# EDUCATIONAL KNOWLEDGE BASED ENVIRONMENT

Galina Dimitrova
AI Dep., Inst. of Math.
Acad.G.Bonchev Str.,bl.8
1113 Sofia, Bulgaria

Krasen Stefanov
Fac.of Math., Univ.of Sofia
A.Ivanov str. 5
1126 Sofia, Bulgaria

Abstract.

In this paper a knowledge based environment is represented which facilitates easy creation and preparation of wide range of educational software. The heart of the environment is the knowledge representation tool providing a hybrid knowledge representation formalism combining frames, production rules and logic. The environment is built on the basis of the logic programming language Prolog.

## 1. INTRODUCTION.

'Artificial intelligence (AI) is the study of how to make computers do things at which, at the moment, people are better.' [].
Although AI is still a young discipline various kinds of representation mechanisms and techniques were developed to solve problems. The results of AI research are more and more applied in other areas of computer science.

AI concepts and methods are also used to improve educational software (Aiken, 1990). The development of Intelligent Tutoring Systems, 'Microworlds' and Knowledge Based Tools provide teachers and learners with suitable tools assisting their work (Briggs, 1990).

In this paper a knowledge based environment for educational purposes is represented. The material is organized as follows: section 2 gives the overall structure of the environment, section 3 describes the knowledge representation mechanism, section 4 points to the current state, intentions for possible future development and applications of the environment.

## 2. OVERALL STRUCTURE OF THE ENVIRONMENT.

The represented knowledge based environment is developed to facilitate easy creation and preparation of wide range of educational software. It is formed from number of programming tools for: natural language understanding; representing and manipulating knowledge; creating games and models.

The whole environment is built on the basis of the logic programming language Prolog, which gives the following advantages:

- declarative as well as procedural approach;
- integrity and simplicity in extending the environment;
- powerful pattern matching mechanism;
- uniform way of representing knowledge and meta-knowledge.

The basic part of the environment is the tool for representing and manipulating knowledge, so it is described in more details in section 3.

Now we will shortly introduce the other tools:

- Tool for natural language understanding.
The tool is based on Definite Clause Grammars (DCG) (Pereira, 1980). It includes linguistic knowledge base and linguistic processor responsible to determine all valid natural language expressions for chosen object area and to form correct expressions.

- Tool for games and models.
It can be used for creating educational computer games or models of objects, processes and phenomenons. The tool uses a graphical extention to Prolog enabling various programming language PROLOG written in C and Assembler languages graphic commands to be used as built-in predicates. Its knowledge base contains key objects in the game or model, their properties, crucial phases, logic of shifting from one state to another, main aims and others.

All tools are independent and can be used individually, but they also have interfacing parts and could be linked with each other, forming more complex programming systems. Each programming tool is easy adaptable to meet various needs.

The main directions in which the environment could be used are the following:
- Knowledge based systems and expert systems, with natural language dialogue, enabling students to work with different knowledge bases.
- Computer games and models for training in various areas and gaining practical skills and abilities.
- Creating intelligent tutoring systems in different areas.

## 3. KNOWLEDGE REPRESENTATION TOOL.

The proposed tool for representing and manipulating knowledge realizes some techniques for knowledge representation in accordance with the frame idea of Minsky (Minsky, 1975): default reasoning, constraints, inheritance, attached procedures.

Frame based representation is a valuable tool for structuring the knowledge. By themselves, however, frame systems are not effective in knowledge processing. In order to obtain real knowledge processing power from a frame system some researchers propose production rules to be provided, too (Console, 1987).

### 3.1. Knowledge Base.

The knowledge base is defined as a collection of frames, in which frames-classes and instances are distinguished, and production rules.

The frame is an abstract data structure, which defines the modeled entity by a set of common properties or relations described by the slots of the frame. Fig. 1. gives an informal description of the frame structure.

```
frame :  <name of the frame>
 &isa :     <name of the superframe>
 <slot1>: &value :   <slot value>
            &default : <default value>
```

&multivalued : yes/no
&constraint : <predicate>
&if_added : <term>
&if_removed : <term>
&if_accessed : <term>
&if_needed : <term>
&query_form : <question>
<slot2> : ...
.
.
.

