Analysing design problem solutions in learning environments. The DomoSim-TPC approach

M.A. Redondo[1], M. Ortega[1], M.F. Verdejo[2]

[1]Dpto. de Informática
Universidad de Castilla – La Mancha
Paseo de la Universidad, 4. 13071 – Ciudad Real. Spain
{Miguel.Redondo,Manuel.Ortega}@uclm.es
[2]Dpto. de Lenguajes y Sistemas Informáticos
Universidad Nacional de Educación a Distancia
Ciudad Universitaria, s/n. 28040 – Madrid. Spain
felisa@lsi.uned.es

Abstract. Currently, the research about collaborative learning tends to focus on the analysis of the interactions happening in learning activities in order to analyze and identify the cognitive advantages of joint activity. However, most of the efforts are devoted to evaluate the collaborative process, forgetting the products generated. In this paper we present a systematic method to characterize and to evaluate solutions generated as consequence of the resolution of a design problem in an environment of collaborative learning.

1. Introduction

One of the current research focus on computer based collaborative learning is the analysis of the interactions happening in learning activities in order to analyze and identify the cognitive advantages of this approach. We can mention recent representative works which provide some functionality to facilitate the study of the interaction between the users and the system supporting the activity. For example, MFK [1] extracts interaction events, organizing them into categories and carrying out a quantitative analysis whose results are shown by means of tables and graphs. The C-CHENE [2] system records data of the interaction structured and organized around the concept of the communication act. The DEGREE [3] environment proposes a method that combines a quantitative study with a qualitative analysis by means of variables with qualitative attributes and fuzzy inference. The study that this system carries out draws conclusions about the attitude of the users and of the group in relation to collaboration. The EPSILON [4] system records actions and contributions of the participants in an activity in order to identify situations of effective and non-effective learning using Hidden Models of Markov. Using complex rules, Muehlenbrock [5] defines protocols of the user's actions where conflicting situations are identified. These protocols are compared with the sequences of dialogue actions that the participants generate. This comparison is used to determine the degree of likelihood of reaching a conflict situation.

The works mentioned previously centre the analysis on the process of elaboration of the results. The conclusions drawn relate to the communication and the collaborative process of elaboration, but they do not analyze the results in depth.

Our objective is to create a method that could help teachers to organize and interpret the great volume of information that a system can record when supporting collaborative learning of modelling tasks. We use DomoSim-TPC [6], an environment where the students learn to solve a kind of design problems with a computer tool, i.e. they have to build a model satisfying a set of requirements. In this kind of scenario, the system can track information not only about the interaction between students, but also the outcome, i.e. the model they have produced. An interesting issue is to study the relationships between the collaboration process and the quality of the outcome. For this purpose, a qualitative method to evaluate solutions is needed.

In this article we describe a knowledge-base method to qualify the solutions generated by students taking into account the process to achieve them. The following section gives an overview of our system. Section 3 describes the information recorded about the student's activity as well as the knowledge about the problem-solving task. The method to carry out the evaluation is discussed in section 4 and the future works arising from this research are pointed out in the conclusions.

2. The DomoSim-TPC environment

DomoSim-TPC[1] is a distributed environment providing support for distance learning of domotics design. The term Domotics is associated to the set of elements that, installed, interconnected and automatically controlled at home, release the user from the routine of intervening in everyday actions and, at the same time, they provide optimized control over comfort, energetic consumption, security and communications.

The system functionality includes the definition, performance, tracking, task analysis and storage of collaborative activities. The students should learn how to design and build domotics models satisfying some requirements and restrictions. Using tools based on direct manipulation and with the help of the system, they specify the design of the model. The decisions taken in the design can be questioned, justified and argued in group, using a tool of argumentative discussion integrated in the system. Then, the model the group has built can be simulated to study its behaviour and its properties [7].

