

IEEE Educational Technology: ITHET03, July7-9 2003 Marakech/Morocco  
**An experimental intelligent tutorial system viewed as a hyper-document**

Kenji Hanakata  
Institute for Informatics  
University of Stuttgart  
Universitaetsstrasse 38, D-70569 Stuttgart  
Germany  
*hanakata@informatik.uni-stuttgart.de*

Guido Lorch  
Institute for Informatics  
University of Stuttgart  
Universitaetsstrasse 38, D-70569 Stuttgart  
Germany  
*lorchg@yahoo.de*

**Abstract** –There are many common features that generally characterize both tutorial systems and hyper-document systems. On the basis of our knowledge representation and management language SCOOOL<sup>1</sup> we developed an experimental intelligent tutorial system to come up with the current problems of development costs, the technical expertise of an authoring system and its limited intelligence function. The quintessence of our experimental tutorial system is that small-/medium-size intelligent tutorial systems can be developed in a short term, with less expense, if the base-system for authoring teaching materials is on the line of SCOOOL's functionality.

## I. INTRODUCTION

Traditional tutorial systems are linearly organized programs that present teaching materials in a sequence similarly specified in the textbook based on which, the system has been developed. The author of teaching materials is constrained to organize his versatile knowledge into a linear order in such a way that students may hopefully achieve their learning goal set by the author. This linear feature characterizes not only the tutorial systems, but also any training and drilling systems, generally speaking, almost all document based systems. It goes almost without saying, that viewing the teaching materials from the learner, however, this linear feature not necessarily means a good property in terms of learning efficiency and quality, as it is well known in any fields of learning psychology. The so-called multi-medial networked information systems of recent technological development open doors for a new era of learning evolution. Medial educators preach for learning effect of multimedia tutorial systems, a large number of education fairs are organized in every year to exhibit state of the art in educational technology. However, these systems are too difficult for teaching authors to write their teaching materials in a given state of the art educational systems or too expensive for short term projects, because they must be developed, whatever the size of materials and the scale of applications would be, by some large scale professional software enterprises that would usually take long period of time in cooperation with the author. For these reasons, it is not easy to develop a multimedia-based intelligent tutorial system within the limit of common educational budget and allowable time. Development costs and time are very crucial factors for the success of a small-scale tutorial system project. Viewing this situation we developed a prototype of a multi-medial intelligent tutorial system based on the visual knowledge-representation language called "SCOOOL", which we have developed for the application of hyper-documents in the past 15 years. In this short report we describe those aspects that

characterize the development of an experimental Intelligent Tutorial System (ITS) by SCOOOL without details of components such as strategy component, user model, expert which are involved in common ITS.

## II. HYPER-DOCUMENT LANGUAGE SCOOOL

In the evolution of the so-called object oriented languages there was a period in which researchers of Artificial Intelligence (AI) and computational linguists tried to represent knowledge in a uniform modular unit called "Frame" (Minsky, 1975), Carbonell, 1970, Bobrow.D. et al, 1977). Unfortunately these experiments did not evolve out of the academic community to a practical application field to be established as a programming language in general use, mainly because of Lisp as the implementation base-language that failed in practice. Taking powerful properties of Lisp as well as the practical framework of object oriented languages into account; we developed a language called SCOOOL, which has the following properties in tutorial application view (programming language proper is not mentioned here, Hanakata 1997):

- 1) Non-dichotomy of class and instance, only persistent objects as basic units (Lieberman 1986, Borning 1986) of interactive manipulation
- 2) Practically infinite number of objects with multiple dynamic inheritance
- 3) Coherent assimilation of row data (text, pictures and line drawings of different formats) into objects
- 4) Standard GUIB environment with dynamic line drawings for reference purpose
- 5) Integration of Internet environment into objects (SCOOOL's own internet browser).
- 6) Simulation and animation within the frame of tutorial presentation
- 7) Lexical terminology support for text presentation
- 8) Easy to learn SCOOOL by self guiding manual environment and versatile accessibility
- 9) extremely cheap development cost and time of application systems

We discuss the *basic* features of an instruction cycle in ITS in terms of SCOOOL's realization.

