WORLD-WIDE INTELLIGENT TEXTBOOKS

Department of Psychology, University of Trier, Trier, FRG

Elmar Schwarz, Peter Brusilovsky, and Gerhard Weber

Abstract: New WWW technologies allow for integrating distance education power of WWW with interactivity and intelligence. Integrating on-line presentation of learning materials with the interactivity of problem solving environments and the intelligence of intelligent tutoring systems results in a new quality of learning materials that we call I3-textbooks. In this paper, we describe the development of ELM-ART, an I3-textbook for learning programming that can be accessed via Internet and that is based on the on-site learning environment ELM-PE.

Introduction

Distance learning with Internet opens new ways of learning for many people. Now, educational programs and learning materials installed and supported in one place can be used by thousands of students from all over the world. World Wide Web (WWW) and WWW browsers provide a good example of powerful modern Internet facilities. Using WWW, a novice user can comfortably browse the Internet finding required pieces of information in different locations worldwide. From the very early days of WWW, there were many trials to use WWW facilities for distance learning. Being a good general tool for Internet navigation, however, WWW and current WWW browsers are not specially designed for distance education. Thus, most existing educational WWW applications use only simple solutions and are much weaker and more restricted than existing 'on-site' educational systems and tools. A number of powerful technologies which proved to be very effective in 'on-site' education are still not implemented within the WWW framework.

Two advanced educational technologies are of special interest to us: interactive hypermedia textbooks and intelligent tutoring systems. A prominent goal of the project we are working at is to integrate 'distance education' power of WWW with interactivity and intelligence offered by these advanced technologies.

From Simple Electronic Textbooks to Interactive Intelligent Textbooks

In the pre-computer society, a school or a university textbook used to be the primary learning support medium both in the classroom and at home. It is not surprising that several generations of researchers on computer use in education considered a textbook as a model for developing computer-based learning support tools. What is surprising is that a very big part of developed 'electronic textbooks' are no more than 'electronic copies' of printed textbooks: they offer the learner nothing more than access to the textbook content. Technically, current electronic textbooks (ET) are much better than their grandparents: first ETs used expensive mainframes and represented only text. CRT displays added graphics, personal computers made them cheap and available, multimedia technology added the possibility to present sound, video, and animation, and, now, Internet and World Wide Web bring the possibility of distance access. From the conceptual point of view, however, most ETs that are currently available on the WWW offer the student not much more than good printed textbooks that are available nowadays. Advanced research on computer use in education showed that computers can be a much better learning support medium. To see the difference let us limit ourselves to programming language textbooks. Traditional printed textbooks give hierarchically structured presentations of programming language concepts and constructs. In addition to the presentation of concepts, good textbooks use a lot of examples--from simple expressions to complete programs--and exercises--from simple tests to programming problems. Simple electronic textbooks just serve the same information on-line, sometimes with use of simple hypertext technology. Two important features, interactivity and intelligence, can be added in more advanced systems. Interactivity turns an electronic textbook from a passive into an active learning medium. Examples of adding interactivity are demonstrated by several 'interactive textbooks' [Boyle et al. 1994], [Fowler & Fowler 1993], [Meyerowitz 1995]. In addition to regular hypertext-based learning materials, some of these systems provide access to a programming environment with a program editor, an interpreter or compiler, and even a graphic program design tool. In such systems, all examples and problems are active teaching operations. The student can not only look at the example but also use the above tools to investigate it: to execute it, to change something, to execute it again, and so forth. The same tools can replace paper and pencil for developing and testing problem solutions interactively.

Another example of adding interactivity to textbooks is demonstrated by program testing and grading systems (e.g., Ceiligh [Benfordet al. 1994]). Ceiligh not only provides on-line access to the text of lectures and programming problems, but also can process student programs (i.e., problem solutions) and provide the student with important feedback. In particular, Ceiligh can test the correctness of a student's problem solution, measure its quality with several metrics, and report the results to the student. Such interactive feedback gets the students much more involved in the learning process. Interestingly, students often try to improve even correct solutions trying to get a better mark.

