DCG + GTE:
Dynamic Courseware Generation with Teaching Expertise

Julita Vassileva
Universität der Bundeswehr München
85577 Neubiberg, Germany
E-mail: jiv@informatik.unibw-muenchen.de

Abstract
This paper discusses the place of GTE as an approach for bridging the gap between CAL and ITS systems. It presents the DCG, our architecture for dynamic courseware generation which allows dynamic planning of the contents of an instructional course with a given goal. Finally a further development of DCG by combining it with GTE is described which includes explicit representation of generic teaching knowledge. This allows dynamic planning of how a selected contents will be presented to the student.

1 Introduction

One possible approach for „intellectualising CAL" is to start from a set of teaching primitives defined at different levels of granularity to manage the flow of instruction in a flexible way. Schemes for controlling the dialogue and presentation of teaching materials have been introduced borrowing formalisms for representing natural language dialogues, for example augmented transition networks, like (Woolf, 1987) and (Murray, 1992). An original approach called GTE for bridging the gap between CAL and ITS (see Figure 1) has been proposed by Van Marcke (1991). It does not use standard AI techniques to intellectualise CAI, but rather takes a task-based perspective by defining instructional task-hierarchies and modelling instruction as performing a sequence of tasks. Though task analysis is widely used for interface design, we are not aware of other approaches employing task-analysis in the design of instructional systems. In the tradition of other CAL → ITS approaches, GTE provides a method for sequencing teaching materials, i.e. it focuses on the problem „*How* to teach?" rather than „*What* to teach?". Though GTE has a „content model" consisting of topics, this model is a static skeleton around which presentation is built. A full traversal of the topic structure takes place rather than a dynamic decision about which topic to present, considering alternative trajectories through the knowledge structure. That is why we classify the GTE as a „delivery planning" approach (Wasson, 1990) in contrast with „content planning" which is typical for approaches approaching the gap from the other direction: ITS → CAL.

Several approaches take the latter direction - applying ITS-shell architectures to achieve flexible and automatic generation of courseware (Elsom-Cook & O'Malley, 1989), (Wentland et. al, 1991), (Brussilovsky, 1992). Our approach for **Dynamic Courseware Generation** (DCG) (Vassileva, 1992) falls into this stream. It is based on an ITS-shell architecture (Vassileva, 1990), whose main idea is global planning of the *content* of instruction (Peachey & McCalla, 1986), (Wasson, 1990). Based on a separate explicit representation of the domain structure (separated from a library of teaching materials), the system dynamically plans the contents of instructional courses. The course-plan is created individually for a given student with a given teaching goal; then it is substantiated with teaching materials and can be changed dynamically according to the changing learning needs of the student. The DCG can be seen as a complementary approach to GTE: it focuses only on the

question of how to plan the contents of a course for a given goal and how to dynamically modify this content plan to adapt it to the progress of the individual student.

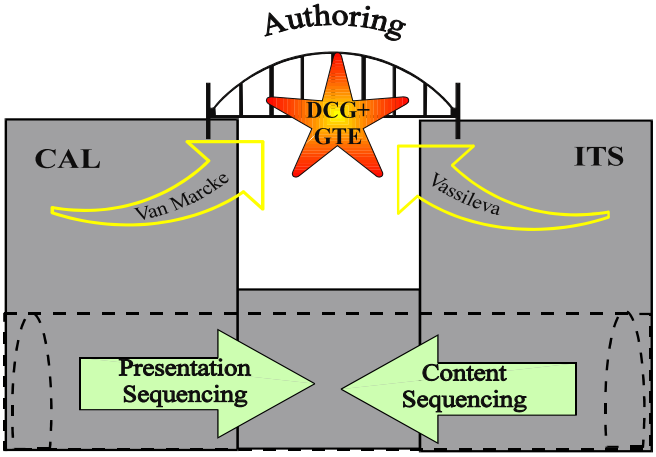

Figure 1: Bridging the gap between CAL and ITS.

In section 2 we shall present the main idea, architecture and functioning of the DCG. In section 3 we shall describe how the DCG can be combined with GTE to provide dynamic planning of both content and delivery in instruction.

2 Dynamic Courseware Generation

The basic idea of DCG is to use a classical mechanism for planning in an AND/OR-graph representation of the domain concepts for automatic generation of a content plan of a course (see Figure 2).