## III INSTRUCTION CYCLE

A typical instruction cycle basically consists of 3 phases

1. Information presentation by system
2. Question presentation by system and answering questions by tutand<sup>2</sup>
3. Evaluation and state transitions of instruction cycle.

<sup>1</sup> Simple C-based Object-Oriented Language

<sup>2</sup> We use the term "tutand" as the student user of a tutorial system

The above 3 phases go in iteration and/or recursion along with the interaction between ITS and tutand, depending on the teaching information strategies the author specifies in the system in view of learning situation inferred by tutand's answering to the questions. The author's teaching strategies usually implies psychological aspects of the learning situation in terms of tutand model.

## 1. Information presentation.

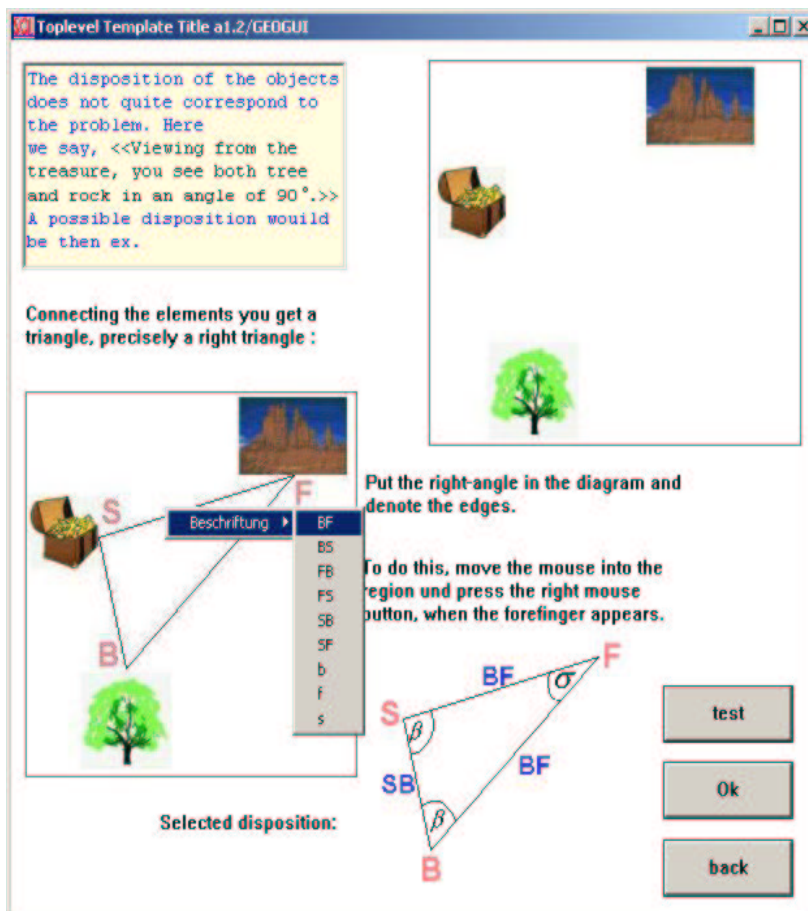
In the first phase of the instruction cycle, instruction specific knowledge (contents) is presented, so that the tutand understands the conceptual world and related factual data. The presentation is designed to aim at not only transferring factual knowledge to the tutand, but basic principles, which govern the surface information presented, or underlying knowledge structure. For this purpose multiple representations as well as aspects of knowledge about themes in focus are driven to facilitate tutand's understanding. Textual, figurative, realistic pictures and their motions find their cognitively efficient usage at right place and moment in the presentation framework to achieve better performance of tutand's knowledge perception and assimilation. The dimensions of presentation are not limited to morphological aspects, such as insertion of an additional explanatory window in form of texts and pictures within a basic parent window, but they extend to the abstraction dimension, which helps the tutand to filter the substantial information from the complex pragmatic situation of the presentation. Animations and simulations are powerful tools to express the complex situation in such a very compact way that would otherwise need a number of text pages. The sequence of presentation becomes similar to that of educational TV programs which requires huge investments and time. Cognitively efficient presentation presumes the tutand's knowledge and, generally saying, intellectual mental state concerning the contents. This requires the possibility of interaction between system and tutand, mostly by offering a group of check box, to find out a better form and level of presentation.