Figure 1. Frame Structure.

The slot &isa links the frame to its parent frame (superframe), so it inherits all the characteristics attached to the superframe.

The inheritance mechanism allows to describe a given slot only once in the hierarchy of frames and to override facets at lower frame levels. The &constraint facet of an inherited slot can be modified only in the direction of greater constraint.

While the frame describes a notion of the entity, the frame instances represent concrete objects. The instance consists of slots with attached values. The slots with attached values at frame level are not presented in the instance description, except a different value is supplied.

Frames, instances, slots and facets are stored as Prolog facts which facilitates the development of a knowledge base editor where it is important to have an efficient procedure for translating frame descriptions into internal frame representation and vice versa.

In the current state of the tool development process rules can be supplied at facet level only. Such rules are associated to events: assignment or supression of a value to an instance slot (&if_added or &if_removed); attempt to inferre the instance slot value from other information (&if_accessed, &if_needed).

We intend to elaborate the proposed frame system with productiion rules so that variables can be used as slot values and instances and general rules can be written applicable to instances of a specified frame.

3.2. Knowledge Base Editor.

The system is provided with a knowledge base editor which allows the user to build, modify and explore knowledge bases by using an interactive system of menus and dialog boxes.

The editor offers the following features:
- Initializing the process of creating a new knowledge base.
- Saving the knowledge base as a file containing Prolog clauses so that the internal knowledge representation can be examined and changed from within any text editor.
- Loading a previously saved knowledge base from a file.

- Manipulating the existing knowledge structure in the following directions: a new frame can be inserted into the KB; frame characteristics can be asserted, removed or modified; instances can be created or removed; instance slot values can be modified.
- Browsing the knowledge base structure: all frames or rules can be listed; desired frame description or instance contents can be displaied; the location of a frame within the hierarchy can be examined.
- Searching through the knowledge base for objects with specified properties.

### 3.3. Knowledge Base Interpreter.

A small number of predicates written in Prolog provide the following set of actions to be performed:
- Asserting frames, instances and rules into the knowledge base;
- Asserting slots or facets to existing frame description;
- Asserting instance slot values. Values attached to slots are checked for validity and the corresponding &if_added facets are fired.
- Generating an instance from a given frame. An unique instance name is generated from the frame name.
- Exploring the knowledge structure. Frame descriptions, slots, slot values, slot facets can be accessed, instance slot values can be achieved.
- Modifying frames. A given frame can be destroied only if it has no frames or instances generated from. A slot of a frame can be modified or deleted (whith all its facets) if only the frame, in which the slot is defined, has no instances and no successors.
- Modifying or removing instances. The user is not restricted in modifying or deleting instances and instance values. When an instance slot value is deleted the corresponding &if_removed facet is fired.

Rules are translated into Prolog programs so that no rule interpreter is needed.

## 4. CONCLUDING REMARKS.

We have represented a knowledge based environment intended to be used in education, paying more attention to the knowledge representation tool. Further improvements of this toll were proposed and discussed.

Now we are planing to apply the environment to build a concrete educational computer systems in areas of geography and travel planing.

References:

Aiken, R. M. (1990) More than Two Vowels: Putting AI into Computing Science Education, in McDougall, A. And Dowling, C. (ed.) Computers in Education, North-Holland, pp. 247-251.

Briggs, J., Nichol, J., Brough, D. (1990) PEG: A Way of Thinking about Things, in McDougall, A. and Dowling, C. (ed.) Computers in Education, North-Holland, pp. 1061-1066.

Minsky, M. (1975) A Framework for Representing Knowledge, in P. Winston (ed.), The Psychology of Computer Vision, New York.

Pereira, F., Warren D. (1980) Definite Clause Grammars for Language Analysis - A Survey of the Formalism and a Comparison with Augmented Transition Networks, Artificial Intelligence, North Holland, vol.3.