The activities that the students carry out are organized in two stages:

1. Planning a design strategy. In this stage the students, in an individual way, plan the steps to build a model satisfying the requirements specified in the problem formulation. In this process the user's actions to create a plan are monitored and analyzed by the system using pattern-matching techniques. The plan created by the user is compared to an optimal plan of design for this problem. This optimal plan is built in an automatic way from a generic plan. The method is explained in detail in [8].

[1] http://chico.inf-cr.uclm.es/domosim

2. Discussion, argumentation and search of agreement in the characteristics of the design plans individually built. In this stage, the participants discuss about the models, about their elements, and the steps to carry out in order to build them. From this process a common proposal is obtained reflecting the view point of each participant.

In order to support the above stages, the system provides the concept of workspace. This is a virtual structured place with resources and tools to perform a task. There are three types of spaces:

- An individual workspace to plan a design satisfying a specification. The main tool in this workspace is PlanEdit. It is an adaptive problem-solving tool for design [7].
- A shared space for discussion and argumentation about the design actions that the learners have planned with PlanEdit. This workspace provides support for issue-based conversation and relates the dialogue contributions with the actions of design.
- Another shared space with the final collaborative proposal obtained from the discussion process. In this workspace the *table of contents* metaphor is used. This table contains and organizes design models in which the learners have reached agreement.

Once a solution has been obtained, the teachers have to evaluate its quality. The objective of this work is to present a method to facilitate the evaluation task.

3. Student model and problem-solving task

In order to monitor and guide students, the system builds a student model and uses knowledge related to the problem-solving task. The system also has a collection of problems, where each problem includes a variety of information to automatically generate an optimized instance plan for that problem.

3.1 Student model

The user model captures information about:

- The student's profile that defines the student’s role in the system. This profile stores restrictions and obligations about the type of actions that the student should carry out.
- The user's interaction with the editor in order to plan the design.
- The sequence of actions that the student dynamically specifies in his/her search of a solution to a problem. Each action is recorded together with an annotation generated by the plan pattern-matcher. Additionally, the time dedicated to the elaboration is also recorded. In this way, each element in this sequence is defined by a unique identifier, the moment the action is planned, the mistake associated to it (if there is one) and the action preceding the current action. This is,

```
<item>::= <id><time><action><prevAction>
<item>::= <id><time><action><error><prevAction>
```

where,

```
<action>::= <kind><area><section><element>
<error>::= WRONG_ACTION | SEQUENCE_ERROR
<error>::= DUPLICATE_ACTION | NON_OBLIGATORY_ACTION
```

2.2 Problem-solving model

Design problems with similarities in their solutions can be classified into categories. For each category a general strategy of resolution is defined: the *General Plan of Design* (GPD). This knowledge is currently acquired with an authoring tool, but we are exploring automatic techniques to infer this knowledge from examples solved by experts.

A GPD is a set of expressions in the form:

```
<action>:<type>:<requirements>:<influences>
```

`<action>` identifies an action, `<type>` indicates whether it is obligatory or not, `<requirements>` indicates the actions that should be realized before the current one and `<influences>` references those actions that should be carried out as a consequence of the current action.

A particular problem is defined by a set of parameters. In some cases these are fixed while in others, they are variable. These parameters may condition the solution. In our domain, we have considered five groups: Identity Data, Environment Information, Housing Characteristics, Restrictions and Necessities of the Design, Information, Hypothesis and Simulation Cases. To every parameter in each group, there is an associated rule of the type `IF <condition> THEN <modifyAction>`. This rule can modify the design actions and their structure. So, the system can modify the GPD according to the specification of the parameters that characterize the problem. As a result, the *Optimal Plan of Design* (OPD) is obtained for the proposed problem. This constitutes a resolution schema structured as a graph, with some controlled flexibility in the tasks to carry out.