Further improvement can be achieved by adding intelligence into ET what means to make them perform some duties usually performed by the human teacher in the classroom. It is important to reduce the load of the teacher in the classroom and also in situations where a teacher is not available (which is mostly the case in WWW-based education). These aspects are investigated in the domain of Intelligent Tutoring Systems (ITS). Existing ITS for teaching programming can support the student in the process of problem solving, provide intelligent analysis of problem solution, and construct for each student an individual learning path, including individual selection of topics to learn, examples, and problems. Originally, ITS were considered as an opposite to ET technology. However, most current advanced ITS for programming (which are developed for the use in the real classroom) always integrate classic ITS features with programming environments, on-line course materials, and other features of interactive textbooks. For example, the ACT Programming Tutor [Anderson et al. 1995] provides problem-solving support, problem sequencing, and on-line course material. GRACE [McKendree et al. 1992] provides problem-solving support, on-line course material, and a program design tool. ITEM/IP [Brusilovsky 1993] provides intelligent course sequencing, adaptive on-line course material, solution checking, and an educational programming environment. ELM-PE [Weber & Möllenberg 1995] provides solution checking, intelligent program analysis, an educational programming environment, and intelligent selection of examples.

New generation ET should integrate on-line representation of learning material with interactivity of problem solving environments and intelligence of ITS. Such ET which we call I3-textbooks (where I3 means integrated + interactive + intelligent) really can provide a new quality over classic printed textbooks. A challenging research goal is to make I3-textbooks available on WWW. Among several other groups [Lin et al. 1996], [Nakabayashi, et al. 1995], [Nkambou & Gauthier 1996] we have started to work in this direction. In the following sections, we will give an example of how our ideas of I3-textbooks can be implemented in the domain of learning programming. We describe our research work on the development and implementation of ELM-ART, a WWW version of ELM-PE, that is currently one of the most advanced intelligent learning environments for programming.

ELM-ART (ELM-Adaptive Remote Tutoring)

From ELM-PE to ELM-ART

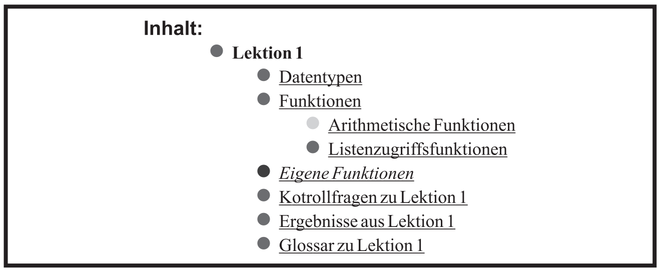
The knowledge-based programming environment ELM-PE was designed to support novices who learn the programming language LISP. It has several features that are especially useful when learning to solve problems in a new, complex domain. These features comprise a syntax-driven *structure editor* that helps beginners to code syntactically correct LISP expressions, *example-based programming* with examples chosen by the user or individually provided by the system on demand, *explanation of run-time errors*, *stepwise visualization* of the evaluation of LISP expressions and the *cognitive diagnosis of LISP code* based on the 'Episodic Learner Model' (ELM). All these features are described in more detail in [Weber & Möllenberg 1995] and in [Weber in press].

Though ELM-PE has been shown to be a very powerful tool for tutoring novices in LISP, there still exist problems. The goal of ELM-PE is to provide an environment under which the user can apply previously acquired knowledge in single programming tasks. But there is no opportunity for the user to repeat or to deepen knowledge. The user has to refer to a common printed textbook that suffers from disadvantages stated in the introduction. This results in the need of integrating a textbook into the environment. Adding this feature to ELM-PE resulted in the development of the I^3-textbook ELM-ART, an adaptive, world-wide available tutoring system.

Integrated Textbook

The first step to an I^3-textbook is to integrate an ITS and an electronic textbook into one single environment. ELM-ART contains a hierarchically structured ET that may be compared with common textbooks. Navigation is supported by links to neighbored pages and to a table of the contents of any desired node within this hierarchy. Additionally, a simple search interface helps to find special contents or topics. In addition to these old well-known techniques, ELM-ART provides adaptive navigation support to protect users from getting lost in the hyperspace. Among several known adaptive navigation support technologies [Beaumont & Brusilovsky 1995], ELM-ART mainly applies adaptive annotation of links. The system adds a dynamic annotation to any link presented to the user which tells the user what kind of page (e.g., text, example, or problem) will appear using a link and whether a link may be useful for the user [Fig. 1].

