On supporting young students in visual logic modeling

Christos Fidas[1,2], Nikolaos Avouris[2], Vassilis Komis[2], Panagiotis Politis[3]

[1] Instance Ltd, Scientific Park of Patras, 26500 Rio Patras, Greece
fidas@instance.gr
[2] University of Patras, 26500 Rio Patras, Greece
{komis, avouris} @upatras.gr
[3]University of Thessaly, Volos, Greece,
ppol@uth.gr

Abstract. Logic reasoning presents notable difficulties for young children. This paper presents Logic Model Creator (LMC), a new environment that supports building and exploration of intuitive visual representation of logic models by young children. LMC logic models are structured as hypothesis, decision and / or counter decision components. These models are built using visual entities which represent the learning concepts of a specific domain. In this paper we focus on the architecture of LMC and the basic functionality of the environment. In particular we describe the the dynamic creation of equivalent logic models according to the so-called Reference Logic Model, constructed by the students' tutor. Furthermore an assessment module which provides immediate advice to the student in order to help them create a valid logic model is presented. Through experimentation it is demonstrated that the users of LMC can have rich interaction and assessment while exploring decision making logic constructs.

1 Introduction

The importance of using models of phenomena, activities or systems in learning has been widely recognized [1]. A number of software tools have been developed during the last years that support learning through modeling. These software environments mostly concern mathematical models of physical phenomena [2], while other modeling activities have also been proposed, like creation of concept maps, modeling of ecological and other complex phenomena [3], etc. A special case of modeling tools and activities relate to modeling logical propositions or logical constructs, proposed by scientists from science education and psychology fields [1]. Their purpose is to support children's reasoning and help them have access to decision making reasoning in a progressive way [4]. The reasoning of the students engaged in logical modeling involves studying and exploring logical propositions that are represented in visual form. Through them it can be deduced how the value of a concept or object property has an effect on other properties, which may in turn, affect other parts of the model.

Logic Models Creator (LMC) is a new learning environment which supports logic modeling activities for students of 11 to 16 years old. LMC is a derivative of the decision support component of an earlier modeling environment, ModelsCreator

Proc. 3rd IFIP Conference on Artificial Intelligence Applications & Innovations (AIAI) 2006, Athens, June 2006

version 2.0 (MCv2), originally built as a tool to be used for qualitative and semi-quantitative reasoning with real world concepts [5]. The original Decision Support component of ModelsCreator included a validation and model diagnosis module described in [6]. The limitations of that module have been tackled in LMC, as discussed in this paper. The logic propositions that can be built and explored with LMC meet the requirements of many curriculum subject matters, like mathematics, science etc., permitting interdisciplinary use of the logic modeling process. LMC puts great emphasis on visualization of the modeling entities, their properties and their relations. Visualization is crucial in supporting the reasoning development of young students and favors the transition from reasoning over objects to reasoning with abstract concepts [7]. This feature is extended also to the simulation of executable models allowing their validation through representation of the phenomenon itself in a visual way and not in an abstract mathematical relation or logical proposition, as it is usually the case. In fig. 1 an example of a model built using LMC is shown. On the left the hypothesis is visualized and on the right hand side the conclusion of the logical proposition. In this example the conditions are tested for deciding to prepare an application for hosting a child over the holiday season by a family.