4. The method to analyze design problem solutions

In DomoSim-TPC an automatic analysis of the design plan generated by the students is carried out in three different stages:

- Dynamically, during the individual elaboration of the models [8], to guide the users toward a good solution.
- During the discussion process, to verify that the model proposed by the group as final solution satisfies the specification of the problem.
- Once the group of students have come to an agreed solution, the outcome is evaluated. To carry out the analysis and evaluation of the final solution

proposed by the students the system compares this solution with the OPD associated to each problem, according to a set of criteria.

Next we detail the criteria and the elements to be taken into account to evaluate a solution.

4.1 Attributes

Some of the attributes we will define are generic for a modelling task while others are dependent of the domain to study.

As generic descriptors we will use the number of objects of the model (or solution), the number of relationships among objects, the parameterization operations carried out with objects or relationships, the design actions that have been planned and the order defined among them.

In domotics, we will study the implications that other factors produce like the number of management areas (comfort, security) and the number of rooms in the house.

4.2 Quantitative analysis

The system performs a quantitative analysis first. This information can be presented by means of tables and graphics in a tool integrated in DomoSim-TPC. The quantitative data is the input to the qualitative evaluation, a rule-based process, producing a classification and a description of the students' solution.

Some of the data to be considered is:

- With respect to the solution:
  - the objects of the model,
  - the relationships among the objects,
  - the parameterization of attributes of the objects.
- With respect to the elaboration process:
  - the design actions carried out to build the model,
  - the order and the relationship of precedence defined among the actions that lead to the construction of the model.
- With respect to the domotics design:
  - the rooms being part of the housing plan on which the problem is formulated,
  - the services that should be automated,
  - the services which should have been studied in the proposed solution but were not done.

4.3 Qualifying a solution

In collaboration with teachers and experts in domotics, we have defined a variety of classes and attributes to characterize a solution. We consider that a design problem solution will be:

- *Valid* if it satisfies the requirements of the specification of the problem or *invalid* if it doesn't.
  Furthermore a *valid* proposal, according to the obligatory character of the included objects and of the design actions specified in the solution, can be classified as *correct* or *incorrect*. If a *valid* proposal doesn't include non obligatory objects then it is a *correct* solution. Otherwise, an *incorrect* solution is a *valid* proposal including non necessary objects according to the problem requirements. We added this category to show whether extra resources are being wasted.
  Also, a *valid* proposal, according to the characteristics of the process outlined for the construction of the solution, can be classified as *well-formed* or as *redundant*.
- According to the complexity observed in the existing relationship between the requirements specification of the problem and the proposed solution, we can determine whether a solution is *difficult* or not. This topic is discussed below.

In order to model the conditions that a solution of domotic design should present to be qualified with some of the previous adjectives we use rules. For example, a solution is valid if when contrasted with the OPD it is observed that:

- There is no lack objects,
- it has not built-in objects that should not be inserted.

And in relation to the procedure to build this solution we can affirm that:

- there is no lack design actions,
- there are not actions considered incorrect
- there are not similar actions planned in a repeated way.

We define:

- *Num* as an overloaded function (its behaviour depends on the type of arguments that it receives) coming from the quantitative study, that counts the number of objects of a certain type appearing in a model or in the OPD,
- *M* as the proposal of the solution designed by the students,
- *P* is the OPD associated to the problem for which *M* is proposed as a solution.

Thus, we can build a rule indicating if starting from *P* we can determine that *M* is a *Valid Solution*. The rule is defined as:

```
IF
            Num(well inserted objects, M) = Num(well inserted objects, P)
    AND
            Num(wrong inserted objects, M, P) = 0
    AND
            Num(replicated objects, M, P) = 0
    AND
            Num(planned actions, M) = Num(obligatory actions, P)
    AND
            Num(wrong actions, M, P) = 0
    AND
            Num(replicated actions, M, P) = 0
THEN
    M IS A VALID SOLUTION
```

In the same vein, the characteristics observed in a non-valid solution could be modelled as follow:

```
IF
        NOT Valid Solution_ (M, P)
THEN
        M IS A NON-VALID SOLUTION
```