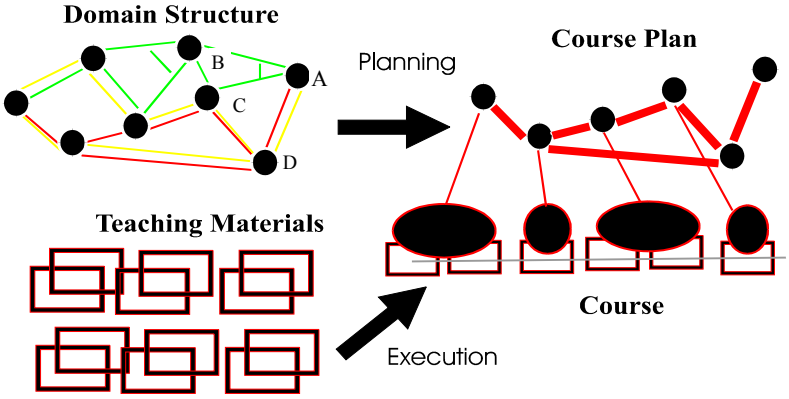

Figure 2: Content planning in the Domain Structure.

The architecture of the system consists of the following components (see Figure 3).

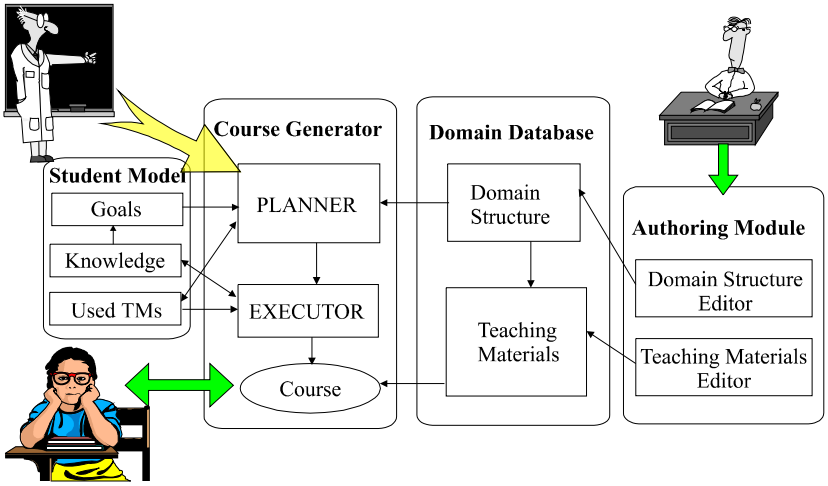

Figure 3: Architecture of DCG

2.1 The Data Base

The subject-dependent knowledge is contained in the Database Component. It contains two parts:

• The **Domain Structure** contains the structure of the subject knowledge that is going to be taught. It is represented as an AND/OR-graph with nodes, corresponding to the concepts (similar in their role to the topics in GTE) and links, corresponding to the semantic relationships between them, e.g., aggregation, generalisation, analogy, implication. If learning a given concept A requires that the student knows two or more other concepts B and C, the node A is linked with these two concepts as a AND-node (see Figure 2). If there are several alternative prerequisite concepts (node D or alternatively both B and C in Figure 1) that are needed to learn A, it is linked with these concepts as an OR-node. Unlike the content model of GTE which is, in fact, a static curriculum which must be learned exhaustively by the student, the Domain Structure in DCG encodes knowledge about the domain. The concepts in DCG have smaller grain size than GTE's topics and the links among them encode different semantic relationships, rather than only prerequisite links. For example, a node can represent a device. The components of this device are represented as other nodes connected with an „aggregation"-link to the device-node, so that the device-node is an AND-node with respect to this link. One domain can be represented with several interconnected AND/OR graphs corresponding to different *aspects*. For example, a technical device can be described with one AND/OR graph (or aspect) representing its structure, another one - representing its functioning and a third one representing its geometry. The Domain Structure (which is a set of AND/OR graphs representing various aspects) can be traversed in various different ways by planning along links with a given semantic and taking specific aspects as „main". In this way completely different content plans can be generated for achieving the same goal from different perspectives (viewpoints). Every node from the Domain Structure has an associated set of Teaching Materials (TMs) with different pedagogical type and media. During the course execution TMs are selected by different teaching tasks to teach the concepts of the plan.

• The **Teaching Materials** (TMs) part of the Data Base contains presentation- and testing-materials that carry out the communication with the student. They are focused on a given concept or relationship. TMs can

have different pedagogical characteristics. For example, one can distinguish among an introduction to a concept, a motivating problem, an explanation, help, exercise, or test. In this sense TMs are equivalent to the "instructional objects" (Van Marcke, 1991) in GTE. The TMs are also classified with respect to the media they use, i.e. textual, graphical image, animation or video etc.

2.2 The Student Model

The Student Model consists of three parts:

• the identifiers of the TMs that were already used, with annotation of their type and success;

• an overlay with the Domain Structure in the data base, containing probabilistic evaluations of the beliefs that the student knows and does not know a given concept or link. These probabilities are updated when the student solves a test material (Diessel et. al., 1994).