Interactive Problem Solving and Testing



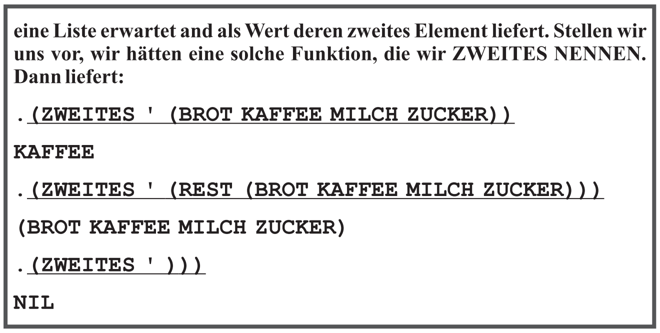
Inhalt:

- **Lektion 1**
  - Datentypen
  - Funktionen
    - Arithmetische Funktionen
    - Listenzugriffsfunktionen
  - *Eigene Funktionen*
  - Kotrollfragen zu Lektion 1
  - Ergebnisse aus Lektion 1
  - Glossar zu Lektion 1

Figure 1: Part of the Table of Contents showing the hierarchical structure of the texbook. According to adaptive annotations the link to the page "Arithmetische Funktionen" is suggested to be taken next, whilst the link "Eigene Funktionen" is not recommended.

An example in a textbook does not help very much if the user does not pay enough attention to it. That's why we want to provide an easy way for students to play with examples by integrating an enhanced evaluator into ELM-ART. This access is currently provided by transforming the text of any example into a link [Fig. 2]. When the user activates this link, the system loads the evaluator page, executes the example, and responds with the result or an error message [Fig. 3]. Employing the evaluator, the user can play with the given examples by modifying some pieces of code or by looking to step by step evaluation leading to a deeper understanding of the internal code evaluation strategy in LISP [Fig. 3]. Any occurring run-time error is explained by ELM-ART in more detail and more suited to novices than common LISP interpreters do. These explanations are sometimes augmented by hints that help correcting the error. The evaluator, supported by the above interface, has enough capabilities to serve as a simple but fully featured Lisp interpreter. Moreover, with its enhancements, the integrated evaluator of ELM-ART outperforms common LISP interpreters with respect to educational purposes.

Whenever the user is asked to solve a programming task,

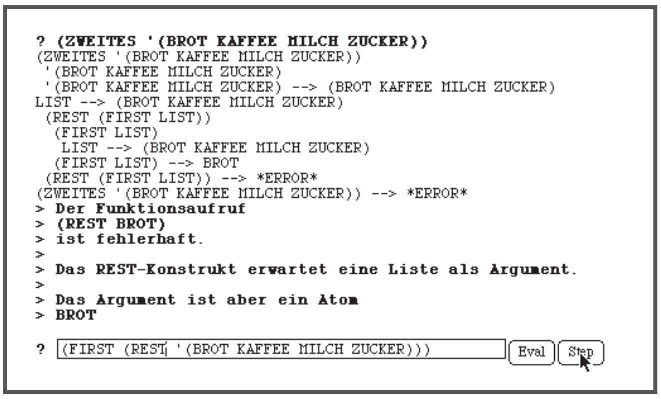


eine Liste erwartet and als Wert deren zweites Element liefert. Stellen wir uns vor, wir hätten eine solche Funktion, die wir ZWEITES NENNEN. Dann liefert:

```
. (ZWEITES ' (BROT KAFFEE MILCH ZUCKER))
KAFFEE
. (ZWEITES ' (REST (BROT KAFFEE MILCH ZUCKER)))
 (BROT KAFFEE MILCH ZUCKER)
. (ZWEITES ' )))
NIL
```

Figure 2: Part of a page from the textbook concerning the problem "ZWEITES." Using the links, the examples are put into an interactive evaluator page where they can be modified and reexecuted.

he or she can check a solution by calling a testing tool that tests the code dynamically with some critical I/O-pairs. In case of an erroneous solution, the user is faced with some counter examples. They may give a hint what type of error has occurred and where in the code it could be found. These examples may be tested interactively by the student