According to the complexity observed in the existing relationship between the requirements specification of the problem and the proposed solution, we can determine whether a solution *is difficult* or not taking into account the personal opinion of the teacher carrying out the analysis. For this, we define a parameter named complexity factor of the model (*fc*). It is calculated this way:

$$fc = \sum_{i=1}^{n} Num_i * p_i$$

where *i* is the object type that can be part of a model (domotic operators, relationships among operators, parameterization operations and design actions), *n* is the number of objects types of the model, Num_i is the number of objects *i*-type that there is in the model, p_i is a value between 0 and 1 that makes reference to the complexity that supposes the presence of a *i*-type object in the model.

The complexity factor is a subjective value depending on the definition of each p_i. The observer will be able to define these values according to their own approach. From the complexity factor we calculate the difficulty level of the solution. This is expressed in the following way:

$$complexity = \frac{fc * NumAdmonAreas * fa}{NumRooms * fh}$$

where *fc* is the previously defined complexity factor, *NumAdmonAreas* is the number of administration areas in which the specifications of the problem are distributed, *fa* is a subjective factor about the way in which the number of administration areas influences in the difficulty of the problem, *NumRooms* is the number of rooms that are part of the housing on which the design problem is proposed, *fh* is a subjective factor reflecting the influence of the number of rooms in the difficulty of the solution.

We could think that, when increasing the number of rooms, the complexity of the problem would increase. However, a building is not generally composed of many different types of rooms, as there may be several rooms of the same type. We assume that the design of the automation of services in an administration area is not very different for the rooms of the same type. Therefore, the design should be similar, reducing the global complexity of the problem. Thus, a solution is difficult if its associated level of difficulty overcomes a certain threshold that the observer can define.

In order to synthesize and relate the qualifiers presented above, we visualize them as a graph structure like the one shown in figure 1. Every designed model has a complexity value that will allow us to qualify it as difficult or not. If the condition C_1 is true then the solution will be *Valid*. If it is false (C_2 is complementary to C_1 being true) the solution will be *Invalid*. A *Valid Solution* can be *Correct* if C_3 is true. If C_5 is true then, a *Correct* solution and *Incorrect* solution, can be considered *Well-formed*.

In the same way, a solution can be *Redundant* if C_6 is true. However all this knowledge is represented in a declarative form, so that it can be incrementally and easily changed or adapted.

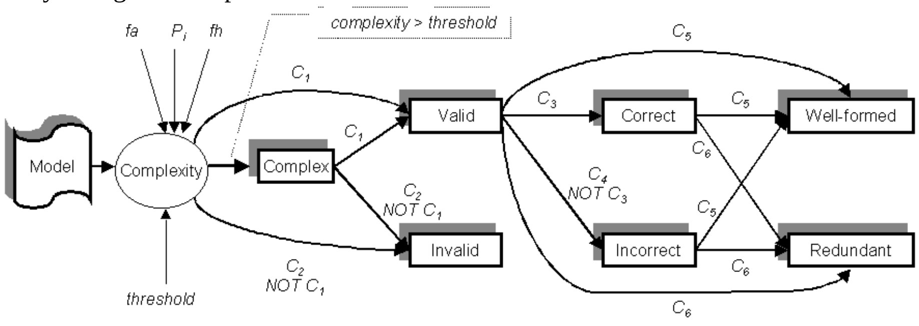
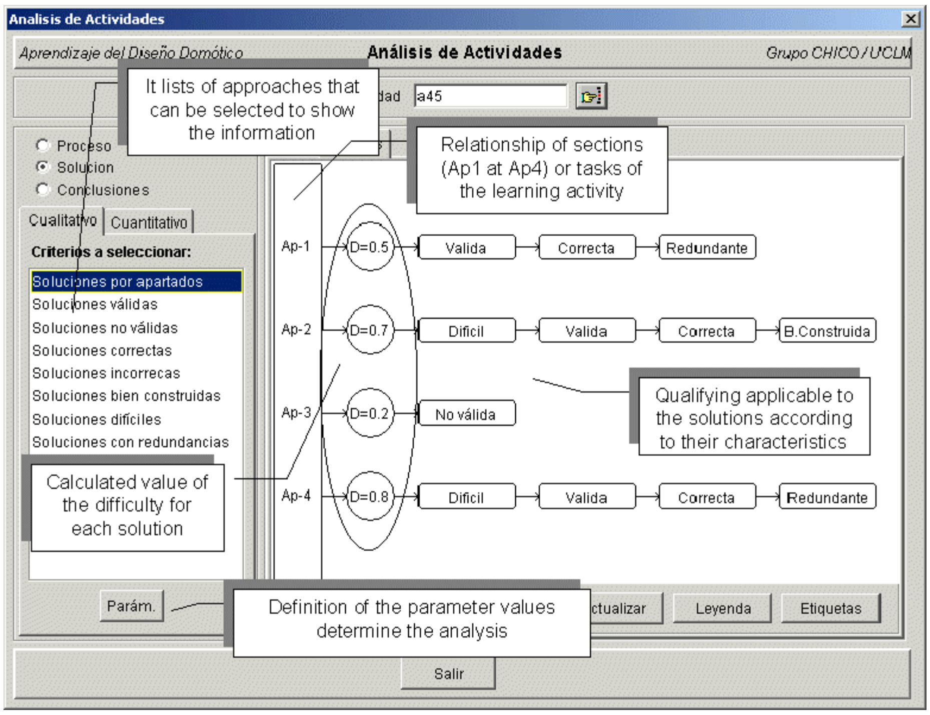