• a list of the accomplished sub-goals (concepts whose belief probabilities exceed a threshold value assigned by the Teacher).

2.3 The Course Generator

The Course Generator is the component that creates the course, carries out the interaction with the student and maintains the Student Model. The Course Generator contains the following components:

- **Planner.** The planning algorithm is a modification of the AO* (breadth-first heuristic search algorithm, for detailed description see Nilsson, 1980). It generates a content plan which is a sub-graph of the Domain Structure. The input for the Planner is consists of: the semantic of the link according to which planning will be made (i.e. a sub-graph of the domain-structure with respect to specific type of arcs), the current state of the student's knowledge (the concepts present in the student model are taken as leaf-nodes in the graph), the teaching goal (a goal-node representing the concept to be learned). The heuristic function h can be selected so as to achieve different criteria for optimality, e.g. the shortest plan, the plan avoiding certain concept, plan predominantly within a certain aspect etc. This plan is passed to the following component.
- **Executor.** It generates a course based on the plan by selecting and presenting appropriate TMs. In the same time the Executor updates the Student Model taking into account the results of the student on the test-atoms. The Executor invokes remedial TMs (explanation of the reason of the error, associated with the test-atom) for providing immediate feedback to the student's errors. In addition, it keeps track of the changes in the student model and is able to re-invoke the Planner to create a new plan, if the level of knowledge is not satisfactory.

2.4 The Authoring Component

The Authoring Component consists of a TMs-Editor, a Domain Structure Editor and an Editor for Instructional Tasks and Methods.

The **TMs-Editor** is a tool that allows creating teaching materials. There are two types of TMs: presentations and tests. A unique name is given to every TM and it is associated with a concept or link from the Domain Structure. A TM can be associated with more than one concept. To provide means for the system to evaluate the degree of knowledge of each of the concepts involved in a given test, the Author has to define a set

of likelihood vectors: two vectors containing the conditional probabilities of the student's knowledge of each concept involved in the test-atom, at correct and at incorrect answer. At least one test has to be created for every node and link, so that the system can judge from the student's success or failure whether she knows the corresponding concept / link. A set of parameters describing each TM with respect to its pedagogical type, media and time allotment are attached.

The **Domain Structure Editor** is a graphical editor which allows developing, extending and modifying the Domain Structure. It allows to insert, delete and move, name and re-name nodes on the screen; to insert, delete and connect links (i.e. create „AND"-nodes); to represent the different semantics of the links with different colours; to view the existing teaching materials in the data-base and to associate them with the nodes and links from the Domain Structure.

2.5 Implementation

The implementation of the DCG runs in a MS-Windows environment. The system is implemented in C++ and OpenScript. Multimedia ToolBook © Asymetrix is used as an authoring tool for creating the TMs. It allows graphical authoring of multimedia TMs. Prototypes of the system have been implemented and tested in several domains: electric toasters, transistors, theoretical surgery, music and typewriter-training.

In order to evaluate the effort spent for creating an hour of instruction, the time spent for authoring has to be divided by the sum of the duration of all possible courses that can be generated by the system (with all possible teaching goals). We obtained an approximate ratio of 18 hours of authoring for one hour of instruction. This is a favourable result in comparison with other authoring approaches for ITS, since the average time of design and authoring for one hour of intelligent instruction is considered to be at least 100 hours.

2.6 DCG - Summary

Dynamic Courseware Generation stays at the cross point of ITS, CAL and Authoring. Its main advantages are:

- **flexibility in the goals of courses**. The AND/OR graph semantic representation of the domain concepts i.e. the possibility to define and use different types of semantic links among them allows the system to decide how to plan the contents of a course for any given goal concept and viewpoint (perspective).
- **individualisation of instruction**. By continuous monitoring of the student's behaviour and maintaining a simple model of student knowledge, the DCG is able to dynamically tailor the content-plan of the course to the student.
- **possibility for more effective authoring**. This is a result of the system's ability to automatically generate different courses with different goals from the same domain structure and teaching materials.

3 Combining GTE and DCG

The DCG doesn't solve the problem of how to present the selected contents (the current concept or relation) to the student. At execution time it just selects one or more TMs trying to use those whose pedagogical type parameters have proven to be successful with the student so far (the student model). However, these TMs remain discrete, there is no smooth transition between the presentations, no possibility to present materials according to a certain teaching strategy.

We see an excellent opportunity for solving this problem in integrating the DCG with GTE. This will provide the system with knowledge of *how* to teach a given contents (represented by instructional tasks and methods) and will allow planning the presentation of the already selected contents (the current goal concept from the content plan) in a pedagogically meaningful way.