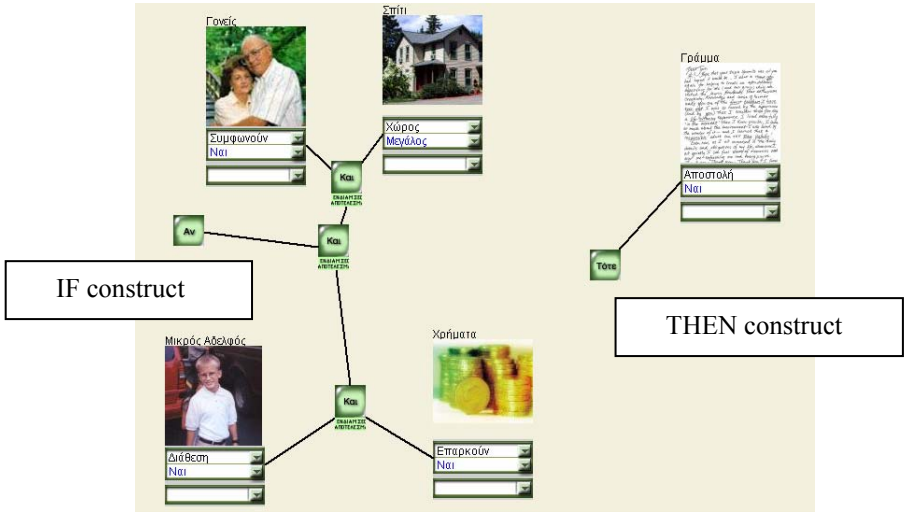

Figure 1. An example of an LMC logic model

An important aspect of LMC, as with the original MCv2, is its open character regarding the ability provided to the teachers in creating new logical domains (i.e. new subject matters) as well as new primitive entities which are needed for the creation of the logic models.

In the rest of this paper we present first the architecture and basic functionality of the LMC environment for the teacher and the student. We describe an example of use of LMC by groups of young students and discuss the implications of this environment in current teaching practice.

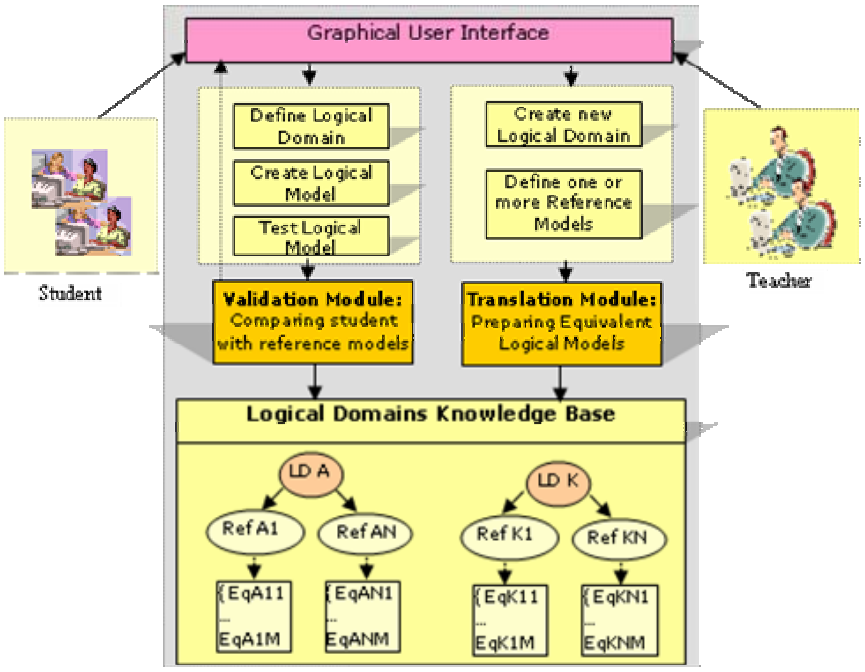

Figure 2: System architecture of LMC

2 Architecture of the Logic Models Creator (LMC) environment

Let M an LMC model, like the one presented in figure 1. This model can be represented as follows: $M= \{ E_i, i=1, \ldots, k, R_j, j=1,\ldots, l, A_m, m=1, \ldots, n \}$

Where Ei represents the node entity i of the model, Aj a property of a given entity, R a relationship connecting them. Examples are Entity=House, Attribute=Size. The relations which connect entities' attributes belong to the following set: AND, THEN, OR, AND, ELSE and NOT. Through them logical constructs can be built by the users using direct manipulation in the activity space. Using such an environment, one may construct expressions of arbitrary complexity.

The equivalent logical expression that can be built is:

Proposition = *IF* Construct *THEN* Construct
| *IF* Construct *THEN* Construct *ELSE* Construct
Construct = (Construct *AND* Construct) | (Construct *OR* Construct) | *NOT*(Construct)
| Attribute=Value

In this section an insight into the architecture of LMC system is given. An overview of the system architecture is provided in figure 2, presenting the main user categories and functionalities of the system described in this paper. A modular approach

has been followed, in order to reduce the complexity of the design. The aim of each module is to provide specific services to the modules with which it is connected, isolating the details of the construction of these services.