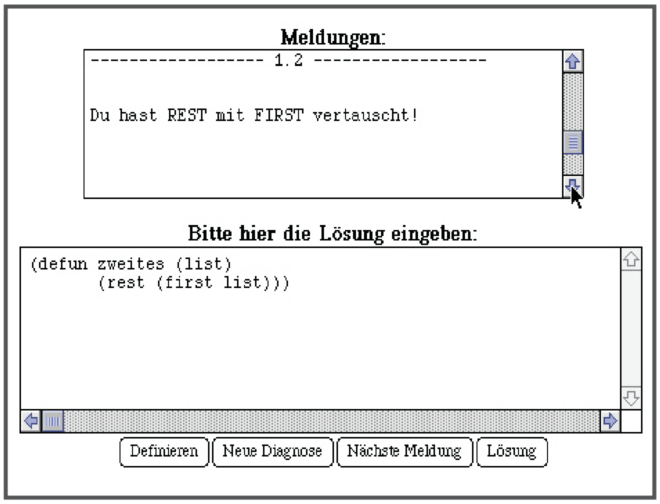
```
? (ZWEITES '(BROT KAFFEE MILCH ZUCKER))
(ZWEITES '(BROT KAFFEE MILCH ZUCKER))
 '(BROT KAFFEE MILCH ZUCKER)
 '(BROT KAFFEE MILCH ZUCKER) --> (BROT KAFFEE MILCH ZUCKER)
LIST --> (BROT KAFFEE MILCH ZUCKER)
 (REST (FIRST LIST))
  (FIRST LIST)
   LIST --> (BROT KAFFEE MILCH ZUCKER)
  (FIRST LIST) --> BROT
 (REST (FIRST LIST)) --> *ERROR*
(ZWEITES '(BROT KAFFEE MILCH ZUCKER)) --> *ERROR*
> Der Funktionsaufruf
> (REST BROT)
> ist fehlerhaft.
>
> Das REST-Konstrukt erwartet eine Liste als Argument.
>
> Das Argument ist aber ein Atom
> BROT

? (FIRST (REST '(BROT KAFFEE MILCH ZUCKER)))
```

Figure 3: Evaluation of the first example of [Fig. 2] based on the erroneous solution in [Fig. 4] using the stepper facility. Swapping FIRST and REST resulted in erroneously calling REST with the atom BROT.

in the evaluator at any time to see how the incorrect results are computed. This simple facility implemented in ELM-ART has been proved to be very powerful in several tutorial systems (e.g., [Barr et al. 1976], [Weber & Möllenberg 1995]).

Intelligent Program Analysis

According to our general approach, our I3-textbook includes an intelligent environment that supports problem solving. For this reason, all programming problems are described in the textbook. Any page describing a programming problem supports the user in different ways. In addition to testing and evaluating code as described above, the user is supported by intelligent program



```
------------------ 1.2 ------------------


Du hast REST mit FIRST vertauscht!
```

```
(defun zweites (list)
       (rest (first list)))
```

Figure 4: Response of the intelligent diagnosis for an erroneous solution for the problem ZWEITES. The error was caused by the swapping FIRST and REST. This is reported in the upper box "Meldungen."

analysis. If testing of a developed solution reports that it is still erroneous and the user is unable to find the error himself, he or she can invoke an intelligent program analyzer. As a result, the user receives messages that describe step by step which part of the code is erroneous and how to fix it [Fig. 4]. Therefore, the user is supported interactively at any time in solving the problem by testing and analyzing the solution as well as by an interface to the evaluator to execute some piece of code.

This feature of ELM-ART ensures that the user will find a correct solution to any problem. This is important because users often tend to assume non-optimal or even buggy solutions to be correct. Doing so, the user will acquire incorrect conceptual knowledge. When the same piece of knowledge is required again for problem solving, the user may completely fail as a result of the acquired misunderstanding. The system even provides feedback, whether the solution is optimal, and, if not, it suggests improvements.