In this combination, the DCG-part decides which concepts will be taught, i.e. it dynamically creates a content plan of the course. The GTE-part provides a representation of instructional tasks and methods, which allows the system to plan dynamically *how to present* the contents related to the current concept in an optimal way for the student, i.e. what types of TMs to select and how to sequence them

The modified DCG + GTE architecture is shown in Figure 4. The changes in the original DCG architecture (Figure 3) and functioning will be discussed in the next sections.

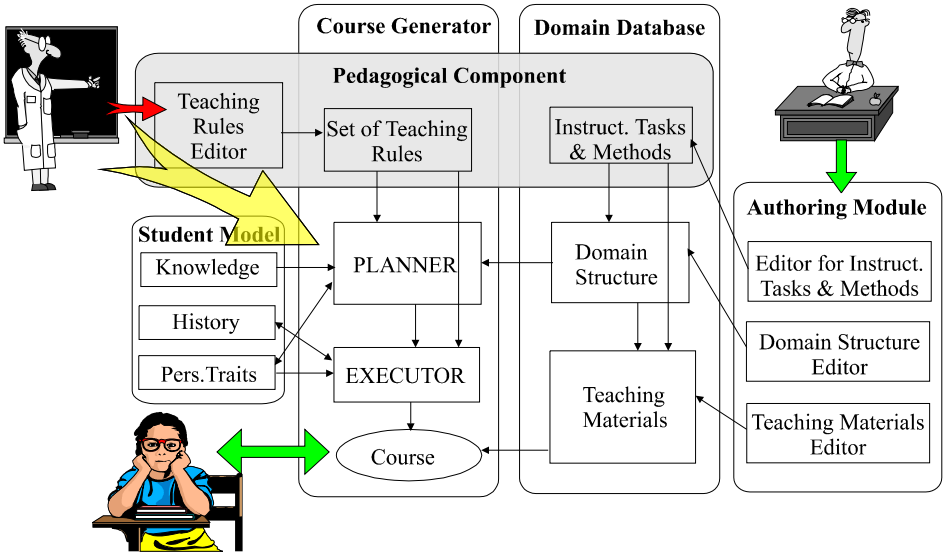

Figure 4: The DCG + GTE architecture.

3.1 Pedagogical Component

A new component is included in the combined architecture containing the generic task representation and teaching strategic (meta-) knowledge This component contains two main parts: a representation of the instructional tasks and methods and a set of teaching rules. Each of these parts has a generic kernel and can be expanded with subject specific knowledge (tasks, methods and rules). In addition an editor for instructional tasks

and methods and for teaching rules is provided for the teacher to extend the teaching expertise of the system according to his preferences.

3.1.1 Instructional Tasks and Methods

This is a part of the Pedagogical Component which contains a representation of instructional tasks and their decomposition into sub-tasks by means of different task-decomposition methods like in GTE (Van Marcke, 1991). These task-structures can be represented with AND/OR-graphs similar to that representing the Domain Structure. For example, Figure 5 represents the generic task "Give exercise" (adapted from Van Marcke, 1991). The sub-task "Remedy" can be decomposed in different ways according to different methods (shown with different types of lines). The tasks and methods can be generic, but as described in (Van Marcke, 1991), the deeper the sub-tasks are in the task-decomposition hierarchy, the more subject-dependent they become. A special editor is provided in the Authoring Module, so that the pre-defined set of generic instructional tasks and methods in the system could be extended with subject-specific ones.

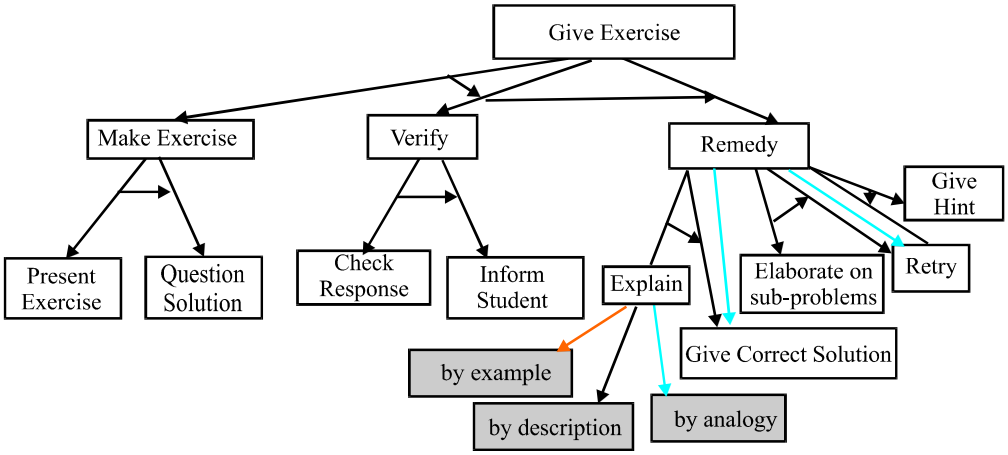

Figure 5: An example of a task hierarchy (adapted from Van Marcke, 1991)

