Knowledge-Based, Automatic Generation of Educational Web Pages

Stefan Trausan-Matu

"Politehnica" University of Bucharest
Computer Science Department
Splaiul Independentei, 313, Sector 6
Bucuresti, 77206

Romanian Academy Research Centre for
Machine Learning, Computational
Linguistics, and Conceptual Modelling
Calea 13 Septembrie, 13,
Bucuresti, 74311

ROMANIA
email:trausan@cs.pub.ro, trausan@valhalla.racai.ro
http://www.racai.ro/~trausan

In “Proceedings of RILW Conference”, Ilieni, iun. 1997.

1. Introduction

The paper presents a knowledge-based program for automatic generation of a highly structured set of World Wide Web (WWW or web) pages, starting from a knowledge base represented using a frame (object-oriented) paradigm. This program is one of the modules of an intelligent tutoring computer system for advanced computer programming [TrN96]. The tutoring system is built around a structured (frame-like) object-oriented knowledge base developed in the XRL environment [Bat87], written in Common-Lisp.

The tutoring system uses different classes of knowledge: domain knowledge, pedagogical knowledge, psychological knowledge etc. It also includes student modelling, which enables flexible explanations and automatic test generation, adapted to each student's knowledge and personality. The most important is the domain knowledge base which is used by several modules. The taxonomic organisation of objects in the domain knowledge base is a starting point for the structure of the generated hypertext in HTML. Text in the generated Web pages may follow some text generation ideas [Tra96]. Links to other relevant Web pages may be included. The automatic generation of Web pages is also adapted to groups of students' personalities and to each student's knowledge.

The knowledge based character permits the easy evolution of the program through the addition of new knowledge about how to organise Web pages. Another consequence is that the generated teaching material is permanently kept up to date.

The organisation of the paper is as follows: The next section introduces the concept of knowledge-based program. The usefulness of such systems and the area of their applications are discussed. The knowledge processing environment used for the development of the web pages generation is shortly described. Section 3 is dedicated to the presentation of the intelligent tutoring system which includes the web pages generator which is described in section 4. Some conclusions are discussed in section 5.

Acknowledgements. The research presented in this paper has been done partially as a research program funded by the Romanian Academy and partially by Grant NR. 905, A55, of the National Research Council for University Research.

2. Knowledge-Based Systems; The XRL Structured Object-Oriented Knowledge Processing Environment

Knowledge-based systems (KBS) are programs in which a clear difference is kept between the knowledge they use and the procedures for processing it. This distinction permits the incremental development of the so-called knowledge bases, in which the knowledge is stored, while the processing procedures remain unchanged and are usually reused for a wide range of applications. This possibility is crucial for the development of computer programs for problems which are usually solved by humans which posses a large amount of knowledge in problem's domain. The reason is, first of all, psychological: It is very hard for a human to describe the whole amount of knowledge it uses. The description of knowledge is much easier in an incremental process. It is easy to understand what knowledge has the system by reading the knowledge base. In addition, the knowledge base may be used for several different purposes (e.g., constructing a solution, understanding a solution or generating explanations).

Knowledge is represented in knowledge-based programs according to several paradigms: predicate logic, rules ("if" situation "then" action), and frames (which have many common points with object-oriented programming). Some environments offer the possibility of the integration of several paradigms. Such an environment is XRL which has been used for the current system.

Learning is a knowledge centred activity: One of the main goals of a learning process is the articulation of a body of knowledge for the considered domain. The skeleton of this body is a taxonomy of the main concepts involved in that domain. These concepts have several attributes and relations connecting them with other concepts. Therefore, a frame-oriented knowledge representation is naturally suited for programs developed for supporting tutoring activities.

The intelligent tutoring system which is the subject of this paper has been developed in the classless object-oriented programming language XRL [BaT87]. In fact, XRL is the substratum of a multiple-paradigm knowledge representation and processing environment (including rules, constraints, demons, and a blackboard-based concurrent refinement system). It was implemented in CommonLisp on several operating system, from MS-DOS to VMS and UNIX. One of our principal reasons for choosing XRL (vs., for example, CLOS) are its declarative, multiple-paradigm knowledge representation facilities and the multiple knowledge processing possibilities.

