DYNAMIC COURSE GENERATION ON THE WWW

JULITA VASSILEVA

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

Abstract

The WWW provides a unlimited resource of materials for learning. However, learners have to find their way through links which sometimes are not relevant to their goals. We have developed a tool for authoring of adaptive CAL courses, called “Dynamic Courseware Generator” (DCG). It generates individual courses according to the learners’ goals and previous knowledge and dynamically adapts the course according to the learners’ success in acquiring knowledge. The DCG runs on a WWW server. The learner is a client; s/he receives a course corresponding to his/her learning goal and is guided through a set of teaching materials on the WWW.

1 Introduction

With the emergence of the WWW it became possible to provide learners with unlimited access to teaching materials (TMs) independently of time and geographic location. However, these materials are spread through the Web and it is not easy to find the relevant information when one wants to learn something specific. Though subjected to severe critics recently, the notion of „instructional course“ is very appropriate for implementation on the WWW because it ensures a goal-oriented way teaching. It is especially suited for adults who are motivated to learn to achieve a specific goal. We have developed a tool for ITS authoring, called “Dynamic Courseware Generator” (DCG). It allows automatic generation of individualized courses according to the learner’s goal and previous knowledge which can dynamically adapt the course according to the learner’s success in acquiring knowledge [1]. The DCG has been used as an authoring tool for generating adaptive courses in several domains and has proved to be very effective [2].

We have recently implemented the DCG on a WWW-server and now it can be used for domain-authoring and automatic generation of adaptive courses on the WWW. A „minimalist“ version of the DCG has been implemented on the WWW and tested in the domain of „Computer Based Learning Systems“. A more „intelligent“ DCG based on teaching expertise is under development.

2 The DCG on the WWW

The main idea of the DCG architecture is the explicit representation of the domain concept structure, separated from the TMs [3], [4]. The concept structure is used by the system as a road-map to generate a plan of the course. Given a certain goal-concept that the learner wants to acquire and a student model containing the concepts already known by the learner (initialized with a pre-test), the planner searches for sub-graphs that connect the concepts known by the learner with the goal-concept. The sub-graph is converted into a linear sequence of concepts by using a simple heuristic. In the “intelligent” version of the DCG the linearization follows certain teaching rules, described later. One of these plans is selected and offered to the learner to follow. The learner sees a sequence of teaching materials related to each concept from the plan. At every point the learner can be tested on his/her knowledge of the current concept and s/he will be given a set of test-items. A student model is created for every learner which is an overlay over the concept structure and uses a simple numeric heuristic to calculate the score of the learner's knowledge of every tested concept. If the learner is not able to achieve the needed score for a given concept, needed to proceed further with the plan, a new planning takes place.

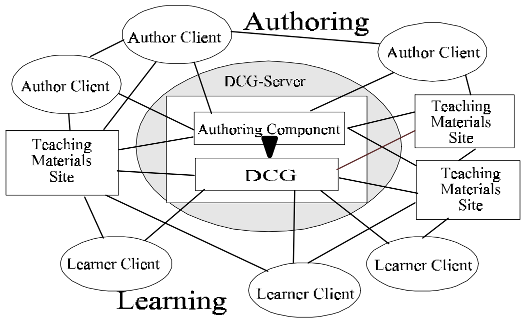


Figure 1: Client-Server Architecture of the DCG on the WWW.

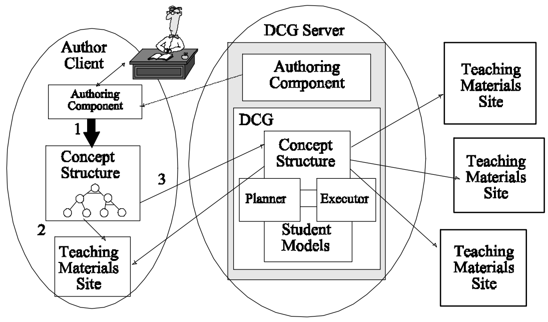


Figure 2: Authoring with the DCG on the WWW.