3.1.2 The Set of Teaching Rules

GTE does not allow a convenient representation of meta-strategic rules for selection of the appropriate methods / tasks for the individual student. We included a Set of Teaching Rules in DCG + GTE which represents „meta-knowledge“ and can be adapted according to different teaching strategies. In this way the system’s selection of teaching tasks for carrying out the content plan can be adjusted easily according to a specific teaching theory or to the teacher’s preferences.

The teaching rules encode different discourse and teaching strategies. They were developed after an analysis of didactic literature (Bohnert, 1995). All teaching rules take into account data from the student model (student's knowledge and personal traits), as well as external factors like time. They can be divided into three main groups: *discourse rules* which manage the generation of the content plan, *strategy and teaching method selection rules* which decide who has the initiative and the selection of instructional (task-decomposition) methods, and *teaching material selection rules* which select TMs appropriate for the student. Most of the rules

from the second and third group are generic. The discourse rules however, are usually subject-specific. Below we shall discuss the teaching rules in more detail.

- **Discourse rules**

These rules manage the selection of a content-plan by manipulating the optimisation criterion of the Planner. For example, they can select only plans according to certain types of semantic. One discourse rule, motivated by (Flammer, 1975), for example, states that in case that the student is intelligent, top-down (deductive) presentation of the plan is appropriate with respect to the aggregation links (i.e. from whole to parts), while for less intelligent students - bottom-up (inductive, i.e. from parts to whole). If the concepts about a technical device are organised according to, functional and structural aspects, one discourse rule states that is better for not-knowledgeable students to use the functional aspect for the main plan allowing links to concepts from the structural aspect, while for knowledgeable students it is more appropriate to teach only the structural aspect (Paris, 1993). If re-planning on the concept level is needed, the discourse rules select whether it will be local re-planning or a global change of the plan.

- **Strategy-selection rules**

These rules define how to select the teaching strategy before starting the execution of the plan. The teaching strategy defines the general principles of teaching, for example, who has the initiative in deciding what to do next - the student or the system. In accordance with GTE, we distinguish between two main types of teaching strategies: **"structured"** and **"unstructured"**. A "structured" strategy means that the initiative is in the the system: it selects which concept will be taught next and how (i.e. with which instructional task). An "unstructured" strategy leaves the choice of a next concept to the student. A highlighting of the "ready to be learned concepts" those whose prerequisites in the plan are considered as known by the student, as in (Beaumont & Brusilovsky, 1995) can help her navigate in the Domain Structure. The student can also choose an instructional task and method for the current concept from the graphical representation of the task hierarchies. In the spirit of some general principles for choosing teaching strategies according to different student's aptitudes (Siegler, 1988), we defined several strategy-selection rules. One, for example, states that if the student is motivated and success-driven, an "unstructured" strategy would be appropriate, while if she is unsure and not confident, the "structured" strategy should be preferred.

- **Method-selection rules**

There are usually three to eight alternative task-decomposition methods for each instructional task (Van Marcke, 1991). The method-selection rules take into account the history of the used instructional tasks and TMs and the student's personal traits and preferences in order to decide which main instructional task(s) to select for the current concept and which task-decomposition method to use. This is done right before planning at the instructional task-level. The method-selection rules represent a kind of meta-knowledge. We believe that a separate explicit representation of such knowledge as in COCA (Major & Reichgelt., 1992) has advantages in comparison with the tagged to the instructional methods relative applicability conditions in GTE. Instead of „black-box" calculation of weights the methods are recommended according to explicit rules taking into account factors describing the situation and the student. It is far easier to adapt the system's selection by changing a rule

than viewing the whole set of instructional methods, comparing the relative conditions and tuning them to fit the teacher's preferences.

We found in pedagogical literature three alternative methods for teaching a concept (Einsiedler, 1976). The **"hierarchical"** method teaches by a sequence of the sub-tasks "introduce", "explain" and "give example", "give exercises" and, finally, "give a test". The **"advanced organiser"** method performs the same task-decomposition with an additional first sub-task which presents explicitly to the student the current teaching goal and the plan of sub-tasks that are going to be executed, what is expected from achieving the current goal, i.e. what is the importance of learning the current concept for the global goal. The **"basic concept"** method's first sub-task is to present a problem (exercise) whose solution requires knowledge of the goal concept. In case the student can't solve the problem, an introduction to the concept is given, an explanation of its main features, an example, the solution of the initial problem, an exercise, and finally a test.