For feeling the flavour of XRL, three objects are below exemplified. The "circle" object inherits the x and y slots from the "shape" object, has a particular slot named radius, and can respond to the draw message with a specific method (which overrides the method of shape). The circle32 object is a clone of the circle object (in a classless object-oriented programming language, there is not a cut between classes and instances; a particular object may be obtained by cloning an existing object):

```
(unit shape
  self (a meta-unit
         draw draw-shape)
  x undf
  y undf)

(unit circle
  self (a meta-unit
         supers (shape)
         draw draw-circle)
  radius undf)
```

```
(setq circle32 (a circle
                  x 2
                  y 3
                  radius 5))
```

An object may have a meta-object which is declared in the "self" slot. The "supers" sub-slot declares the list of objects from which the current object can inherit slots and methods. The assignment of methods to messages for the current object is described as pairs in the self slot.

Sending messages is accomplished with "msg". For example, the drawing of circle32 is done by sending the draw message to it:

```
(msg 'draw  'circle32)
```

In XRL, any slot may have a value which is a clone, like in the following example:

```
(unit map
  self (a unit supers (functional_on_list))
  name map
  body (a iterate_on_list
         c2c (a cons 1st (a funcall name f)
                     rest (a reccall name map args (f l)))))
```

3. The Intelligent Tutoring System

An intelligent tutoring system is a complex computer program that can be used by a learner as a personal, tireless teacher which adapts to the learner's cognitive particularities and his individual progress. Such systems are based around a large amount of knowledge from the teaching domain. Some of them use also pedagogical knowledge. Learner's particularities and the progress he achieved may be stored in the so-called "student's model".

The generator of web pages was developed as one of the parts of a complex intelligent tutoring system for computer programming in high-level languages [TrN96]. This system has, as a kernel, a structured object-oriented knowledge base for the programming domain. Starting from this knowledge base, the system can:

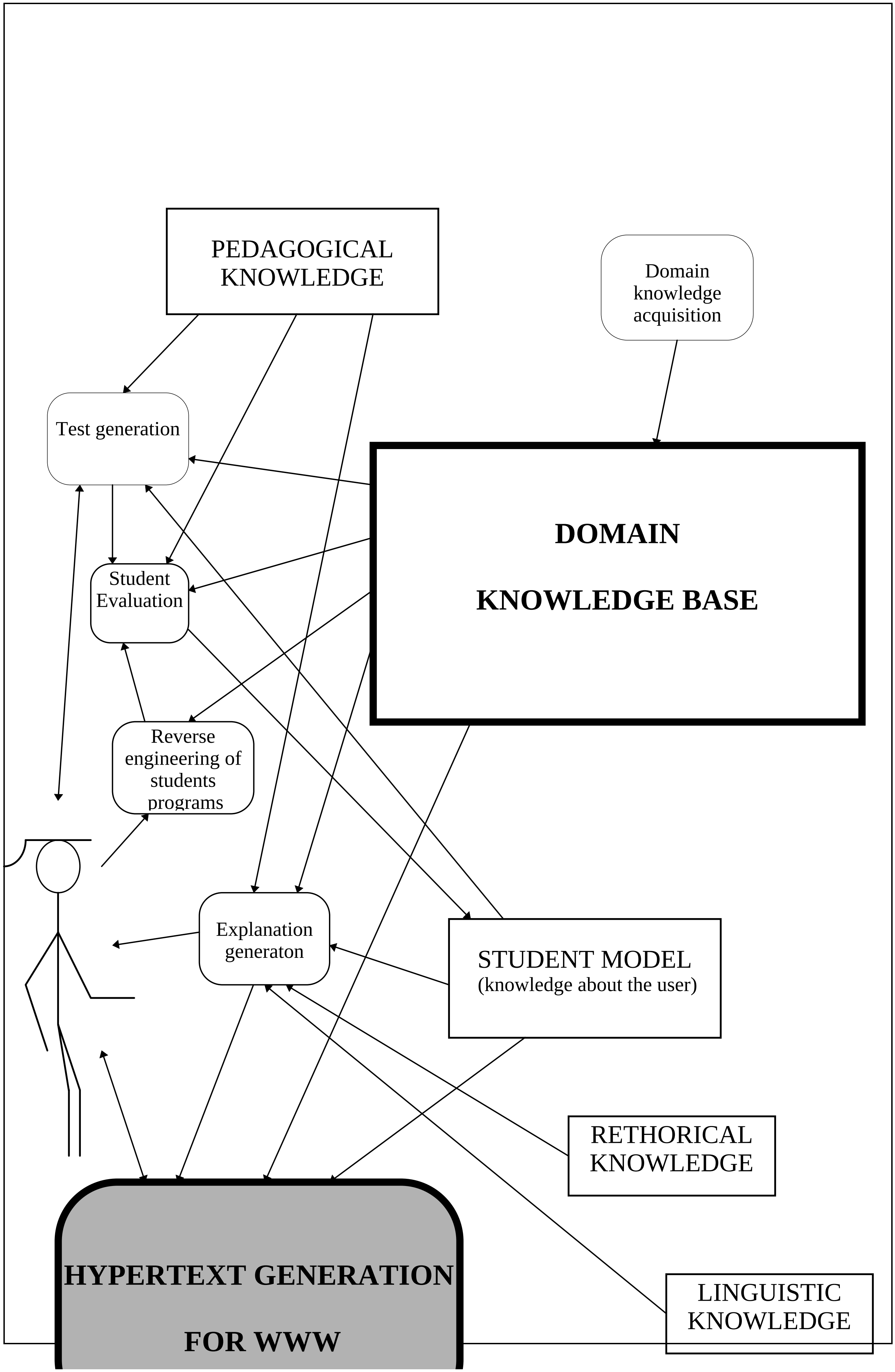
- generate flexible explanations in natural language;
- generate programs for illustration purposes;
- generate a huge series of tests;
- evaluate students results for tests, and develop a student's model;
- understand student programs (as in [Tra94]);
- generate a highly structured collection of web pages.