We adopted a client-server architecture for the system on the WWW (see Figure 1). The DCG is placed on a WWW-server; it offers teaching and authoring services. A client can be any WWW-browser. Learners and authors /teachers are clients and use the teaching and authoring services offered by the DCG-server. The TMs used in the courses are also distributed on various sites on the WWW. A special graphical editor for concept structures allows the creation of concepts, connecting them among each other with various types of semantic relations (e.g. abstraction, aggregation, analogy, temporal, causal, etc.). One-directional 1:n and n:1 relations are used to represent hierarchies with respect to abstraction, aggregation and causal relations, and bi-directional 1:1 relations to represent analogy and temporal relations. Every concept and relation can be linked to the http-addresses of one or more teaching materials (represented as html-files) and testing materials. They are html-files with an attached standard JAVA-applet which carries our the interaction and answer evaluation. JAVA was selected since it is platform- and browser-independent, while script languages are browser-dependent and CGI would have been too slow. A TM can address one or more concepts/relations and one concept/relation can be presented with one or more TMs. For the creation of the teaching and testing materials the author can use every html-editor or reuse existing html-sources. The creation of testing materials requires only link to the JAVA-applet and assigning the right answer to the question. The authoring process is shown in Figure 2.

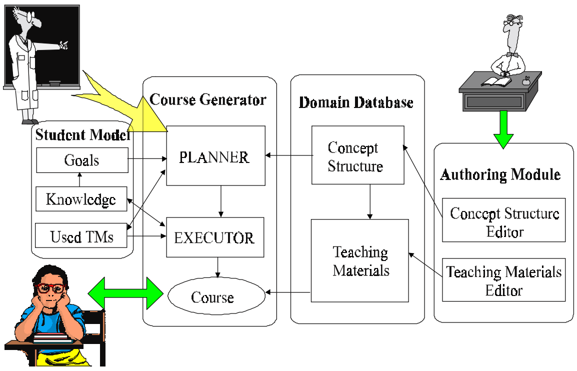
3 DCG on the WWW: the Compact Version

The compact version of the DCG runs on the WWW. It teaches currently in the domain of Computer Based Learning.

The DCG architecture is distributed to meet the requirements of a client-server organization. Some of the components (compare Figures 3 and 4) are kept on the DCG server (the Domain Concept Structure and the Planner), some exist both on the DCG server and the client (the Student Model and the Executor) and some are spread throughout the WWW (the TMs).

3.1 Architecture

The *Domain Concept Structure* is kept on the DCG server. It is represented as an AND/OR graph consisting of the domain concepts connected with relations with various semantics. For different domains different semantic relations are important, for example, in technical domain most often the type of relation is aggregation. In procedural domains causal and prerequisite relations are more important. Planning can take place with respect to one or several semantic relations, depending on the learning goal. A learning goal is defined as a goal-concept and semantic of relations with respect to which the plan should be made. The system offers for every domain a set of possible meaningful goals in a menu from which the learner can select.



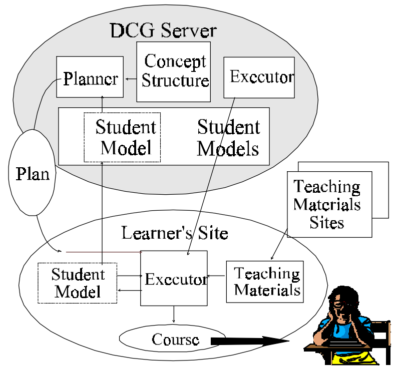


Figure 3: Architecture of the DCG [2]. Figure 4: Learning with the DCG on the WWW.

The *Teaching Materials* are html-files which can be distributed on different sites in the WWW. At authoring stage http-links from the concepts to the desired html-files are provided. Test materials consist of an html-file with associated standard applet for evaluation of the answer. So far applets for three types of standard test-materials have been implemented: multiple choice, blank-filling, and rearrangement of objects on the screen. Test materials must be defined for every concept with an indication of their difficulty and a coefficient, showing how much a correct / wrong answer contributes to the overall score of the related concept/link in the Student Model.