Following Siegler (1988) we defined a rule asserting that the "basic concept" method has to be preferred for motivated students, the "advanced organiser" method is appropriate for not concentrated students, the hierarchical method - for concentrated ones.

The method-selection rules can also state that an appropriate method should be selected according to the history in the student model. For example, for the task "clarify concept" we have three alternative methods: "explain by description", "explain by example" and "explain by analogy". „Explain by example" will be selected if this method has been used successfully with the student before.

- **Teaching Materials Selection Rules**

For the current instructional sub-task the teaching rules decide how to select a TM on an appropriate type of media (i.e. text, graphics, animation or video etc.). They take into account the model of the student's preferences in order to select among alternative TMs which have the same pedagogical characteristics. They, however, might give a priority to a certain type of media which is preferred by the Teacher.

3.1.3 Authoring

- **Teaching Rules Editor**

The Teaching Rules Editor allows the *Teacher* to define her own discourse-, strategy-, method- and TM- selection rules. This is done by assigning conditions for the rule (variables from the student model) and effects (the appropriate choice of discourse, strategy, method or TM. However, the Teaching Rule Editor itself doesn't solve the problem of discovering the rules. How can one get such rules? Three approaches are possible:

- to define them ad-hoc, following some guidelines from existing didactic theories - the current solution;
- to interview teachers or to ask them to implement the rules directly themselves;
- to analyse protocols of teaching sessions, to identifying cases, generalise them to scripts, if possible, or apply machine learning techniques to generate decision trees and rules.

- **Editor for Instructional Tasks and Methods**

The majority of task hierarchies and decomposition methods are generic and can be defined in advance. However, like in GTE, a small set of instructional tasks and methods can be domain specific. The editor for instructional tasks and methods is supposed to be used by the A*uthor* to provide subject-specific instructional

tasks and methods for a specific domain. Its functionality is similar to the Domain Structure Editor: it allows creating, deleting, modifying of instructional task-structures (instead of Domain Structures). Alternative task-decomposition methods are represented by linking the task-nodes with arcs which have different colours, thickness and pattern exactly in the way the semantic of the different inter-concept relationships in the Domain Structure. Every leaf-node (not decomposable) sub-task is provided with a list of parameters denoting the appropriate pedagogical types of TMs which should be presented for this task.

3.2 The Student Model

The DCG+GTE's student model contains in addition a representation of the student's personal traits and preferences (playing the role of the „students attributes" in the student model in GTE). The traits and preferences are parameters accounting for features of the student which are important for the method selection rules. In the current version of the system, they include: intelligence, self-confidence, motivation and concentration. A set of variables accounts for the preferred by the student types of media.

The „Used TM" part of the DCG student model is generalised into a „history" representing in addition the instructional tasks / methods that have been used with annotation about their success.

3.3 Course Generation

The Course Generator is the component that creates the course, carries out the interaction with the student and maintains the Student Model. The Course Generator contains, like in DCG, two components: Planner and Executor.

3.3.1 Planning

The course Planner is the same AND/OR graph planning program as in the DCG. However, while in the DCG it is used only for planning the contents of the course (using the Domain Structure), in the combined DCG + GTE system it is used also during the execution to plan the teaching task-sequence for every current goal-concept. The Planner could have been integrated in the Executor since it is called by it very often: first - for planning and re-panning the content of the whole course, and second - for planning the presentation of each concept in the course. However, we decided to leave the Planner as a separate component. In this way the plan, regardless of whether it is a content- or a delivery-plan, exists physically and can be seen from outside. In this way the system has a certain conservatism or a tendency which it tries to follow until it fails, which, we argued (Vassileva, 1995a) has some pedagogical advantages. This is different from GTE which decides dynamically at every moment which method to use next. Another reason for this separation is the envisaged implementation of the DCG+GTE on the WWW where the architecture needs to be broken into small modules some which are downloaded from the server to the clients (Vassileva & Deters, to appear).

- **Content Planning**

The teacher invokes the Planner and assigns a teaching goal for the course, link semantic with respect to which the plan should be created and the maximum depth of traversing the graph. If for the particular domain there are discourse rules which assign these parameters for a certain teaching goal, the task of the Teacher is only to select the goal-concept. The Planner is activated to create a course plan. The course plan imposes only a

partial ordering on the nodes involved in it. The final ordering of sub-goals is done by the discourse rules or takes place at run time according to the selected teaching strategy (for example, by the student).