The system considers two basic user categories a) the students and b) the teachers who interact with the visual environment in order to accomplish specific tasks. While the main task of the students is to build and check the correctness of their logical models the main task of the teachers is to create new logical domains and define the reference models.

In the following some typical interaction scenarios of teachers and students with LMC are described in order to demonstrate the functionality of the architecture.

2.1 Teacher: Creation of a new logical domain

An important aspect of LMC is its open character regarding the ability provided to the teachers of creating new logical domains (i.e. new subject matters) as well as new primitive entities which are needed for the creation of the logical models. As a consequence, the educational environment brings additional value since it can adapt easily to the educational needs of different curriculum domains. Each logical domain represents one or more logical problems which describes decision making concepts that the students must explore. For each logical problem the teacher can create entities which describe verbally and visually the concepts included in the problem domain. Each entity may include a set of a attributes which describe specific characteristics of the entity. Some of them might be irrelevant to the problem at hand. Furthermore to each attribute of an entity can be assigned one more possible value. These values belong to a set which is defined in the creation phase of an entity.

2.2 Teacher: Defining a reference model

In the frame of a logical domain a teacher can define more than one Reference Model against which the students constructs will be subsequently tested. We consider these as models which describe alternative correct solutions to a given logical problem. The Reference Model should not violate rules related with the syntax of logical propositions of LMC. The knowledge representation used for expressing the Reference Model has been a matter of discussion during development of LMC and the previous environment MC. As discussed in [6], a first attempt was to express the Reference Model through Prolog statements, however this approach produced rigid logical models. An alternative proposed here is to use Truth Tables for representation of the Reference Model.

So for each Reference Model in LMC a teacher must complete a Truth Table which contains all the combinations of different events that exists in the Reference Model. If the hypothesis graph connects a set of $\{G_1,\ldots,G_N\}$ events of different attributes and each attribute can take values from a set $\{G_{11},\ldots,G_{1M}\}$ than the whole set of different states in which the hypothesis graph can be found is the Cartesian product $G_{if}=\{G_{1M} x G_{2M} x \ldots G_{NM}\}$

In a similar way we can define the decision part of the statement. If it connects a set

of {1,...,M} events of different attributes then we add to the truth table M columns which will be associated with values by the teacher. These values of the attributes in the decision graph depend on the values of the attributes of the hypothesis graph.
A truth table (see figure 3) is produced which contains the combinations of the hypothesis graph (according to the reference model of the teacher) and the values of the decision/counter decsion graph according to the values of the hypothesis graph.
In fig.3 the user interface of a Truth Table filling phase is shown. While the combinations of the hypothesis graph are filled automatically by the system the values of the attributes in the decision part must be filled by the teacher. After the teacher has finished the completion of the values the system produces logical equivalent propositions for each logical sentence in the Truth Table as described in the following.

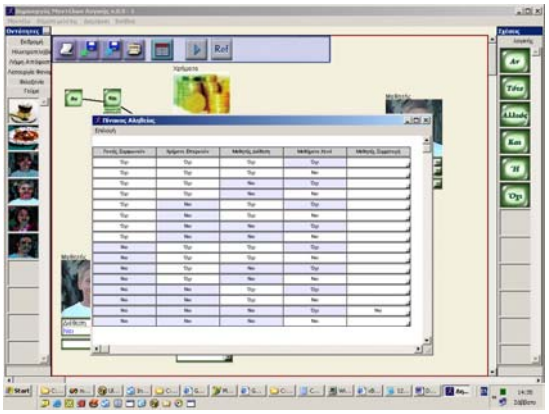

Figure 3.: Completion of the Truth Table values for definition of a Reference Model

2.3 Preparing equivalent logical models

As mentioned in section 2.2, every entity is defined by a closed set of attributes and each attribute can take a value from a closed set. This way, if an attribute N is defined by M different states it can take a value from a closed set of M values $M=\{V_{AN1},V_{AN2},\ldots,V_{ANM}\}$. Furthermore if we consider that in a logical sentence of the Truth Table an attribute N has taken a specific value K that belongs in M, then we can deduce that: $V_{NK} \Leftrightarrow NOT(M-K)$
i.e. attribute N has taken value K is equivalent with the fact: attribute N has not taken all the other values that exist in M except K, which in fact can be a way the students can express themselves while solving a logic problem.