The tutoring system is tailoring its behaviour according to each student's personality. For example, explanations are different for each personality group and each lesson has a script determined by the student's particularity. For example, one of the possible scripts for a lesson has four steps, starting with browsing the web:

```
(unit script_version_1
  self (a unit supers (script))
  steps (examine-WWW
         testing
         bad-answer-analysis
         stop)
  examine-WWW
    (a browse-WWW
        next-step (testing)
        . . .)
  testing
  . . .
)
```

The addition of new knowledge is very simple due to the structured object-oriented knowledge representation language. A knowledge acquisition module is also provided. Rhetoric rules [Man88, Tra96] are taken into account for natural language text generation.

In the following picture, the structure of the tutoring system is depicted:


PEDAGOGICAL KNOWLEDGE
Domain knowledge acquisition
Test generation
DOMAIN
KNOWLEDGE BASE
Student Evaluation
Reverse engineering of students programs
Explanation generaton
STUDENT MODEL
(knowledge about the user)
RETHORICAL KNOWLEDGE
LINGUISTIC KNOWLEDGE
HYPERTEXT GENERATION
FOR WWW

4. Knowledge-based generation of web pages

A lot of people know today that the World Wide Web (WWW) is a network of nodes (pages) distributed on the whole planet and which can be browsed by anyone with direct manipulation, easy to learn tools like Netscape. Many of them know also that such a network of nodes is called also a hypermedia.

Unfortunately, my experience shows that even many computer professionals do not know that hypertext and hypermedia were introduced by Douglas Engelbart, in the first sixties, as a "Conceptual Framework for Augmenting Human Intellect" [Eng95]. Moreover, Theodor Nelson, who coined the term "hypertext", defined it as the hyperspace of concepts from a given text [Rad91], "a system for massively parallel creative work and study … to the betterment of human understanding" [Nel95].

These quotations emphasise the importance of considering hypertext (hypermedia) as a tool for augmenting human understanding and the necessity of a conceptual framework behind it. These two ideas are fundamental in our approach. First of all, one of the main goals of any learning process is the understanding of the theory of the domain taken into account. This is, of course, one of our main goals too, which we hope that our tutoring system and, especially, the web pages generator is supporting.

The second idea, which advocates the major role of a conceptual framework, is implemented by generating web pages which are linked according to the relations among concepts in a knowledge base. For each concept in the knowledge base there is a web page. The generated web pages are linked according to explicit (e.g. inheritance) or implicit (e.g. sibling) relations.

The World Wide Web has been proved as a very attracting and, meanwhile, very useful space to wander for almost anyone, including students. However, one of the deficiencies of the web is the lack of orientation. You could spend even whole days surfing in cyberspace, forgetting the starting point, the path you followed, or the starting goals (all these might be one of the causes of its attractiveness, it may become something like a narcotic). This fact might be good for fun but it is not at all a desired purpose in a tutoring context. Therefore, I think that a very clear structure of the links topology, easy to understand for anybody is a prerequisite, especially in web organized tutoring pages. The same idea is advocated by Kelvin Clibbon: "Cognitive overload and disorientation limits the effectiveness of hypertext for learning. By cognitively adapting a hypertext system to the user and by providing instructional cues, the effects of these problems might be reduced." [Cli95].