There are two instances of the *Student Model*. The first one is dynamic and is held locally on the client (learner's) site. A copy of this instance is stored on the DCG-server every time the learner finishes his/ her session and when a re-planning is required. It is used to make statistics of learners' success with different concepts which is useful for improving the TMs and domain concept structure (e.g. decomposing complex concepts). The current state of learner knowledge is represented as an overlay with the concept structure, containing the system's probabilistic estimations of the extent to which the learner knows a certain concept. These estimations are calculated using a simple formula from the number and difficulty of successfully solved tests-items related to this concept. Also a list of all used TMs is kept in the Student Model.

3.2 Learning with the DCG

The learner sends a request for a course to the DCG-site stating the desired domain and learning goal. The DCG sends him a preliminary test to initialize a Student Model or, in case of a registered user, it checks whether his/ her Student Model on the DCG-server is not too old. On basis of the learner's goal and the model of the learner's current knowledge, the Planner generates a plan of the course. The plan is a sequence of concepts and relations that have to be taught during the course. The learner receives his/ her individual course-plan and a copy of the Executor, which is a JAVA program and runs locally on the learner's site (see Figure 4). The Executor loads on the client machine several alternative teaching materials for each concept in the plan. The Executor chooses which concept from the plan the learner is prepared to learn next and presents TMs for this concept (see Figure 5).

If some TM contains links to documents that have not been included by the author in the list of TMs for the concept, the learner can freely browse the Web, following these links. A new browser is opened for free browsing, so that the learner can get back to the course exactly at the point where he left. The new knowledge, gained in this way is not taken into account in the Student Model. Thus, if the learner acquires knowledge about concepts which are going to be taught later in the course and in this way some part of the course becomes obsolete, the learner will still follow the preplanned course. However, the free browsing is monitored – the Executor stores the URLs which have been visited in the Student Model. This information is useful for the author to see what links are typically followed by learners and eventually to include them as TMs in the DCG, thus updating the set of TMs and keeping pace to some extent with the dynamics of the WWW.

When the set of downloaded TMs is exhausted or the if learner wishes explicitly to be tested - then the Executor presents a test on the current concept and updates the Student Model locally at the learner’s site. In case that the learner is not able to perform successfully on the test related to a certain concept, the Executor tries to teach him again by loading and presenting additional TMs. If it fails again, it contacts the DCG site; it sends there a copy of the Student Model and requests from the Planner a new course-plan, appropriate for the new state of the learner’s knowledge.

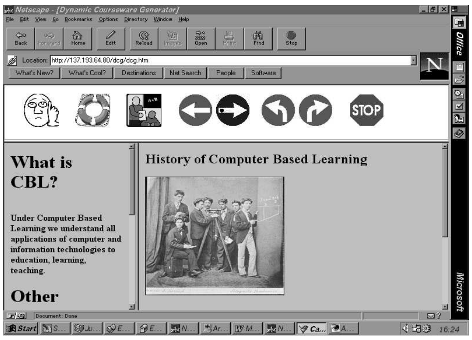


Figure 5. The DCG-Interface for the Learner.

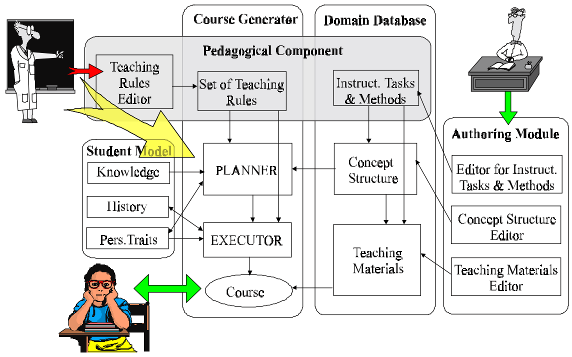


Figure 6: The „intelligent“ DCG architecture.