Implementational Considerations

System Structure

Yielding in the goal of integrating as many features as possible, integrated textbooks make use of large textual databases for the actual textbook and of extensive knowledge bases about the subject for tutoring and example analyzing. For adaptation to individual users, knowledge about the user (e.g., the learner history) is required as well. I3-textbooks also employ several tools that require computational expensive algorithms handling the intelligent capabilities of the system. That is why fully featured intelligent textbooks are suitable only for expensive high-end computers. A reduction or limitation of any of these resources and features would not only result in a loss of comfort but also would lower the intelligence of the system. Since the user spends a lot of time in reading and thinking, the time between single user requests is rather high compared with computation periods caused by these request. Therefore, employing one high-end computer per student is an enormous waste of computational power and of physical resources. The only way to avoid either a loss of quality or a waste of money is distributing the system following the well-known client-server paradigm.

If we want to provide world-wide availability of I3-textbooks, the WWW is definitely the best choice to do that. This world-wide information service is supported by several front-end clients on nearly all common computer systems. Since these WWW-browsers can be run with small amount of memory, disc-space, and computational power, this kind of interface provides cheap world-wide access.

Platform of ELM-ART

ELM-ART is implemented based on CL-HTTP described in [Mallery 1994]. This tool provides a fully featured HTTP-server completely implemented in Common LISP and has shown to be an optimal platform for our purposes. CL-HTTP offers a Common Gateway Interface to handle incoming URLs from all over the world via the Internet. Any valid URL is associated to a response function implemented in LISP that is called by the server on an incoming request. The received URL and enclosed form values that may contain an arbitrary amount of incoming data are submitted as function parameters. Therefore, the response function can answer the request by using Common LISP standard I/O-functions to generate an HTML page as an adaptive response. Since a LISP function is called to handle the request, any desired information can be included to this page very effectively and, of course, any arbitrary interactive or intelligent tool can be integrated this way. Because the communication between client and server is so flexible, the system can be highly interactive and adaptive meeting exactly the requirements of I3-textbooks.

Selected Examples

The possibilities rising from a programmable WWW-server are enormous and not at all exhausted by ELM-ART. By explaining some special features integrated into our implementation we want to give an impression of the facilities that concern advanced adaptive text display, higher interactivity with complex data, integration of external tools, and enhanced feedback). Adapting Pages. This is facility is the basic part of our I3-textbooks. We avoid static pages as in older textbooks and make them adaptive by computing the page on the fly when the user requests it. For this purpose, our actual textbook is represented in a database that can be accessed via page identifiers to receive an outline page. These outline pages contain--besides the textual information--adaptive formatting options to enable the system to build an HTML-page based on the user's interaction and learning history. When the page is created it will be sent to the user who will not notice at all that the page was just generated for his or her individual purposes [Fig. 1].

Interactive Evaluator. To integrate the evaluator, we provide a form-based interface by which the user can submit own LISP code for evaluation. This is done by inserting HTML fill-out-forms in the page to be filled with LISP code by the user. By clicking a submit button, a HTTP-request attached with the form-contents is sent to the server. The associated LISP function calls the evaluator while its output is redirected to the remote client. At the bottom of the evaluator output, a new fillout form is inserted so that the user gets the feeling of a real LISP interpreter (listener) [Fig. 3]. Current HTML-standards and available network throughputs unfortunately do not allow a proper implementation of a real stepper facility. So, we decided just to show complete, scrollable traces of the execution that can be received by clicking a special submit button. Intelligent Solution Diagnosis. The implementation of an interface to the diagnostic part of the I3-textbook is even more difficult. While the input of the solution can be handled similar to the evaluator interface, the output of the diagnosis is more difficult to handle because it contains helpful information and hints up to the complete solution that should be presented step by step to animate the user to work out as much as possible on his or her own. We solved this problem by enclosing the diagnostic feedback into a scrollable window so that only one message is visible at one moment. The user can look at all messages step by step and can simultaneously improve the solution according to the pieces of information just read from the feedback window [Fig. 4].