- **Instructional Task & Method Planning**

In exactly the same way the Planner creates an instructional task & method-plan when it has to teach each concept or relation from the content plan. In this case it works on the structure of instructional tasks and methods instead of the Domain Structure. According to the method selection rules the Planner selects a type of task-decomposition method according to which it decomposes the main task for the concept until atomic tasks are reached (see Table 1). The task plan consists of a sequence of such atomic tasks.

Table 1. Differences between Content and Task/Method Planning.

	Content Planning	**Task/ Method Planning**
AND/OR Graph represents:	Domain Structure	Instructional Tasks & Methods
Nodes:	concepts	instructional tasks
Links:	semantic relations among concepts	task decomposition methods
Selection managed by:	discourse-, strategy-selection rules	method selection rules

3.3.2 Execution of the Plan

The Executor consults the Discourse Rules and invokes the Planner to create a course plan for achieving the teaching goal. A main teaching strategy (structured or unstructured) is selected by checking the strategy-selection rules. If an *unstructured* strategy is selected, the system lets the student choose the current concept from a graphical representation of the plan and an instructional method from the task-hierarchies representation. If a *structured* strategy is selected, the Executor consults the discourse rules again and chooses the current concept or link to be taught; then it selects a main teaching task for the concept. Then it consults the method-selection rules, selects a task-decomposition method and invokes the Planner again to create a plan of the sub-tasks which are needed to implement the chosen method. Finally, the TM-selection rules are consulted to select an appropriate TM (see Figure 6).

The selected TM is presented to the student, then the next sub-task from the task plan is executed etc., until a sub-task involving a testing TM is executed which checks whether the concept is learned. Then the Model of the Student's Knowledge is updated according to the TM's conditional probabilities. With both main strategies, the unstructured and the structured one, it may happen that the student is not able to acquire some concept within the time provided for it. A sign for this is the insufficient knowledge probability of the concept in the student model ("sufficient" is a probability threshold defined by the Author). In this case the executor invokes the Planner to find a new content plan, bypassing the difficult concept. Our system provides two principal types of re-planning, local plan repair and global re-planning (Vassileva, 1995b). Local plan repair means that only the part of the plan related to the current goal 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 re-planning means finding an alternative plan for the main teaching goal.

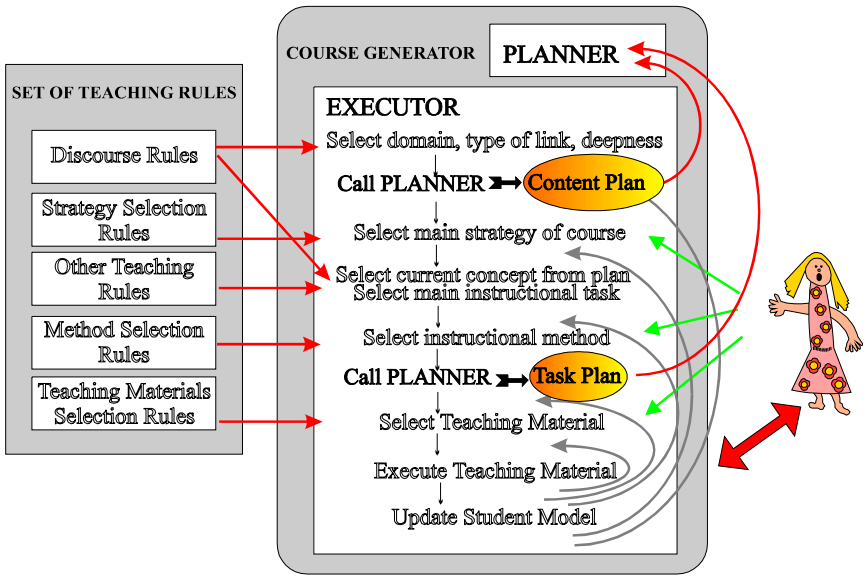

Figure 6: Course generation and execution.

4 Conclusions

GTE provides an original and flexible way of representing generic teaching expertise. It allows a high level of individualisation of the presentation of a given teaching contents and provides for adaptivity to the needs of the individual learner. A higher degree of flexibility and adaptation can be achieved by combining it with a planning program which is able to generate content plans depending on the *teaching goal* and the *viewpoint* assigned by a teacher. Such a content planner, the Dynamic Courseware Generator (DCG) has been developed and evaluated in several domains. The DCG and the GTE are in some sense complementary: GTE centres around the representation of teaching expertise, while DCG centres around the representation of conceptual domain knowledge. We describe the architecture and functioning of a combination of GTE + DCG and we suggest that such a combination will give an opportunity not only for intelligent presentation of the material, but also for intelligent selection of contents to be taught. This is a further step towards adaptivity to the needs of the student and towards flexibility not only with respect to the variety of teaching methods, but also with respect to teaching goals.