From an ideal perspective, everybody has to find WWW structured according to his needs and cognitive particularities. This might become a reality, on a small, local scale, when a collection of web pages are generated on the spot, for each user in turn. It became, in fact, a reality in the knowledge-based web pages generation system described in this paper.

All the preceding considerations in this section, and some general pedagogical considerations, determined us to introduce some principles which were the basis for our web page generator system:

- students must be encouraged to explore and experiment themselves but they need a guidance; browsing carefully generated WWW pages could be used as a part of a tutoring process;
- the system must tailor to each user; this includes also the web pages;
- all the explanations, other texts, and web pages must have behind a highly structured conceptual framework which correspond to the taxonomic organization of the domain knowledge base.

For illustrating the possibilities of the web pages generator, let us consider the following fragment of a taxonomy of programming concepts:

```
. . .
PROCEDURE
        CLOSURE
        CONTAINERPROC
            . . .
       SORTING_ALGORITHM
           RADIX_SORT
           INSERTION_SORT
           HEAP_SORT
           QUICK_SORT
           MERGE_SORT
           SELECTION_SORT
           BUBBLE_SORT
. . .
```

Each above concept is described as an object in XRL. The system generates a HTML file for each such concept. The collection of generated HTML files have a set of links among them. These links are automatically written by the system and they reflect some relations which are explicitly or not encoded in the taxonomy of concepts represented in XRL:

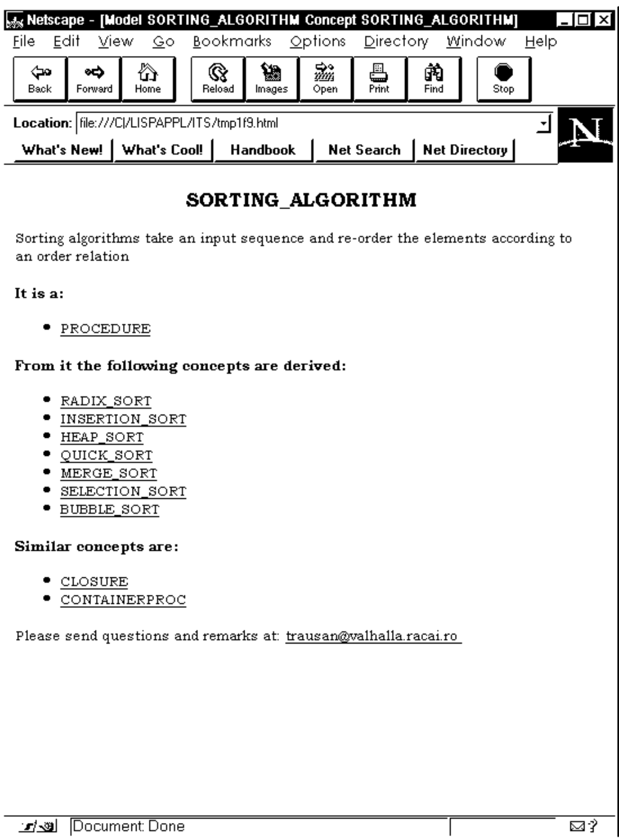
- “is-a” relation;
- “derived from” relation;
- “similar to” relation;
- “characterized by”;
- “consists of”.

For each concept, a definition may be written by the professor in the “text” slot of the concept. This slot may also be generated automatically by the natural language generation sub-system [Tra96].

For example, for the sorting_algorithm concept:

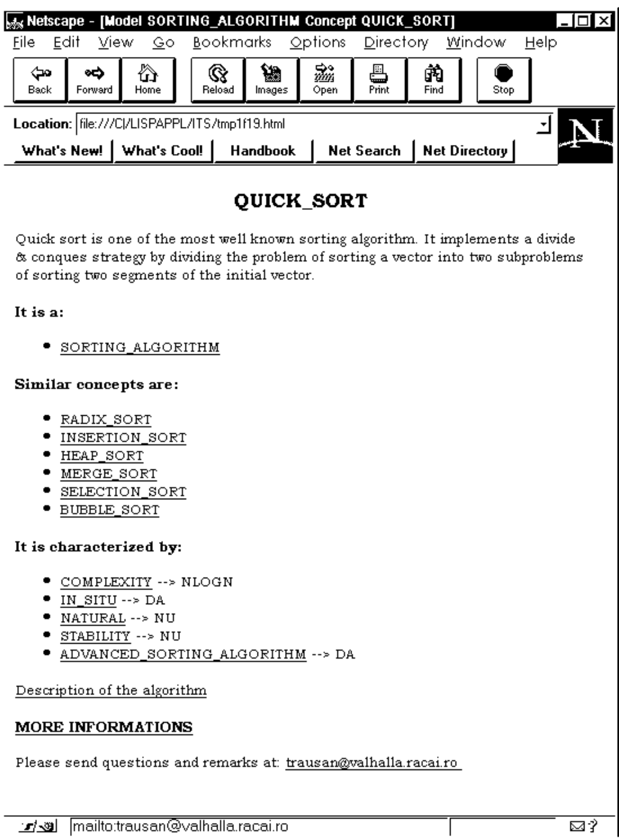
```
(Unit sorting_algorithm
    SELF
      (a UNIT
        SUPERS  (PROCEDURE))
    ALTS  (BUBBLE_SORT SELECTION_SORT MERGE_SORT QUICK_SORT HEAP_SORT
           INSERTION_SORT RADIX_SORT)
    text "Sorting algorithms take an input sequence and re-order the
elements according to an order relation")
```

