Providing Assistance during Decision-making Problems Solving in an Educational Modelling Environment

Panagiotis Politis[1], Ioannis Partsakoulakis[2], George Vouros[2], Christos Fidas[3]

1 Dept. of Primary Education, University of Thessaly,

38221 Volos, Greece

ppol@uth.gr

2 Dept. of Information and Communication Systems Engineering, University of the Aegean, 83200 Samos, Greece

{jpar, georgev}@aegean.gr

3 Dept. of Electrical and Computer Engineering, University of Patras,

26500 Rio Patras, Greece

fidas@ece.upatras.gr

Abstract In this paper we present the model-checking module of decision-making models in the frame of ModelsCreator, an educational modelling environment. The model-checking module aims to assist young students to construct qualitative models for decision-making problems solving. We specify the decision-making models that may be built and we explain the model checking mechanism. The model-checking mechanism compares the student model with the reference model constructed by the teacher and provides immediate advice to the student to help him create a valid model. So, the model-checking module of a decision-making model aims to facilitate student to structure convincing decisions in the proper situations.

1 Introduction

Modelling is a procedure of describing physical or simulated systems that provides an important means for individuals to examine and understand all the aspects, constraints, characteristics, entities, relations and processes that support the behaviour of every such system. Thus, modelling can be a very strong tool to help young students to appreciate the world and to discover new forms of expression (Becker & Boohan, 1995), (Teodoro, 1994). ModelsCreator is a modelling learn-

ing environment that supports expression of different kinds of models (semi-quantitative models, quantitative models, and decision making models) mostly for students 11-16 years old. In this paper we present the ModelsCreator (Dimitracopoulou et. al., 1997), (Dimitracopoulou et. al, 1999) decision making module (Partsakoulakis & Vouros, 2002).

ModelsCreator decision-making module offers support to students to be able to construct models with or without doubts (expressed by probabilities). The module provides generic techniques for models' validation to successfully assist students discover mistaken features in their models and attain an accord with the instructor. The models meet the requirements of many curriculum subject matters, permitting interdisciplinary use of the modelling process. ModelsCreator puts great emphasis on visualization of the modelling entities, their properties and their relations. Visualization is crucial in supporting the reasoning development of young students and favours the transition from reasoning over objects to reasoning with abstract concepts (Teodoro 1997). 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 one, as it is usually the case.

2 Architecture of the Environment

If M is a model then it can be represent verbal as follow:

$M= \{ E_i, i=1, \ldots, k, R_j, j=1,\ldots, l, A_m, m=1, \ldots, n \}$

where E represent the node entities of the solution, R the relationships connecting them and A the attributes of the entities that participate in the solution.
The decision making models consist basically of entities and relations which connects their attributes. Seven supported types of relations make the environment a very powerful tool for creating and testing decision models. The supported types of relations are the AND, THEN, OR, AND, ELSE and NOT types whereas two types of object notes are supported.

2.1 Structure of Entities

A major concern in the designing phase of the ModelsCreator environment was to provide the ability to end- users to define and manage their own libraries of entities. A component of MC that contributes to its open character is the editor of entities. The end users can define entities and insert them in their object libraries. Properties are assigned to these entities and iconic representations that correspond to the defined properties states, thus providing it with behaviour.

These educational entities provide furthermore interfaces to the COM standard and support the XML semantics thus enabling their integration to the MC envi-

ronment and increasing their manageability, reusability and maintenance. The DTD and the XML structure of an entity are given in the appendixes A and B.

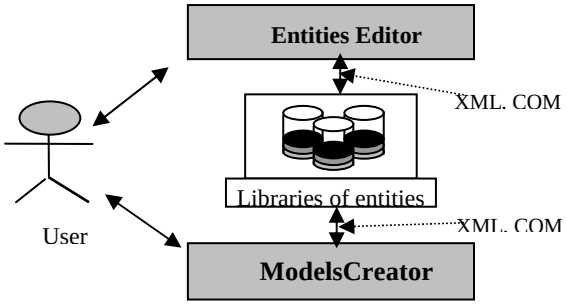

Fig. 1 The entities follow the COM standard and the XML specification

2.1 Constructing Decision-making Models

Models for decision-making are typically qualitative models. Each decision making model has precisely one hypothesis part, precisely one decision part and at most one counter decision part. The student selects certain properties/attributes, sets the desired values and relates them with the appropriate logical connective. For instance, the IF connective applies to an AND expression, while the AND connective, applies to an OR expression and to a property of an entity.

Using such an environment, one may construct fully parenthesised expressions of arbitrary complexity that are according to the following formal grammar:

```
Expression = if Construct then Construct
           | if Construct then Construct else Con-
struct
Construct = (Construct and Construct)
       | (Construct or Construct)
       | not(Construct)
       | Entity_Property_or_Attribute
```

3 Testing a Model

Testing a model means to find out its correctness. But some times a solution to a decision making problem might not be totally true or false. Furthermore there are alternative correct solutions to a decision problem. The model-checking mechanism of the decision-making module aims to facilitate the cooperative process between students and instructors to make an agreement about the situations in which a decision is valid.