In conventional tutorial systems tutands can have only limited chance temporally to deviate the course fixed by the teaching strategy for understanding some technical terms or graphic objects contained in the presentation, since this requires tremendous flexibility and inference capability of ITS. The *tutand knowledge navigation* can extend, over the domain specific knowledge base prepared by the given system, to those knowledge resources that are available in the Internet environment. It should be pointed at this place that the HTML-formatted web-pages in association with java-applet are not powerful enough as tutorial pages (Benyon, D. et al.1997)

## Presentation tools for ITS in SCOOOL

### 1) Graphic User Interface Design (GUI).

Within a given top level presentation window the designer can interactively design a number of sub windows that recursively contain other windows, those visual interaction object (known as common controls in Win32 of Microsoft) and superposed by dynamic line drawings that can change its figures in runtime usage. (Fig. 1)



**Fig.1<sup>3</sup>** Geometry course “treasure hunt”

For our example of a simple geometry course with a treasure hunt (translated from German), a non-software specialist could very easily make this sort of tutorial page within a short time. This page contains buttons, text windows, picture windows that contain mouse-sensitive line drawing (triangle) with a pop-up menu. If the tutand selects an item, then the elements of the triangle below right are labeled. The position of treasure, tree or rock can be changed by the tutand. Some words in the texts can be designed to create a popup menu, which gives tutand a chance to supply more information about the explanation. Even a portion of given texts can be generated to adapt itself to the context and to the tutand state. Any region within a given text, picture and line drawing can be dynamically defined to be associated with certain tutand action, so that ITS' GUI can exhibit its more intelligent adaptive behavior. After a few hours of training a novice user can design this sort of GUI within a few hours excluding the time to prepare for 3 gif-formatted pictures. The design time for creating the instruction cycle strategy, mostly distributed among the objects associated with these graphic objects are also not included in the GUI

<sup>3</sup> The original German text is translated into English.

design time. The above treasure hunt example applied for triangle geometry makes smooth transition to a more abstract explanation and problem formulation (Fig.2)

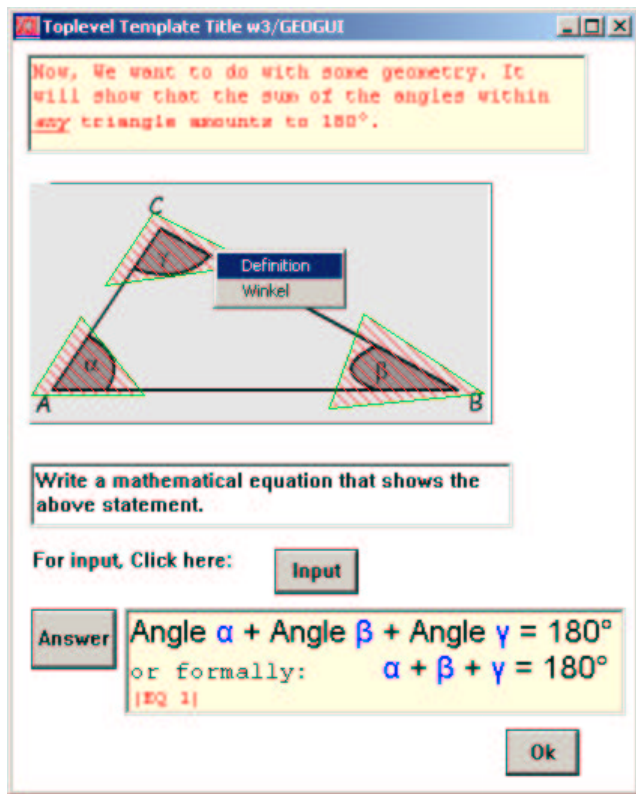


Fig.2 Smooth transition from simple treasure hunt geometry to an abstract general rule of trigonometry.