the following web page was generated:



Another example is the quick_sort concept, derived from sorting_algorithm. In this example, another facility is presented: The professor may specify links to other files (for example, the source slot indicates a file which contains the source code for the quick_sort algorithm) or links to other pages, anywhere on the web.

```
  (Unit QUICK_SORT
    SELF
      (a UNIT
        SUPERS  (sorting_algorithm))
    DISCR  (complexity IN_SITU natural stability
            advanced_sorting_algorithm)
    complexity  (A complexity VALUE NLOGN)
    IN_SITU  (A IN_SITU VALUE DA)
    natural  (A natural VALUE NU)
    stability  (A stability VALUE NU)
    advanced_sorting_algorithm  (A advanced_sorting_algorithm
                                    VALUE DA)
    text "Quick sort is one of the most well known sorting algorithm.
It implements a divide & conques strategy by dividing the problem of
sorting a vector into two subproblems of sorting two segments of the
initial vector."
    links "http://sundy.cs.pub.ro/~trausan/quicksort.html"
    source qsort)
```



5. Comments and future improvements

One important advantage of our approach is that the structure of the generated web pages and of the local network of pages follow a very precise and easy to understand framework. In the terminology of [THH95], web pages generated within our approach are coherent (the reader can construct a mental model of the taxonomy) and the cognitive overhead is minimal (it is easy to recognise the taxonomic organisation and, moreover, it is always used by the system). However, we consider that some facilities for the visualisation of the concept taxonomy could decrease the cognitive load.

The usage of natural language text generation was only experimented till now. We intend to extend it in the near future.

References

[BaT87] Barbuceanu, M., Trausan-Matu, St., Integrating Declarative Knowledge Programming Styles and Tools in a Structured Object Environment, in J. Mc.Dermott (ed.) Proceedings of 10-th International Joint Conference on Artificial Intelligence IJCAI'87, Italia, Morgan Kaufmann Publishers, Inc., 1987.

[Cli95] Clibbon, K., Conceptually Adapted Hypertext For Learning, Proceedings of CHI’95, http://www.acm.org/sigchi/chi95/Electronic/documnts/kc_bdy.html

[Eng95] Engelbart, D.C., Toward Augmenting the Human Intellect and Boosting our Collective IQ, Communications of the ACM, vol.38, no. 8, pp. 30-33, aug. 1995.

[Man88] Mann, W.C., Text Generation. The Problem of Text Structure, in D.McDonald, L.Bolc, Natural Language Generation Systems, Springer-Verlag, 1988.

[Nel95] Nelson, T.H., The Heart of Connection: Hypermedia Unified by Transclusion, Communications of the ACM, vol.38, no. 8, pp. 31-33, aug. 1995.

[Rad91] Rada, R., HYPERTEXT, From Text to Expertext, McGraw-Hill, 1991.

[THH95] Thiring, M., Hannemann, J., Haake, J.M., Hypermedia and Cognition: Designing for Comprehension, Communications of the ACM, vol.38, no. 8, pp. 57-66, aug. 1995.

[Tra94] Trausan-Matu, St., Intelligent Reverse Engineering, PhD Thesis, “Politehnica” University of Bucharest, 1994

[Tra96] Trausan-Matu, St., Integrarea sistemului de asistare a instruirii cu un sistem de intelegere si generare a limbajului natural, raport RR-18, Academia Romana, CCAIAPLNMC, dec. 1996.

[TrN96] Trausan-Matu, St., Negreanu, L., Sistem inteligent de asistare a instruirii, raport RR-14, Academia Romana, CCAIAPLNMC, iunie 96.