Applying Genetic Classifier Systems for the Analysis of Activities in Collaborative Learning Environments

ANA I. MOLINA,[1] FRANCISCO JURADO,[1] RAFAEL DUQUE,[2] MIGUEL A. REDONDO,[1] CRESCENCIO BRAVO,[1] MANUEL ORTEGA[1]

[1]*Department of Technologies and Information Systems, Computer Science Faculty, University of Castilla—La Mancha, Paseo de la Universidad 4, 13071 Ciudad Real, Spain*

[2]*Department of Mathematics, Statistics and Computer Science, Faculty of Science, University of Cantabria, Avenida de los Castros s/n, 39071 Santander, Spain*

Received 19 January 2010; accepted 21 December 2010

ABSTRACT: The analysis of activities in CSCL (*Computer-Supported Collaborative Learning*) environments can provide us with some interesting conclusions about collaborative learning processes themselves. Specifically, such an analysis can show the effectiveness of such processes and allow for the definition of intervention mechanisms which can motivate and engage the students in the learning activities. Until now, this analysis has focused on the collaboration process and the resulting product separately. We hypothesize that the use of Artificial Intelligence techniques can be useful for the production of a rule-based system that considers both the process and the product. Of the existing techniques, we propose the use of Genetic Classifier Systems (GCS) for their ability to evolve and adapt. The use of these rules allows for the identification and characterization of learning situations, in addition to the generation of feedback that can guide the students and the group towards a more effective learning experience. At the same time, the rule system can adapt to the new learning activities. The analysis method proposed in this article focuses on CSCL environments in which the collaboration takes the form of a conversation. We also present a tool that implements this approach and the results of its application to some learning activities, using an environment for collaborative learning of design. © 2011 Wiley Periodicals, Inc. Comput Appl Eng Educ 21:704–716, 2013; View this article online at wileyonlinelibrary.com/journal/cae; DOI 10.1002/cae.20517

Keywords: CSCL; machine learning; Genetic Classifier Systems; collaboration and interaction analysis

INTRODUCTION

In recent decades there has been an increased effort to study the way in which collaborative systems can facilitate new forms of learning in communities and the effects of these systems from a cognitive point of view. As a result, a research discipline known as CSCL (*Computer-Supported Collaborative Learning*) has emerged [1]. One of the main challenges currently facing the CSCL discipline is the development of computational methods which can automate processes to analyze the interaction between the learners and to provide conclusions about the collaboration. These analysis processes are designed to characterize the actions performed by the learners and evaluating their impact on the learning process [2]. Collaborative learning activities often involve learners working together to solve problems set by a teacher. In such cases, the analysis process should also take into account the solutions built by the learners. The results of such an analysis processes could be used for various purposes, such as providing an evaluation of the learning process, offering advice to the learners that could guide their activities, or providing data that could enable software developers to understand how the collaborative system is being used.

In our research, we focus on a specific kind of CSCL environment, which supports collaboration by means of a conversation-based discussion [3]. Some systems propose the use of *sentence openers* in order to help structure the conversation. These elements make it easy to identify what a user's intention is when he/she makes a contribution to a conversation. This structuring approach facilitates the analysis of activities because it enables each communicative act to be categorized. However,

Correspondence to F. Jurado (francisco.jurado@uclm.es).

the collaboration analysis process should be completed with techniques which calculate variables describing both the learners' working process and the solution built in the end. Moreover, it is necessary to use computational methods that take advantage of this description or characterization in order to intervene in the collaborative activity and help learners with their activities.

In this article, we present the hypothesis that it is possible to use methods and tools based on Artificial Intelligence (AI) techniques to analyze collaborative activities whilst considering the characteristics of both the process and the solution, with the aim of identifying and characterizing specific situations that can take place during a learning activity. Very few proposals have focused on a comprehensive analysis of the collaboration process and the product obtained as a result of this process. One of them is described in Ref. 4. This study proposes a framework for guiding the process and solution analysis in collaborative learning environments. However, as its authors point out, the framework lacks support for the configuration and calibration phase of the analysis and, in particular, for obtaining the inference rules used during the analysis. In this article, we address the challenge of finding a method to meet this requirement. After studying the AI techniques currently in existence, we opted to use Genetic Classifier Systems (GCS) due to their ability to evolve. The use of GCS allows for the production of a set of rules that are able to adapt to the particularities of the different learning activities that take place in a CSCL environment. Once the learning situations have been identified and classified according to this set of rules, intervention mechanisms can be provided in order to guide the learner or the group towards the most suitable path of action or to increase their motivation. The proposal developed in this study has been implemented within an intelligent subsystem, which manages the information stored in a knowledge database. This database will evolve over time, increasing its potential to characterize new situations. This article presents this proposal together with the results of some experiments that have already been carried out.