Fig. 1. Attributes observed in a design problem solution.



It lists of approaches that can be selected to show the information

Relationship of sections (Ap1 at Ap4) or tasks of the learning activity

Qualifying applicable to the solutions according to their characteristics

Calculated value of the difficulty for each solution

Definition of the parameter values determine the analysis

Fig. 2. User interface of the tool to draw conclusions.

4.4 The tool to show the conclusions over the solutions

We have designed a tool to give support to obtain conclusions on the solutions the students generate. This tool has been included in DomoSim-TPC and incorporates the necessary functionality to allow an analysis such as the one we have described. The user interface is shown in figure 2. For the selected representation type the characteristics of the solutions constituting the activity are presented organized from left to right.

For example, the solution to the first section presents a difficulty of 0.5, it is valid and correct, although it presents redundancies. The solution proposed for the second section is valid, correct and well-formed, in spite of being a solution with high level of difficulty. However, the solution to the third section is not even valid.

The definition of the factors values and the characteristics depending on the observer are carried out interactively as shown in figure 3, in an easy and intuitive way.

With this approach, the tool is open so that in future works new qualifiers can be included and the rules associated to them can be defined.

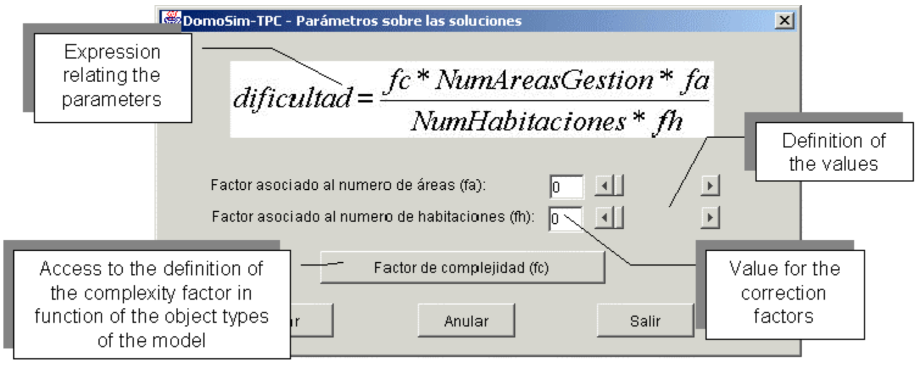

DomoSim-TPC - Parámetros sobre las soluciones
Expression relating the parameters
$dificultad = \frac{fc * NumAreasGestion * fa}{NumHabitaciones * fh}$
Definition of the values
Factor asociado al numero de áreas (fa): 0
Factor asociado al numero de habitaciones (fh): 0
Factor de complejidad (fc)
Access to the definition of the complexity factor in function of the object types of the model
Value for the correction factors
Anular
Salir

Fig. 3. Definition of parameters for the qualitative analysis of the solutions.