4 A More “Intelligent” DCG Under Development

The DCG by itself can not decide how to present the selected contents (the current concept or relation) to the learner. At execution time it just selects one or several TMs for each concept, but does not know how to sequence them. In this way the TMs remain discrete, there is no smooth transition between the presentations, and no possibility to present materials according to a certain teaching strategy. One way to solve this problem is to integrate the DCG with the GTE architecture [5]. In this combination, the DCG-part decides which concepts will be taught, i.e. 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 for the learner way, i.e. what types of TMs to select and how to sequence them The extended DCG + GTE architecture is shown in Figure 6. The combination of the DCG and GTE as a stand-alone application and its evaluation as an authoring tool has been described in more detail in [2].

In order to implement the more intelligent version of the DCG on the WWW several changes are needed in the contents of the components and their re-distribution in the overall architecture (among clients and servers). In principle the extension from the compact DCG to the „intelligent“ DCG involves three major steps:

- adding a Pedagogical Component with a representation of teaching tasks and strategies;
- planning not only on conceptual, but also on presentation level (of teaching tasks);
- indexing of the TMs (html-files) with respect to their teaching function.

4.1 Pedagogical Component

This component contains the knowledge that manages the sequencing of TMs while executing a course. It is downloaded on the client's site. The Pedagogical Component contains two main parts: a representation of the instructional tasks and methods and a set of teaching rules. In addition an editor for instructional tasks and methods and for teaching rules is provided for the authors to extend the teaching expertise of the system according their preferences, if they wish (see Figure 6).

The *Instructional Tasks and Methods* is a sub-component which contains a representation of instructional tasks and their decomposition into sub-tasks by means of different task-decomposition methods, like in GTE [5]. These task-structures can be represented with AND/OR graphs similar to that representing the Concept Structure. For example, Figure 7 represents the generic task "Give exercise".

The *Set of Teaching Rules* represents the system's teaching strategies. In this way the system's selection of teaching tasks for carrying out the content plan can be adjusted according to a specific teaching theory or to the teacher's preferences. The teaching rules encode different discourse and teaching strategies in order to manage the selection of the content plan (if several alternative plans are possible), the selection of instructional (task-decomposition) methods, and the selection of TMs. Most of the strategy-, method- and task-selection teaching rules are domain-independent. The rules defining a specific discourse strategy, however, are usually subject-specific. They take into account data from the Student Model (the learner's knowledge and personal traits) as well as external factors like time. Our set of teaching rules we developed after an analysis of didactic literature [6]. Four groups of teaching rules can be distinguished:

- *Discourse rules* **-** These rules manage the selection of a content-plan by manipulating the optimization criterion of the Planner. For example, they can select only plans according to certain types of semantic. 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. The discourse rules define also way of linearizing the plan (since the Planner generates a partially ordered sub-graph as a 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 parameters of teaching, for example, who has the initiative in deciding what to do next - the learner or the system. A "structured" strategy means that the initiative is in 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 learner by, for example, highlighting of the "ready to be learned" concepts.
- *Method-selection rules* **-** There are usually three to eight alternative task-decomposition methods (see Figure 7) for each instructional task [5]. The method-selection rules take into account the history of the used tasks and TMs and the learner'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.

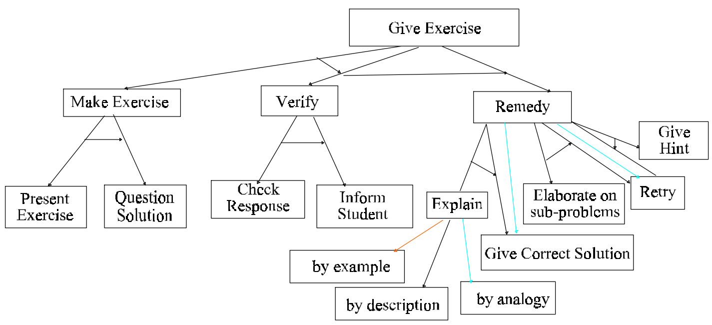


Figure 7: An example of a task hierarchy (adapted from [5]).