To overcome this problem a reference model has been specified for each logical domain (any curriculum subject). This way exist for each logical a reference model which consist of a number of alternative right models which have been specified by the domain owner. In this way a better evaluation can be achieved while testing the correctness of the user model.

To evaluate his model the user has to specify first the logical domain. The logical domain module informs the translation module to set the right reference model. Then the model module informs the translation module which starts preparing the interface for the prolog module. This interface consists of the reference model and the user model in ASCII format. The prolog module checks the reference with the user's model and provides to the user the appropriate feedback.

3.1 The Prolog Model-checking Module

The model-checking module consists of four major steps. First, the two models (student and reference model) are converted using model-preserving formulas in equivalent Conjunctive Normal Forms (CNF). After that, the two models are compared and the results of the comparison are raised intermediately and can be visualized in the form of a comparison table. Mistaken aspects of student's model may be diagnosed by inspecting the comparison table. Finally, the mechanism decides on the appropriate feedback message that should be given to the student

3.1.1 Converting the model

To sustain the fundamental relation between the hypothesis and the decision part of the model, the student-model and reference-model are recorded independently. Each part is then converted using tautologies in CNF. A sentence in CNF is a conjunction of a set of disjunctive formulas (figure 5). Each disjunctive formula consists only of atomic formulas.

$$\overbrace{(a_1 \ddot{\mathrm{I}}\ a_2 \ddot{\mathrm{I}}\ \ldots \ddot{\mathrm{I}}\ a_{n_a})}^{(A)} \Omega \ldots \Omega \overbrace{(z_1 \ddot{\mathrm{I}}\ z_2 \ddot{\mathrm{I}}\ \ldots \ddot{\mathrm{I}}\ z_{n_z})}^{(Z)}$$

Fig 2: The Conjunctive Normal Form

Before comparing the two models the mechanism simplifies the sentences by removing redundant elements (atomic formulas present more than once in the same disjunctive sentence, disjunctive formulas that form part of the whole formula and that imply the whole formula).

3.1.2 Comparing student model and reference model

The two models are equivalent in the case that each disjunctive formula of one of the models is implied by the other one. So, each disjunctive formula is converted to a set of atomic formulas to compare the two models in CNF. For example the formula $a \ddot{I} b \ddot{I} c \ddot{I} d$ is converted to the set $\{a, b, c, d\}$. In other words, each model is converted to a set of sets, where each inner set corresponds to a disjunctive formula and contains atomic formulas.

The comparison process achieved by the model-checking mechanism results the recording of the atomic formulas that are missing or that are surplus in each disjunctive formula in both models. The comparison table reflects the result of the comparison process. Entries of this table correspond to pairs of disjunctive formulas. Each entry (i,j) contains a sub-table with the missing and surplus elements of the i-th disjunctive formula of the student model compared with the j-th disjunctive formula of the reference model.

3.1.3 Judging the student model

If in each row and column of the comparison table there is a sub-table with no missing or surplus atomic formulas, that means that for each disjunctive formula of student's model exists a matching formula in the reference model, so the student model and the reference model are equivalents. If the two models are not equivalents, then the model-checking mechanism can diagnose different situations by inspecting the comparison table.

If the student has over-specified the situations where a decision can be formed, or the student has specified the right properties/attributes for the right entities, but has not assigned the proper values for (at least) one of these properties/attributes that means that the student has missed at least one atomic formula. This formula may correspond to an entity participating to the model, to a property, to an attribute or to a value assigned to a property/attribute of a participating entity.

If the student has under-specified the situations where a decision can be formed that means that at least one atomic formula in student model is surplus. This formula may correspond to an entity participating to the model, to a property, to an attribute or to a value assigned to a property/attribute of a participating entity.

If the student has related two entities with a wrong logical connective, the diagnosis is also based on atomic formulas that are missing and surplus.

All the mentioned cases are not measured the same. A total score is computed that specifies the providing feedback to the student. So, the student model is assigned a score value that is a number between 0 and 100 (Table 1).

Table 1. Situation and score assigned during student model's judgment

Score	Situation
0	Non recognizable error
10	Surplus entity
20	Entity missing
30	Surplus property
40	Property missing
50	Wrong value in property
60	Wrong probability value
70	Connective misuse
100	Equivalent to reference model

3.1.4 Providing feedback to the student

The score assigned during the student model's judgment specifies the feedback message provided by the environment. To each score level one or more messages are assigned with a preference. Messages with lower preference are more or equally detailed.

The purpose of the system is to facilitate the student to assemble his model equivalent to the reference model. The feedback messages should give the suitable assistance students construct their valid models. To attain this aim, the system creates and displays messages of increasing factor.

If the system has diagnosed that the student over-specified the situations for making a decision, e.g. by specifying a surplus entity, it is proposing to the student to confirm the entities in the model. In case that the student does not improve his score, the checking mechanism provides suggestion by prompting the student to ensure if there are any surplus entities in the student model. After that, if the student insists in the same invalid situation, the checking mechanism provides a more detailed assistance by saying that the specific entity is not related to the situation being modelled.