At first the hypothesis, decision and counter decision graphs are considered. Equivalent graphs are produced according to the above approach. If $l_{if}=\{l_1,\ldots,l_N\}$ is the set of equivalent hypothesis graphs, $l_{then}=\{l_1,\ldots,l_k\}$ is the set of equivalent decision graphs and $l_{else}=\{l_1,\ldots,l_J\}$ is the set of counter decision graphs then the whole set L of equivalent logical models is the Cartesian product of $l_{if,}$ l_{then} and l_{else} : $L=\{l_{IF} \times l_{THEN} \times l_{ELSE}\}$

2.4 The Logical Domain Knowledge Base

The knowledge base consists of the logical domains, the logical problems, the Reference Models defined by the teachers and the equivalent Logical Models deduced by the system as described in section 2.3. Each logical problem can have one or more Reference Models. Furthermore each Reference Model can have one ore more equivalent Logical Models, created automatically by the system using the above approach. Each logical model in set L (all the correct models for the logical problem) is saved in and expressed internally in the form of logical propositions: *Logical Proposition= If (Hypothesis graph) Then (Decision graph) Else (Counter decision graph)*

2.5 Student: Defining a logical domain

In order to evaluate a model the student has to specify first the logical problem in a logical domain. The logical domain module informs the active logical domain module about the logical problem that has been selected by the student. The active logical domain module gets form the knowledge base the set of all correct logical models that describe the selected logical problem.
After the logical domain has been specified the student can create and test the correctness of his logical models. First the translation module is activated in order to translate the students model from the graphical into textual representation similar with the representation of the logical models in the knowledge base. The validation module compares the student model with the models in the knowledge base of the active logical domain and provides to the user the appropriate feedback.

2.6 Student: Validation of a student model and providing feedback

We consider that a logical problem consists of a set of correct logical models L $L=\{l1,...,lN\}$. The purpose of the system is to support the student with appropriate feedback in order to build a model that is equal to a correct logical model in the knowledge base of the active logical domain. To attain this aim, the system creates and displays messages using a relevance factor.
In the case that the student model is equal to a model in set L a message is produced in order to inform the student about the correctness of his model. In any other case the system has diagnosed that the student model has no equal model in set L, it sttempts to find a model in set L which is similar with the students model. With the aim to achieve this goal the validation module scores each model in set L regarding with the student model using the following equation.

$$score = f_{Entities}+f_{attributes}+f_{attributeValues}+f_{Realtions}$$

where

- $f_{Entities}$ =(Student.Correct.Entities/Total.Needed.Entities)
- $f_{attributes}$ =(Student.Correct.Attributes/Total.Needed.Attributes)
- $f_{attributeValues}$ =(Student.Correct.AttributesValues/Total.Needed.AttributesValues)
- $f_{Realtions}$ =(Student.Correct.Relations/Total.Needed. Relations)

The logical model with the highest score is defined as the closest logical model to the developed student model. The validation module specifies the feedback message provided to the student according to the level of similarity between the student and the closest model..
The validation module of LMC checks the level of similarity in (a) the Entities level (b) the Attributes level (c) the Attributes values level and (d) the Relations level and provides the student with appropriate feedback messages in order to support and scaffold the modelling process.