The article is organized as follows: the next section reviews the main research proposals that have attempted to automate the analysis of collaboration and interaction in learning systems; third section briefly describes the context in which our proposal is applied; fourth and fifth sections describe the proposed analysis method and the tool which implements it, respectively; sixth section discusses the results of an experiment in which the implemented subsystem has been used; and last of all, we present some conclusions and indicate future work.

RELATED WORK

In this section we review some systems and approaches that address the analysis of computer-supported learning activities. In CSCL analysis, the recording of interactions in log files is a widespread technique. These log files store the content of the messages exchanged by learners when communicating with each other and coordinating themselves during the collaborative activity. However, these log files do not usually include descriptive information that categorizes the purpose of each message (to ask, to propose, to answer, etc.). Lund et al. [5] propose the use of structured user interfaces to categorize the type of contribution that each participant makes to the dialog process. A paradigmatic example is the C-CHENE system [6], which includes a structured chat with closed sentences (e.g., "*I do not know*"). Other messages are semi-open (e.g., "*I Propose to*…") and are designed to be completed by the learners. The use of both semi-open and closed messages allows for discrimination of the type of message used and an analysis process of the learners' conversational acts possible. EPSILON [7] stores the actions and contributions made by learners whilst they attempt to solve Object-Oriented Design problems as a group. This information is used in an automated study that analyzes not only the problem-solving actions but also the communication process. The McManus and Aiken approach [8], implemented in the Group Leader system, compares the learners' conversation act sequences with the most highly recommended sequences (which are represented using finite state machines). This system analyzes the dialog sequences and provides the users with advice during their interaction. The DEGREE system [9] analyzes the characteristics of group work and assesses the effectiveness of the learning process with AI techniques based on fuzzy logic that process data about the use of communicative acts in order to characterize the learning process. To achieve this goal, DEGREE combines a quantitative study with a qualitative analysis and proposes a set of predefined analysis indicators (*initiative*, *creativity*, *elaboration*, *conformity*, etc.). Puente [10] proposes the application of statistical methods to analyze the students' contributions in group editing of wikis. None of the systems mentioned above, however, focuses on studying the interrelation between communication and argumentative discussion or the result of these processes.

We find some systems which focus on analyzing learners' solutions to set problems in the area of ITS (*Intelligent Tutoring Systems*). Belvedere [11] evaluates the consistency of diagrams that represent scientific inquiry processes. KERMIT [12] proposes database modeling problems to be solved by learners, and evaluates the quality of the solutions they produce. Collect-UML [13] adds supports for analyzing solutions using UML class diagrams to a collaborative environment. However, these approaches do not compare the solution's evaluation with the quality of the building process.

We can also find several theoretical frameworks for collaboration analysis in the literature [14,15]. Of particular interest is the framework proposed by Bravo et al. [4], which is the only comprehensive approach that combines an analysis of actions and dialog with an analysis of the solution.

Considering the various approaches presented above, we conclude that there is a clear need for computational methods which allow for an analysis of the recorded information in order to reach useful conclusions about learning activities. Some authors propose the use of computational methods as useful techniques for the analysis of discussion processes. Among these techniques, we find Hidden Markov Models [16], Decision Trees [17], Petri Nets [8], and Plan Recognition [18]. Other proposals have also used Bayesian Networks to analyze only the collective process [19] or the resulting products [20], but they do not address both aspects jointly. Techniques based on Genetic Algorithms and Classifiers have been applied in the field of learning systems. Therefore, these techniques have been used to intervene in students' activity by adapting the user interface [21], to select the problems to be solved according to the knowledge level of the students [22], or to personalize the e-learning system according to preferences of the students [23]. However, these proposals apply genetic algorithms and classifiers to the work of individual students, since they are not aimed at working with the collaborative process.

In summary, we can see how CSCL systems apply techniques to analyze students' group working process. The ITS include features that analyze the solutions built by learners to solve problems. However, there is a lack of flexible approaches which take into account both the students' collaborative activity and the results of the collaboration in the form of a solution, exposing a gap in the analysis research carried out to date. At this point, we propose the use of Genetic Classifiers to analyze both aspects (collective work and solutions) and to intervene actively in the learning process. The proposals that have used this type of technique show how it is particularly useful for intervention purposes thanks to its ability to evolve.