4 Conclusions

The ModelsCreator decision-making module aims to help young students to build qualitative models during decision-making problems solving. Decision making models comprise entities and theirs properties that participate in a decision making state, represented by the student model, to another, correct model, the reference model. The model-checking mechanism compares the student model with the reference model and provides active comment to the student to help him create a suitable model. The model-checking mechanism of the decision-making module

aims to assist the communication between students and teachers to reach an accord about the circumstances in which a choice is convincing.

References

1. Beckett, L., Boohan, R.: Computer Modelling for the Young - and not so Young – Scientist, Microcomputer Based Labs: educational research and Standards, R. Thinker (ed), Springer Verlag, ASI series, Vol 156, (1995) 227-238
2. Dimitracopoulou, A., Vosniadou, S., Ioannides, C.: Exploring and modelling the real world through designed environments for young children. In Proceedings 7th European Conference for Research on Learning and Instruction EARLI, Athens, Greece (1997)
3. Dimitracopoulou, A., Komis, V., Apostolopoulos P., Politis, P.: Design Principles of a New Modelling Environment Supporting Various Types of Reasoning and Interdisciplinary Approaches. In Proceedings 9th International Conference of Artificial Intelligence in Education, IOS Press, Ohmsha, (1999) 109-120
4. Fidas, C., Komis, V., Avouris, N., Dimitracopoulou, A.: Collaborative Problem Solving using an Open Modelling Environment. In G. Stahl (edited by), Computer Support For Collaborative Learning: Foundations For A CSCL Community, Proceedings of CSCL 2002, Boulder, Colorado, Lawrence Erlbaum Associates, Inc., (2002) 654-655
5. Partsakoulakis, I., Vouros, G.: Helping Young Students Reach Valid Decision through Model Checking. In Proceedings 3rd Hellenic Conference on Technology of Information and Communication in Education, Rhodes, Greece (2002) 669-678
6. 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: Psychological and Educational Foundations, NATO ASI Series, Vol. 137, Berlin: Springer Verlag. (1994) 179-186
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)

Appendix A

```
<!ELEMENT Entity(NAME,TYPE,COMMENT,GUID, ICON,
ATTRIBUTE+,ENTITYSTATE+)>
          <!ELEMENT NAME(#PCDATA)>
          <!ELEMENT TYPE(#PCDATA)>
          <!ELEMENT COMMENT(#PCDATA)>
          <!ELEMENT GUID(#PCDATA)>
          <!ELEMENT ICON(#PCDATA)>
          <!ELEMENT ATTRIBUTE
(ID,NAME,TYPE,COMMENT,VALUES, ATTRIBUTESTATE+)>
```

```
            <!ELEMENT ID(#PCDATA)>
            <!ELEMENT NAME(#PCDATA)>
            <!ELEMENT TYPE(#PCDATA)>
            <!ELEMENT COMMENT(#PCDATA)>
            <!ELEMENT VALUES(MIN,MAX,DEFAULT)>
              <!ELEMENT MIN (#PCDATA)>
              <!ELEMENT MAX (#PCDATA)>
              <!ELEMENT DEFAULT (#PCDATA)>
            <!ELEMENT ATTRIBUTESTATE(VALUES+)>
              <!ELEMENT VALUES(ATTRSTATEID,MIN,MAX)>
              <!ELEMENT ATTRSTATEID (#PCDATA)>
              <!ELEMENT MIN (#PCDATA)>
              <!ELEMENT MAX (#PCDATA)>
          <!ELEMENT ENTITYSTATE (ID, ICON
FILENAME,ATRIBUTE+)>
            <!ELEMENT ID(#PCDATA)>
            <!ELEMENT ICON(#PCDATA)>
            <!ELEMENT ATRIBUTE(ATTRSTATELIST+)>
            <!ELEMENT ATTRSTATELIST(ID,ATTRSTATEID)>
              <!ELEMENT ID(#PCDATA)>
              <!ELEMENT ATTRSTATEID (#PCDATA)>
```

Appendix B

```
<?xml version="1.0" encoding="UTF-8"?>
< ---!DOCTYPE Sustem SYSTEM
"../ModellerLibraryElement.dtd">
<Entity Name="ENTITY_NAME" Type="ENTITY_TYPE" Com-
ment="xxx">
          <GUID value="xxxx-xxxx-xxxx-xxxx"/>
          <Icon path="xxx" />
          <Attribute id="xx" Name="ATTR_NAME"
Type="ATTR_TYPE" Comment="xxx">
          <Values max="xxx" min="xxx" default="xxxx">

          <AttributeState >
          <Values AttributeStateID ="xx" max="xxx"
min="xxx" default="xxxx">
</ AttributeState>
          <AttributeState >
          <Values AttributeStateID ="xx" max="xxx"
min="xxx" default="xxxx">
</ AttributeState>
```

```
            <AttributeState >..……..
            </Attribute>
            <Attribute ……
            <EntityState id="xxx">
              <Icon Filename="xxx" />
              <Attribute ID="xxx" AttributeStateID ="xx"
/>
              <Attribute ID="xxx" AttributeStateID ="xx"
/>
              <Attribute…..
            </EntityState>
            <EntityState >……
</Entity>
```