pictorial representation, they (row data) can be stored and retrieved in form of “card”. Fig.4 shows an example of SCOOOL object (left) which represent the graphic card (right) with superposed by a line drawing

## 2) Integration of Web-page into GUI of ITS

Indisputably saying, Internet offers the largest knowledge resources on which any tutand on a network can have an access. A web-address stored in an object can be activated directly or implicitly through a series of inference rules triggered by a method, so that the activation of a web-link and the embedding of a web-page into a given GUI take place in a uniform procedure with other interactive objects. The tutand does not notice his link activation. In Fig.3 the left sub window shows the simulation of trigonometry, where the tutand moves the top tip, while angles inside the triangle change. The texts in the right window created by our local GUI explains why the sum of the three angles is equal to 180 degree by showing the three angles put together on a horizontal line above the top tip. Though the left triangle graph is presented from a web-page, the author can synchronize the content of the right window with the mouse move of the tutand on the left window to be able to support the dynamic association of his try and error by textual explanation. This semantic synchronization is, without saying, limited by the third-party web-page. Some expensive simulations or animations developed by a third party can be integrated into the ITS, if they are found to be appropriated to the given context. (For reusability of teaching materials, see ex. CAREO Project 2001, Targeteam<sup>4</sup>, ARIADNE<sup>5</sup>)

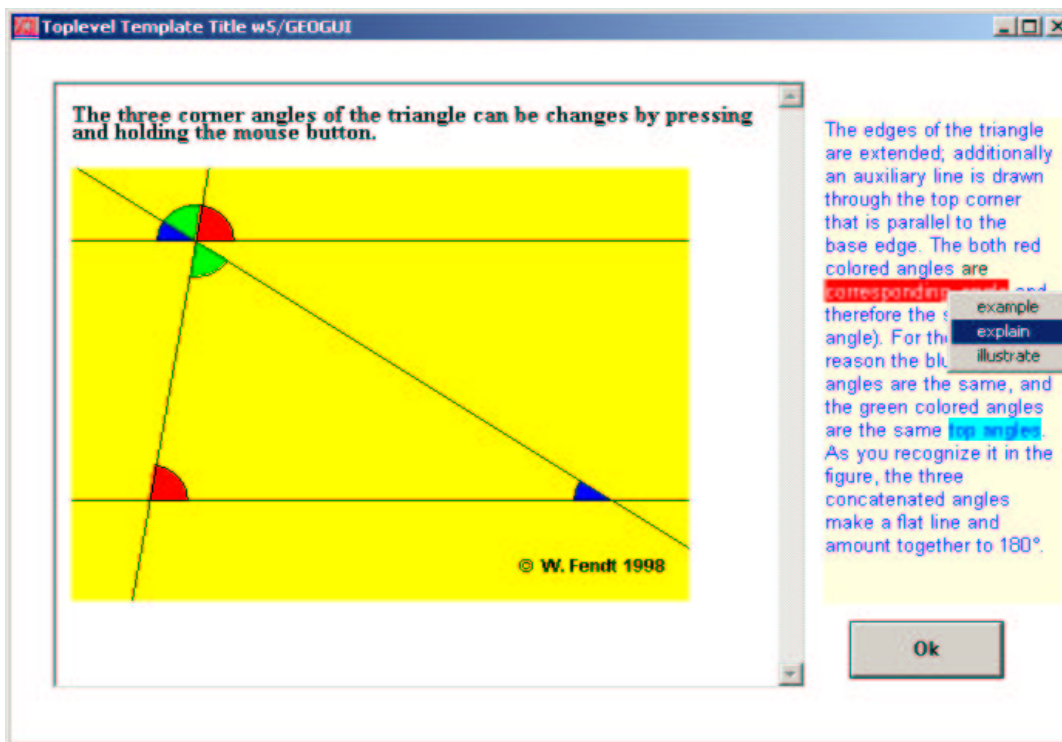


Fig.3 Integrating a Web page into ITS GUI and coordinating contents with the local texts in context