3 Case study of use of LMC

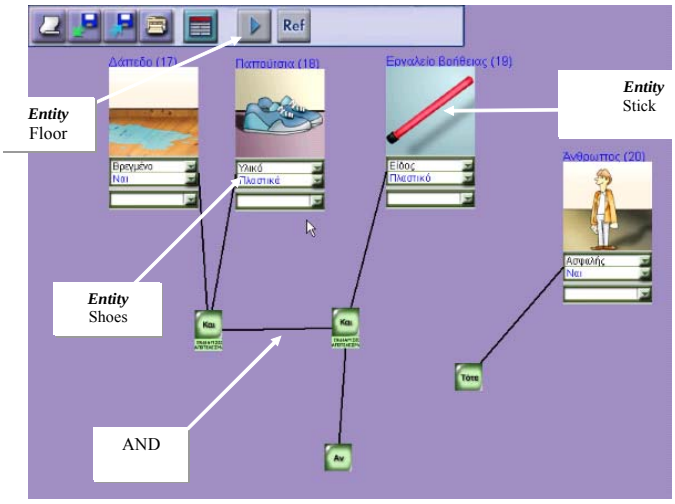

Figure 4:The logic model exploring “the electric shock” problem

In a recent case study, that involved use of LMC, a pair of two 11-year old students (a boy and a girl) of the final year of a primary school of the city of Volos, in Greece were asked to explore a logical model under the supervision of their teacher. The model is based on a scenario of a dog that is in conduct with a live wire and received an electric shock. The children were asked to investigate the conditions under which they could safely rescue the dog. The mode includes attributes like Material of the *stick* to touch the dog, Material of the *shoes* of the child, Material and condition of the *floor* (see figure 4). The teacher asked the students to investigate various alternatives and to check the validity of the model. The session that lasted one hour was recorded and subsequently analyzed using a dialogue annotation scheme.

An interesting finding of the study was that the two children were engaged in dialogue with the LMC environment and discussed their own experiences related to the subject domain. They investigated for instance the conducting capability of materials like plastic and rubber in relation to the shoes and inferred that plastic is insulating material, as in cables of household electric appliances. One of the children recalled

that her grandmother received a strong shock when she touched a bare live cable. The messages received by LMC were considered relevant and supported the specific task. The children seemed to trust the software environment when they engaged in dialogue with it and expressed their wish to further interact with models in other subject domains. Despite the fact that the children of this age group were lacking strong conceptual models of the domain, they managed to reason about it with the support of LMC.

4 Conclusions

In this paper we described the Logical Model Creator (LMC), an innovative environment that permits building and exploring Logic Propositions. The architecture of LMC and the user interface were presented in this paper. The concept of the Reference Model is used for diagnosing the validity of logic models built by students of 11 to 16 years old. A Truth Table is used as interface component for permitting to the teacher to define the valid states of the Reference Model. This is based on all possible values of the Entity Attributes in the "if graph" of the Model. Through this Table the teacher can specify all possible accepted states of the entities of the *decision* and *counter decision graph* of the model. In a case study, involving primary school students, it was found that the environment was intuitive to use and explore, while the messages received by the environment were considered useful in the specific domain.

Finally it should be mentioned that the LMC environment, is useful in addition to exploring models in various subject matters, for introducing young students in concepts of logic, like Boolean operators and IF-THEN-ELSE constructs.

References

1. Bliss J. From Mental Models to Modelling, in H. Mellar, J. Bliss, R. Boohan, J. Ogborn, C.Tompsett (Eds). Learning with Artificial Worlds: Computer Based Modelling in the Curriculum, The Falmer Press, London, 1994.
2. Teodoro, V. D.: Learning with Computer-Based Exploratory Environments in Science and Mathematics. In S. Vosniadou, E. De Corte, H. Mandl (Eds.), Technology -Based Learning Environments, NATO ASI Series, Vol. 137, Berlin: Springer Verlag. (1994) 179-186
3. Soloway E., Guzdial M., Hay K.E., (1994). Learner Centred Design: The challenge for HCI in the 21st Century, *Interactions*, Vol. 1. No 2, April, pp. 36-48
4. Ogborn J. (1990). A future for modelling in science education, Journal of Computer Assisted Education, Oxford, Blackwell Scientific
5. Fidas, C., Komis, V., Avouris, N., Dimitracopoulou, A. : Collaborative Problem Solving using an Open Modelling Environment. In G. Stahl (ed.), Proc. CSCL 2002, Boulder, Colorado, USA, Lawrence Erlbaum Associates, Inc., (2002) 654-655
6. Partsakoulakis I., Vouros G.. Helping Young Students Reach Valid Decisions Through Model Checking. Proc. 3td ETPE Conf., pp. 669-678, Rhodes, Greece, 2002.
7. Teodoro V.D. Modellus: Using a Computational Tool to Change the Teaching and Learning of Mathematics and Science, in "New Technologies and the Role of the Teacher" Open University, Milton Keynes, UK, 1997.