- *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 learner's preferences in order to select among alternative TMs which have the same pedagogical characteristics.

4.2 The Student Model

The Student Model in the „intelligent“ version of the DCG contains in addition a representation of the learner’s personal traits and preferences (playing the role of the „student’s attributes“ in the student model in GTE). The traits and preferences are parameters accounting for features of the learner 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 types of media preferred by the learner. All traits and preferences are static and are assigned by the learner himself. The „Used TMs“ part of the DCG Student Model is generalized into a „history“ representing in addition the instructional tasks / methods that have been used with annotation about their success.

4.3 Course Generation

In the „intelligent“ version of the DCG planning and re-planning takes place more often than in the compact version. Therefore it makes sense to download the Planner together with the Executor at the client’s (learner’s) site. We created a component called „Course Generator“ which generates the course, carries out the interaction with the learner and maintains the Student Model. It is downloaded on the client’s site together with the Pedagogical Component, which is needed to decide dynamically how to sequence the presentation of TMs for every concept of the plan. So, in Figure 4 instead of the Executor component on the learner’s site will be placed the Pedagogical Component and the Course Generator (consisting of the Planner and the Executor).

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

	Content Planning	**Task/ Method Planning**
AND/OR Graph represents:	concept 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

The Course Planner is the same AND/OR graph planning program as in the „compact“ DCG version. However, while in the „compact“ DCG it is used only for planning the contents of the course (using the concept structure), in the „intelligent“ DCG system is used also during the plan execution to dynamically plan the teaching task-sequence for every current goal-concept. It creates a task/method-plan by taking as input the structure of instructional tasks and methods instead of the concept structure. According to the method selection rules, the Planner selects a type of task-decomposition method and decomposes the main teaching task for the concept until atomic tasks are reached (see Table 1). Thus the task- plan consists of a sequence of such atomic tasks.

The Course Generator 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 learner 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 and 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.

The selected TM is presented, then the next sub-task from the task plan is executed etc., until a sub-task involving a test item is executed which checks whether the concept is learned. Then the Model of the Student's Knowledge is updated according to the test item's conditional probabilities. If the learner fails to acquire the concept (insufficient knowledge probability of the concept in the Student Model), 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. 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.

The "intelligent" DCG is being implemented currently. We believe the pedagogical expertise will help to create a more adaptive and individualized instruction. However, the question of how to combine pedagogical planning with the advantages provided by the WWW for explorative learning, i.e. how to combine the existing teaching methods and tasks with an explorative way of learning, becomes more important. A theoretical solution of this problem might be to adopt a reactive planning framework [7]. However this requires a certain degree of interactivity in the URLs visited during the learner's exploration (free browsing) which can't be expected from occasionally visited URLs on the Web. It seems that there are limits to the "intelligence" that can be incorporated into a system which uses the potential of the WWW.

5. Using the DCG on the WWW

The DCG has been already evaluated as a stand-alone authoring tool in several domains and its advantages have been shown in [2] quite clearly (18 hours of authoring for 1 hour of instruction). The „compact" version of the DCG has been experimentally implemented at the Federal Armed Forces University in the domain of Computer Based Learning Systems. The goal of the implementation was to see how such a system can be integrated in the teaching process of the University. The results of the test showed that the DCG on WWW will not make a „breakthrough" in the paradigm of university teaching, but it can be easily and usefully integrated in the existing organization. The following main applications were outlined:

