEEE, New York.

Major, N. (1993) Reconstructing Teaching Strategies with COCA, *Proceedings of AI-ED'93,* 66-73.

Miller, G., Galanter, E., Pribram, K. (1960) *Plans and the Structure of Behavior*. Holt, Reinhart and Winston.

Minton, S., Bresina J., Drummond, M. (1994) Total Order and Partial Order Planning: A Comparative Analysis, *Journal of Artificial Intelligence Research*, 2, 227-262.

Murray, W. (1989) Control for intelligent tutoring systems: a blackboard based dynamic instructional planner. *Proceedings of the 4th International Conference on AI and Education*, Amsterdam, 150-168.

Murray, W. (1990) A blackboard based dynamic instructional planner. *Proceedings of the 8th National Conference of AI,* Boston, MA, 434-441.

Nilsson, N. (1980) *Principles of Artificial Intelligence*, Tioga Publ.: Palo Alto.

Peachey, D., McCalla, G. (1986) Using Planning Techniques in Intelligent Tutoring Systems, *Int. J. Man-Machine Stud.*, 24, 77-98.

Self, J. (1994) The Role of Student Models in Learning Environments, *IEICE Trans. Inf. & Syst.*, E77-D, no.1. 3-8.

Slagle, J. (1963) A heuristic Program that Solves Symbolic Integration Problems in Freshman's Calculus, *Journal of the ACM*, 10, 507-520.

Spector, L., Hendler, J. (1990) An Abstraction -Partitioned Model for Reactive Planning" in Y. Wilks and P. McKevitt (eds.) *Proceedings of the 4-th Rocky Mountain Conference on AI,* Computing Research Laboratory, New Mexico State University, Las Cruces, N.M., June 1990.

Spensley, F., Elsom-Cook, M., Byerley, P., Brooks, P., Federici, M., & Scaroni, C. (1990). Using multiple teaching strategies in an ITS. In C. Frasson & G. Gauthier (Eds.) *Intelligent Tutoring Systems: At the crossroads of Artificial Intelligence and Education.* Norwood, N.J.: Ablex.

Van Marcke, K. (1992) Instructional Expertise. in C. Frasson, G. Gauthier & G.I. McCalla (Eds.) *Intellignet Tutoring Syst*, Lecture Notes in Computer Science No.608: Berlin-Heidelberg.

Vassileva, J. (1990) An Architecture and Methodology for Creating a Domain Independent Plan-Based ITS. *Education & Training Technologies Internaional,* 27 (4), 386-379.

Vassileva, J. Radev, R. Dimchev, B., Madjarova, J. (1991) TOBIE: An Experimental ICAI-Software in Mathematics. *Proceedings CALISCE'91*, Lausanne, 145-150.

Veloso, M., Perez, M. & Carbonell, J. (1990) Nonlinear planning with parallel resource allocation. In *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling and Control.*

Wasson, B. (1990) *Determining the Focus of Instruction: Content Planning for Intelligent Tutoring Systems*, Doctoral Thesis, Department of Computational Science, University of Saskatchewan.

Wilkins, D. (1988) *Practical Planning: Extending the Classical AI Planning Paradigm*, Morgan-Kaufmann: San Mateo.

Woolf, B., McDonald (1984) Building a computer tutor: Design Issues. *IEEE Computer,* 17(9), 61-73.

Woolf, B., Murray, T. (1987) A framework for representing tutorial discourse. *Proc. 9th IJCAI*, Los Altos, CA.

Acknowledgements:

I am thankful to Bojko Dimchev for the original implementation of TOBIE (now five years ago), to several graduate students at the institute of the Technical Computer Science in Munich who implemented parts of the system; to Ulrich Hoppe and to the anonymous reviewers for commenting on earlier drafts. This work has been supported by project I-406 of the Bulgarian Ministry of Science and Higher Education.