Conclusion

The system ELM-ART described in this paper provides an example of a new kind of an electronic textbook. As many others textbooks available today on the WWW, ELM-ART provides remote access to a hypermedia-structured learning material which includes explanations, tests, examples, and problems. Specific features of our system include the possibility to play with examples, to solve the problems, and to get intelligent support which usually can be provided only by a human teacher. ELM-ART integrates the features of an electronic textbook, a learning environment, and an intelligent tutoring system. It is a real I3-textbook (i.e., integrated + interactive + intelligent textbook). We consider adding interactivity and intelligence to WWW educational applications to be an important direction of research.

References

[Anderson et al. 1995] Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). *Cognitive tutors: Lessons learned. The Journal of the Learning Sciences, 4, 167-207.*

[Barr et al. 1976] Barr, A., Beard, M., & Atkinson, R. (1976). *The computer as a tutorial laboratory. International Journal of Man-Machine Studies, 8, 567-596.*

[Beaumont and Brusilovsky 1995] Beaumont, I., & Brusilovsky, P. (1995). *Adaptive educational hypermedia: From ideas to real systems. In H. Maurer (Eds.), Proceedings of ED-MEDIA 95 - World Conference on Educational Multimedia and Hypermedia. Charlottesville, VA: AACE. 93-98.*

[Benford et al. 1994] Benford, S. D., Burke, E. K., Foxley, E., Gutteridge, N., & Zin, A. M. (1994). *Ceilidh as a Course Management Support System. Journal of Educational Technology Systems, 22, 235-250.*

[Boyle et al. 1994] Boyle, T., Gray, J., Wendl, B., & Davies, M. (1994). *Taking the plunge with CLEM: the design and evaluation of a large scale CAL system. Computers and Education, 22, 19-26.*

[Brusilovsky 1993] Brusilovsky, P. L. (1993). *Towards an intelligent environment for learning introductory programming. In E. Lemut, B. du Boulay, & G. Dettori (Eds.), Cognitive models and intelligent environments for learning programming. Berlin: Springer-Verlag.*

[Fowler and Fowler 1993] Fowler, W. A. L., & Fowler, R. H. (1993). *A hypertext-based approach to computer science education unifying programming principles. In T. Ottmann & I. Tomek (Eds.), Proceedings of ED-MEDIA'93 - World conference on educational multimedia and hypermedia. Charlottesville, VA: AACE. 203-209.*

[Lin et al. 1996] Lin, F., Danielson, R., & Herrgott, S. (1996). *Adaptive interaction through WWW. Proceedings of ED-MEDIA'96 - World conference on educational multimedia and hypermedia. Charlottesville: AACE.*

[Mallery 1994] Mallery, J. C. (1994). *A Common LISP hypermedia server. Proceedings of the First International Conference on the World-Wide Web.*

[McKendree et al. 1992] McKendree, J., Radlinski, B., & Atwood, M. E. (1992). *The Grace Tutor: a qualified success. In C. Frasson, G. Gauthier, & G. I. McCalla (Eds.), Proceedings of the Second International Conference, ITS'92. Berlin: Springer-Verlag. 677-684.*

[Meyerowitz 1995] Meyerowitz, J. (1995). *PasTIL: a multiple-environment interactive Pascal tutor. In H. Maurer (Ed.), Proceedings of ED-MEDIA'95 - World conference on educational multimedia and hypermedia. Charlottesville, VA: AACE. 786.*

[Nakabayashi et al. 1995] Nakabayashi, K. et al. (1995). *An intelligent tutoring system on World-Wide Web: Towards an integrated learning environment on a distributed hypermedia. In H. Maurer (Ed.), Proceedings of ED-MEDIA'95 - World conference on educational multimedia and hypermedia. Charlottesville: AACE. 488-493.*

[Nkambou and Gauthier 1996] Nkambou, R., & Gauthier, G. (1996). *Use of WWW resources by an intelligent tutoring system. Proceedings of ED-MEDIA'96 - World conference on educational multimedia and hypermedia. Charlottesville: AACE. [Weber in press] Weber, G. (in press). Episodic learner modeling. Cognitive Science.*

[Weber and Möllenberg 1995] Weber, G., & Möllenberg, A. (1995). *ELM programming environment: A tutoring system for LISP beginners. In K. F. Wender, F. Schmalhofer, & H.-D. Böcker (Eds.), Cognition and computer programming. Norwood, NJ: Ablex Publishing Corporation.*