- *Lecture Support, Distance and Continuous Education*. The courses generated with the DCG can be used as additional learning materials (as an interactive script) supporting lectures given regularly or occasionally at the University. The specifics of our university is that the all students are officers and are obliged to serve in the Army five years after graduating. Interactive courses on the WWW accompanying the lectures offered at the university would provide a „umbilical cord" between our students and their Alma Mater. It this way they can deepen and refresh their knowledge permanently.
- *Re-use and Sharing of Domains*. The distributed architecture of the DCG allows for authors to collaborate and cooperate in editing domain structures and relating TMs to the concepts. It also allows a reuse of TMs and domain structures. Libraries of often used concepts and corresponding URLs can be developed. An electronic domain represented in the DCG, linked to actual documents on the WWW, such as "hot" scientific papers, or just textbook explanations can be shared by lecturers teaching the same subject at different universities, where everyone can make extensions and modifications according to his/her personal view.
- *Learners as Authors*. Modern learning theories point out the positive effects of letting the learner create his/ her own understanding and knowledge structures feeling: motivation because of feeling "ownership" of the problem, development of searching – and meta-cognitive – skills. For this reason often students are left to plan a lesson themselves, for example, by organizing lectures as seminars. This enables them to create an own view of the domain, and an ability to search for new information. The DCG as an authoring tool can be used for carrying out this type of projects. For example, a student or a team can be assigned the task of authoring a certain theme (sub-domain). The student/ team has to review the literature, to discover important concepts and

relationships and create a domain structure, to create or find related materials on the WWW and to link them to the concepts in the domain structure. The structure and materials will be discussed and criticized by the lecturer and the class, and the domain produced in this way could be used later by the DCG for automatic generation of instructional courses on this theme.

6. Related Work

The DCG on the WWW is an example of a ITS-authoring system scaled up to work on the WWW. Another example of such system is ELM-ART [8]. They are similar in that they use a concept-based indexing of the educational materials which allows building a secondary navigation map. However, there are significant differences between ELM-ART and DCG. ELM-ART is one specific adaptive teaching system, working only in one particular domain, while the DCG is an authoring tool for creating adaptive teaching systems. In ELM-ART the presentations (i.e. the TMs) are automatically generated by the system's knowledge base and LISP is needed for this purpose. The DCG uses directly html-files, which can be collected from the WWW. There is also a difference in the style of teaching. ELM-ART provides a hyper-space for the learner to explore, which is in line with the constructuvist tendencies in learning environment design. The DCG produces goal-oriented courses in the traditional CAI sense. Other adaptive teaching system on the WWW exist [9], [10] which are similar to ELM-ART, but differ significantly from the DCG.

We believe that ITS authoring tools with distributed architectures like the DCG can provide a good start towards WWW-based adaptive educational systems.

Acknowledgement: Thanks to Ralph Deters, for the discussions and the implementation of the DCG on WWW and to the two anonymous reviewers for their useful comments.

References

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

[2] Vassileva J., Dynamic Courseware Generation: at the Cross Point of CAL, ITS and Authoring. in *Proceedings of ICCE'95*, Singapore, AACE: Charlottesville, 1995, 290-297.

[3] Vassileva J., Dynamic Courseware Generation within an IST-shell Architecture. Proceedings ICCAL'92, Lecture Notes in Computer Science No 602, Springer: Berlin-Heidelberg, 1992, 581-591.

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

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

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

[7] Vassileva J. Reactive Instructional Planning to Support Interacting Teaching Strategies, Proceedings of the 7-th World Conference on AI and Education, Washington, AACE, 1995, pp. 334-342.

[8] Brusilovsky, P., Schwarz, E., Weber, G., A Tool for Developing Hypermedia-Based ITS on WWW, Position Paper for ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs, Montreal, 1996. Also at: http://advlearn.lrdc.pitt.edu/its-arch/papers/brusilovsky.html

[9] Ibrahim B. & Franklin, S., Advanced Educational Uses of the WWW. *Proceedings of the Third WWW Conference,* 1995.

[10] Kay J.& Kummerfeld B., An Individualized Course for the C Programming Language. *Proceedings of the Second WWW Conference,* 1994.