5. Conclusions

In this paper we have described the characteristics of the method we propose to evaluate solutions to design problems. These solutions are proposed by students carrying out learning activities in a collaborative supported environment. The evaluation consists in a quantitative and qualitative knowledge-based study that draws conclusions and considerations on the proposed solutions and on the strategy followed to build them. This knowledge is represented in a declarative form.

With this view a tool has been designed and integrated in the DomoSim-TPC system. This system is used experimentally in some schools of "*Formación Profesional*" (Technical Training) in Ciudad Real (Spain). The opinion of the teachers working in these educational centres has been crucial to determine the characteristics that we evaluate and the way we do it.

This method can combine with others devoted to analyze the collaborative discussion and argumentation process that the individuals develop to justify design decisions. The information obtained from the solutions of the problems and with respect to the collaboration provides us with an excellent framework to study the existing relationships between these two aspects. That is, to determine if there are any characteristics of the collaborative process determining a particular type of solutions and vice versa. For example, we will be able to study in an experimental way if a deep discussion process produces very elaborated solutions, causing the approached concepts to be internalized in the mind of the students. These are the future research prospects that arise from this work.

Acknowledgements

The authors would like to acknowledge Junta de Comunidades de Castilla – La Mancha and Castilla – La Mancha University for the support provided to the DomoSim-TPC project which this work is part of.

References

[1] Hoadley, C.M., Hsi, S., & Berman, B.P., (1995). Networked Multimedia for Communication and Collaboration. *Annual Meeting of the American Educational Association. San Francisco*, EE.UU.

[2] Baker, M., & Lund, K., (1997). Promoting reflective interactions in a CSCL environment. *Journal of Computer Assisted Learning*, No. 3, Vol. 13, pp. 175-193.

[3] Barros, B., Verdejo, M.F., (2000). Analyzing student interaction processes in order to improve collaboration. The DEGREE approach. *International Journal of Artificial Intelligence in Education*, 11, pp. 221-241.

[4] Soller, A., Wiebe, J., & Lesgold, A., (2002). A Machine Learning Approach to Assessing Knowledge Sharing During Collaborative Learning Activities. *Proceedings of Computer Support for Collaborative Learning* (CSCL'2002). Boulder, CO, pp. 128-137.

[5] Muehlenbrock, M., & Hoppe, U., (1999). Computer supported interaction analysis of group problem solving. *Proceedings of the Computer Support for Collaborative Learning Conference* (CSCL'1999). Palo alto, CA: Stanford University, pp. 398-405.

[6] Bravo, C., Redondo, M.A., Bravo, J., & Ortega, M., (2000). DOMOSIM-COL: A Simulation Collaborative Environment for de Learning of Domotic Design. *Reviewed paper in Inroads the SIGCSE Bulletin of ACM*, vol. 32 (2), pp.65-67.

[7] Bravo, C., Redondo, M.A., Ortega, M., & Verdejo, M.F., (2002). Collaborative Discovery Learning of Model Design. *Proceedings of Intelligent Tutoring Systems 2002* (ITS2002). Springer-Verlag, in Press.

[8] Redondo, M.A., Bravo, C., Ortega, M., & Verdejo, M.F., (2002). PlanEdit: An adaptive tool for design learning by problem solving. *Proceedings of 2nd Adaptive Hypermedia and Adaptive Web-Based Systems Conference* (AH2002). Springer-Verlag, in Press.