## 3) Uniform representation of texts, pictures and graphics as objects

In view of distance between structured knowledge representation in form of objects and row data such as text and

<sup>4</sup> <http://www.targeteam.org/>

<sup>5</sup> <http://ariadne.unil.ch>

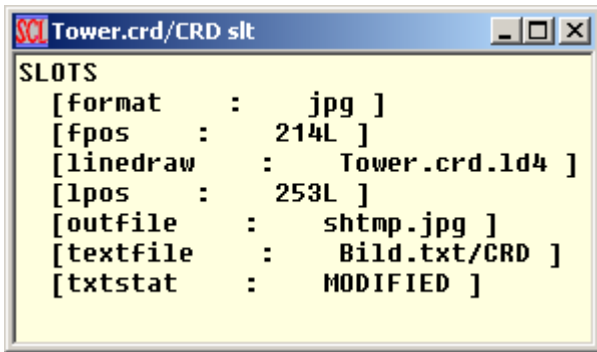


Fig.4 shows the object (above) that describes as a header for the (graphic) card (right). SCOOOL's card system enables GUI to handle different types of texts, pictures and line drawings in an uniform way and to convert them to active components such as by assigning dynamic menus or text attachments. Card objects serve as management units for structuring the GUI-objects.

## 2. System's question presentation and tutand's answering question

Though the state of the art of the natural language processing technology is very advanced in comparison to that in 1990's, being mainly promoted by the development of automatic translation projects, it is still far from the real application to the free answering analysis in any tutorial system. Instead of making an ITS more intelligent to "understand" the free answer sentences of tutands, it offers more elaborate multiple choice to recognize the quality of tutand's answering questions. For this purpose it is not enough to offer a multiple choice check boxes or list box in GUI, but ITS needs a sequence of context sensitive choices that control its query process depending on the tutand's preceding choices. For instance, the selection of an item in the first combo box decides the scope of possible



items in the second combo box or list. In our experimental ITS in SCOOOL, this dynamic query control is not restricted to a uniform query mode, but it often requires mix mode, depending on the contents of the query in a given context. Creation of interactive visual objects is dynamic in the sense that they can be created completely new from nothing or from prototype templates by methods at a run time when they are needed. This means that a visual object for the next query is created depending on the interactive process until now.

Fig.5 shows a snap shot of a query interface in which ITS tries to figure out how much the tutand knows about the geometry. The results of a series of queries are used to decide the level of presentation.