THE DomoSim-TPC ENVIRONMENT AND PlanEdit TOOL

The analysis method we propose is oriented towards CSCL activities that promote the learning of design by the solving of problems. The solutions to the problems are built following a process of Collaborative Planning of Design (CPD) [24]. This is an asynchronous collaborative process in which the learners use an intermediate and abstract representation to express the solution to a design problem. This representation is the plan of actions that must be carried out in order to build the solution. The plan is designed collaboratively by a group of learners. This collaborative process is implemented by means of argumentative discussion techniques that are described in [25]. In the CPD process, the learners build design plans individually. They must then show their plans to the other members of the work group, justifying and discussing their design actions. Each group member can make critiques, comments, suggestions for improvement and even counterproposals that could imply a radical change in the initial design plan. Therefore, the plans, initially built in an individual way, evolve to reflect the point of view of each participant in the process.

Collaborative Planning of Design

The analysis method we propose is designed for evaluating learning activities in application areas in which the CPD approach is applicable. Specifically, we take as our basis the learning domain of domotics installation design for automated services in homes and other buildings. This is a complex domain in which to validate the results of our research. In this context we have developed a group learning system called DomoSim-TPC [26]. This system includes a tool called PlanEdit [27] which provides support for CPD. The user interface of PlanEdit (Fig. 1A) is divided into a number of separate areas: the problem formulation, the list of tasks to carry out, the icon bars representing design actions/operators, the sequence of design actions already planned, the current action being carried out, and a set of buttons used to support several general functions. The design actions which the student must choose are displayed in the user interface as icons on toolbars. They are grouped into four categories according to the components of an action: the kind of action, the management area, the plan of the house, and the domotical operator. Figure 1B shows the user interface of the discussion and argumentation tool in DomoSim-TPC. In the middle is a hierarchical structure in the form of an inverted tree which is used to organize and relate the contributions. On the right, various interaction buttons are presented. We have highlighted those buttons designed to manage the information shown in the process diagram and those meant to facilitate the posting of contributions and, in this way, make dialog between the participants possible. These buttons are enabled or disabled depending on aspects such as what kind of contribution has been selected and the characteristics of the dialog defined by the coordinator (teacher).

Analyzing Collaborative Learning Activities

The general model of activity analysis in the DomoSim-TPC system is shown in Figure 2. The processed information has two different origins: the organization of the activity and its realization. With regards to the organization of the activity we consider some general characteristics, the proposed problem to be solved, and the optimal strategies that shape its solution. Each problem consists of several tasks to be solved. Each task can be assigned to one of the activity's participants. Each task has an associated design plan, proposing the best strategy for completing the task (the so-called *Optimal Plan of Design*, or OPD). These plans are compared with the solutions built by the stu-

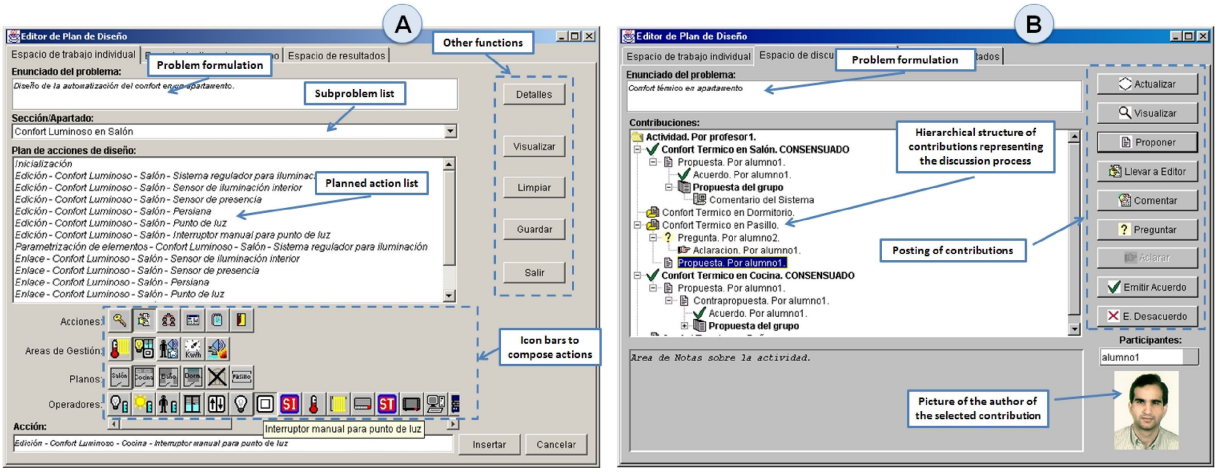


Figure 1 (A) PlanEdit user interface. (B) User interface of the discussion and argumentation tool. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