Acknowledgements: This work has been partially supported by Project I-406 with the Bulgarian Ministry of Science and Higher Education.

References

Beaumont, I. & Brusilovsky, P. (1995) Adaptive Educational Multimedia: from Ideas to Real Systems, *Proceedings of ED-MEDIA'95,* Graz, AACE.

Bohnert, A. (1995*) Analyse und Entwicklung pädagogischer Lehrstrategien für ein intelligentes tutorielles System*, Magisterarbeit, Philosophischen Fakultät der Rheinischen Friedrich-Wilhelms-Universität zu Bonn.

Brusilovsky, P. (1992) A Framework for Intelligent Knowledge Sequencing and Task Sequencing, *Proceedings ITS'92, Lecture Notes in Computer Science No. No 608*, Springer: Berlin-Heidelberg, 499-506.

Diessel Th., Lehmann A., Vassileva J. (1994) Individualized Course Generation: A Marriage Between CAL and ICAL. *Computers Educ.*, Vol. 22, No.1/2, 57-64.

Einsiedler, W. (1976) *Lehrstrategien und Lernerfolg: eine Untersuchung zur lehrziel- und schülerorientierten Unterrichtsforschung*. Weinheim: Beltz.

Elsom-Cook, M. & O'Malley, C. (1989) Bridging the gap between CAL and ITS, I.E.T., The Open University, Great Britain, 1989.

Flammer, A. (1975) Wechselwirkungen zwischen Schülermerkmalen und Unterrichtsmethoden. In Schwarzer R. & Steinhagen K. (Hrsg*.) Adaptiver Unterricht: zur Wechselwirkung von Schülermerkmalen und Unterrichtsmethoden*. München: Kösel, 27-41.

Major, N. & Reichgelt, H (1992) COCA: A shell for Intelligent Tutoring Systems. In Frasson, C., Gauthier, G. & McCalla, G.I. (eds.) *Proceedings ITS'92*, Berlin: Springer Verlag., 523-530.

Murray, T. (1992) Tools for Teacher Participation in ITS Design. In Frasson, C., Gauthier, G. & McCalla, G.I. (eds.) *Proceedings ITS'92, Lecture Notes in Computer Science No 608*, Springer: Berlin-Heidelberg, 593-600.

Nilsson, N. (1980) *Principles of Artificial Intelligence*, Morgan Kaufmann Publ., Los Altos, CA.

Paris C., (1993) *User Modeling in Text Generation*, Pinter Publishers: London.

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

Siegler, R. (1988) How Content Knowledge, Strategies and Individual Differences Interact to Produce Strategy Choices, in Schneider & Weinert (eds.) *Interaction among Aptitudes, Strategies and Knowledge in Cognitive Performance.* Springer: New York, 74-88.

Van Marcke, K. (1991) A Generic Task Model for Instruction, in *Proceedings of NATO Advanced Research Workshop on Instructional Design Models for Computer Based Learning Environments*, Twente.

Vassileva J. (1990) An Architecture and Methodology for Creating a Domain-Independent Plan-based Intelligent Tutoring System, *Educational & Training Technologies International*, 27,4, 386-397.

Vassileva J. (1992) Dynamic Courseware Generation within an ITS-shell Architecture. *Proceedings ICCAL'92, Lecture Notes in Computer Science No 602*, Springer: Berlin-Heidelberg, 581-591.

Vassileva J. (1995a) Reactive Instructional Planning to Support Interacting Teaching Strategies, in J. Greer (ed.) *Proceedings of the 7-th World Conference on AI and Education*, 334-342, AACE: Charlottesville.

Vassileva J. (1995b) Dynamic Courseware Generation: at the Cross Point of CAL, ITS and Authoring. *In* McCalla G. and Jonassen D. (eds.) *Proceedings of ICCE'95 - International Conference on Computers in Education*, Singapore, 5-8- December, 1995, 290-297.

Vassileva J. & Deters, R. (to appear) Dynamic Courseware Generation on the WWW, to appear *in Proceedings of CAL'97*, Exeter, 23-26 March 1997.

Wentland, M., Ingold R., Vaniorbeek C., Forte E. (1991) HIPOCAMPE: Towards Learner-Sensitive, Content-Optimized Interactive CAI, *Proceedings CALISCE'91*, Lausanne, EFPL, 233- 240.

Woolf, B. (1987) Theoretical Frontiers in Building a Machine Tutor, in G. Kearsley (ed.) *Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley: Reading, 229-267.

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