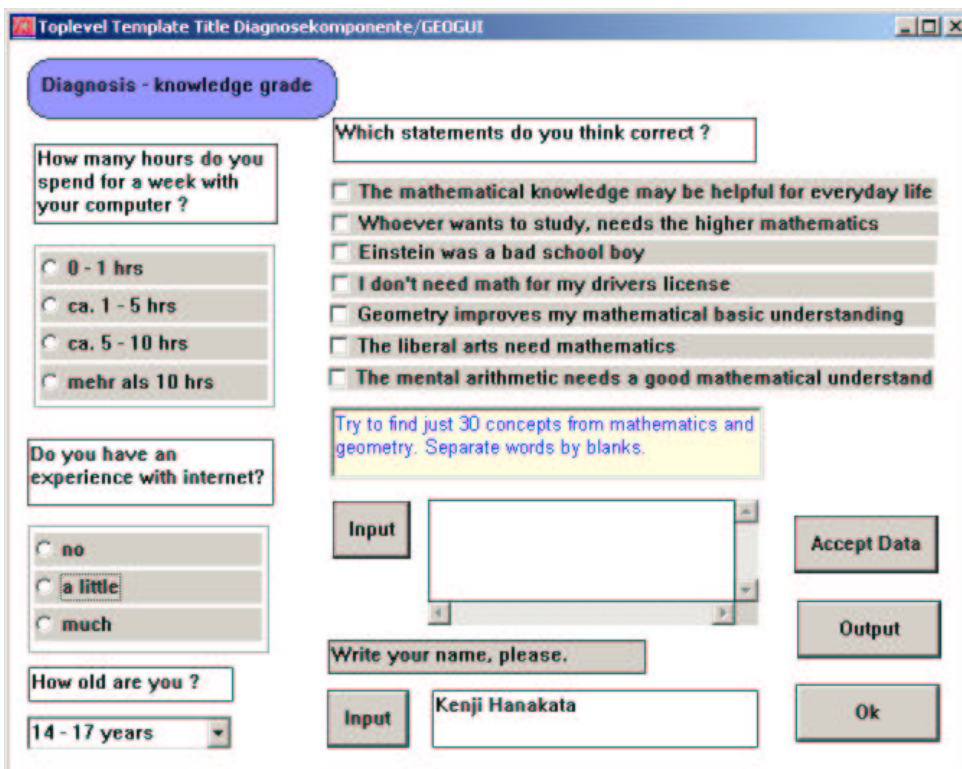


Fig.5

ITS query for initial classification of tutands supplemented by dynamic feedback information to improve the adaptive instruction strategy

### 3. System's evaluation and presentation of results

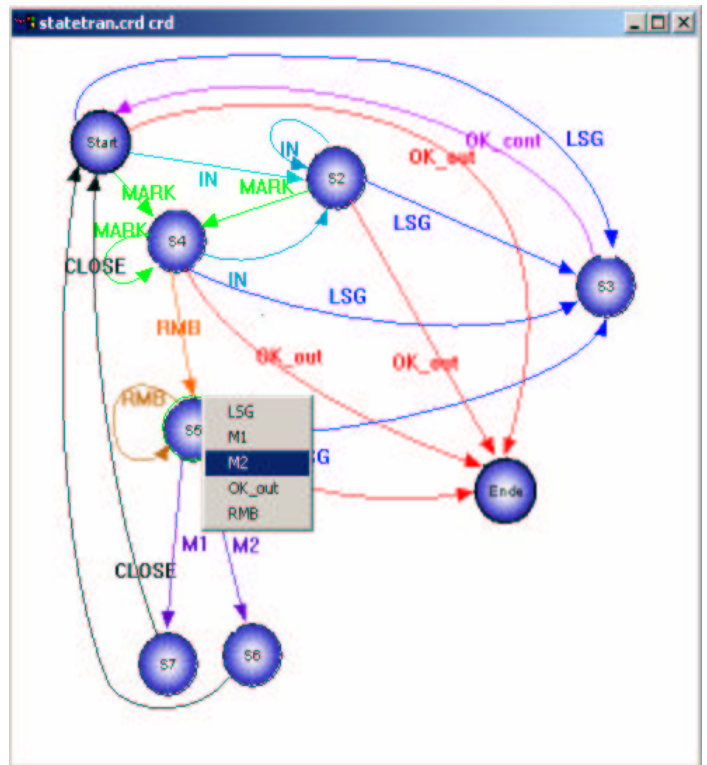
As indicated at the beginning of this chapter III, instruction cycle may continue in such a tutand-initiative recursion that ITS' presentation may be invoked by tutand's request for clarification or for additional information. The instruction cycle may also continue in an ITS-initiative way to achieve the local goal of the instruction cycle. Correspondingly the purpose of the ITS' evaluation is to keep the whole process of ITS-tutand interaction under control to bring the tutand to the overall goal of the author. The content of the instruction cycle depends on both state of the ITS and the tutand. We describe the overall instruction process basically as a state transition diagram, where each state of the diagram contains the state description of both ITS and tutand modell. This evaluation of the instruction cycle process is supported by the interrelation among those components such as expert component, instruction strategy component, didactic component that are all common in any intelligent tutorial system and are not explained here. In Fig.6 a *basic* state transition diagram is shown that describes instruction cycle process.

Every state transition is associated with a tutand or ITS action. Each state is formulated as an object, the outgoing arcs are registered as so-called slots in the object. Beside transition arc slots, the object may have other slots which describe the state. These state description slots trigger the transition by inherited method or state specific method.

Since the state transition graph is also created by SCOOOL's GUIB (Graphic User Interface Builder) it can be used to simulate the state transition, that means, the author clicks on a state node, the GUI offers a menu (see state S5) which is specifically assigned to the node, he selects an item playing as a tutand and sees if the intended action can be triggered or examines whether the content of the page presented as action is appropriate to the knowledge-state of the tutand. Though the continuous improvement of a tutorial system is only possible by means of the real feedback from the practice, this sort of simulation environment is indispensable for the author to design an ITS-tutand dialogue modell. The discussion about this topic seems beyond the scope of our paper as an application of hyperdocument and must, therefore, be dealt with in a separate occasion.

#### IV CONCLUSION

We developed an experimental intelligent tutorial system viewed as an application of hyperdocument on the basis of our interactive knowledge representation language called SCOOOL, which has been evolved from our continuing research work for interactive knowledge-based hyperdocument system. In this experimental system we didn't aim at building any practical system, instead, we kept attention to those bottle neck problems mentioned for a non-software-specialist when he designs an ITS. So far we concentrated ourselves to those techniques for ITS which are common in dealing with hyper-documents. The result of our experimental system is very encouraging in the sense that the development cost and time can be substantially reduced. In the following Table(Tab.1), time factors for layout of those ITS pages presented in this paper are given to help readers very roughly to estimate the technical time they need to create them. In figures 1~6 the time for



**Fig.6** A state transition diagram for ITS instruction cycle that underlies the design of ITS-tutand interactive processes. Since this diagram is also implemented in SCOOOL, it serves as a simulation tool for author's design work. A pop-up menu specifically created at a state node offers tutand/ITS possible actions.

creating gif-pictures, writing texts and simulation part behind the figure 6 is not included. For a very rough comparison you need for one line of SCOOOL method about 10 lines of VisualBasic++.

Fig.1	1.5 hours
Fig.2	1 hour
Fig.3	0.5 hour
Fig.4	0.5 hour
Fig.5	1.5 hours
Fig.6	2 hours

**Tab.1** Rough time for creating interactive pages copied as figures in this paper. Without saying, the more complex the associated actions, the more time you need.

Training time of SCOOOL for those authors who are not familiar with programming is unknown. However, one-day course for CS-students is enough to start and run along with the SCOOOL's self-guidance.

On the other hand, the author of teaching materials for ITS feels more powered in designing a control mechanism for ITS-tutand interaction. Now we are very motivated to expand our attention to those components such as didactic components, dynamic query strategy component, expert components, tutand answer evaluation component that are not common in hyper-documents, but characterize ITS in general, especially their synergetic effects that give more intelligent power to ITS to be designed.

## V. REFERENCES

- 1) Minsky,M. 1975. A framework for representing knowledge. In P.Winston (ed.), "The psychology of computer vision." New York: McGraw-Hill
- 2) Carbonell,J.R. 1970. "AI in CAI: An artificial intelligence approach to computer-assisted instruction." *IEEE Transaction on Man-Machine Systems*.MMS-11
- 3) Bobrow,D.G.Kaplan,R.M.,Kay,M.Norman,D.A. Thompson,H.Winograd,T. 1977. "GUS, a frame-driven dialog systems." *Artificial Intelligence* 8:155-173
- 4) Lieberman,H. 1986. "Using prototypical objects to implement shared behavior in object-oriented systems." OOPSLA'86, SIGPLAN Notice 21.no.11
- 5) Hanakata,K. 1997. *SCOOOL*. Research Report University of Stuttgart
- 6) Benyon,D. Stone,D.Woodroff,M. 1997. "Experience with developing multimedia courseware for the WorldWideWeb: the need for better tools and clear pedagogy." *International Journal of Human-Computer Studies* 47, 197-218
- 7) Borning,A.H. 1986, "Classes versus prototypes in object-oriented languages". In Proc. of ACM/IEEE Fall Joint Computer Conference pp36-40,.