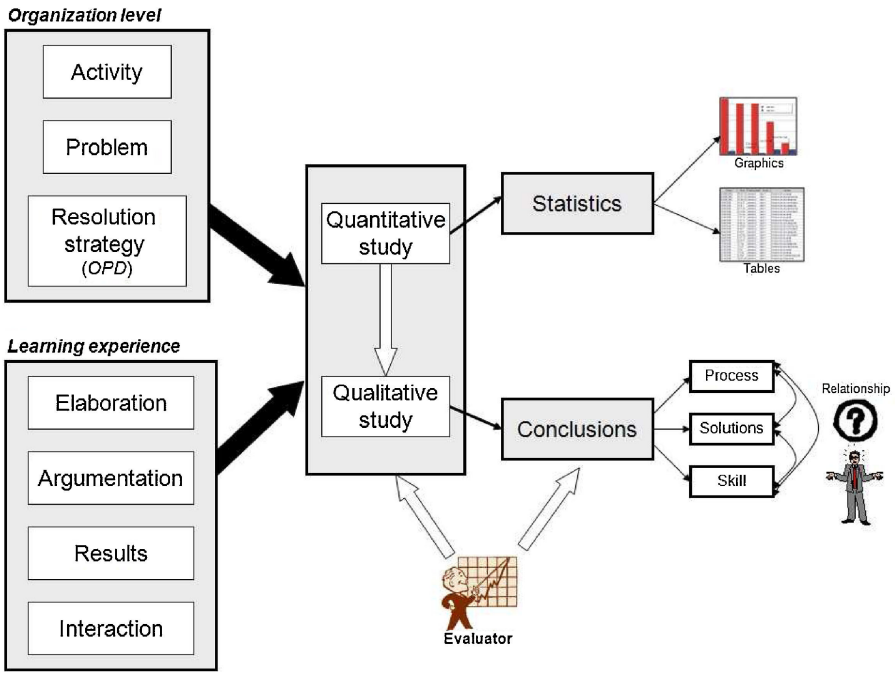


Figure 2 General outline of the framework for the analysis of experiences in DomoSim-TPC. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

dents. In this way, the analysis subsystem can automatically determine and evaluate the differences between them. The information related to the learning experience consists of the data that are registered when the users interact with the system. Specifically, these data come from the individual elaboration of models using the PlanEdit tool, which is described in Ref. [27], from the discussion and argumentation process, and from the table of contents (results) generated at the conclusion of the activity. Some general data about the how the participants have used the system are also taken into consideration.

The previously mentioned information is processed in a quantitative study, in which the information is organized and statistics reflecting an evaluation of particular aspects of the realization of the activity are generated. This evaluation is presented in the system in tabular and graph form. In addition, we can export the data for processing in other, more specific tools. This quantitative study is complementary to another, qualitative one, in which rules and fuzzy logic are used to model other characteristics. Therefore, with the statistical information generated earlier and considering the evaluator's approaches while carrying out the analysis, some conclusions about the learning activity can be inferred. These conclusions are classified into those related to the elaboration and discussion process, those related to the built solutions, and those related to the level of skill reached by the participants. With these conclusions and with all the information obtained previously, finding some relationships between them should not be difficult.

PlanEdit incorporates a subsystem for monitoring and analyzing learning activities (the *Learning Activities Monitor*). This allows the interaction within the work group to be visualized and analyzed. To do so, several aspects of the collaboration are modeled using indicators represented by linguistic variables. These variables are used to obtain conclusions in a fuzzy logic-based inference process [28]. This process is based on the method proposed by DEGREE [9]. In addition, the tool includes another subsystem to analyze the products that the learners build (the *Product Analyzer Subsystem*). This analysis is carried out by applying pattern matching techniques that interpret and contrast the learners' plans with ideal models represented by expert knowledge [29]. As such, the framework provided by PlanEdit is a valid context for the study of the relationships between the collaborative process that is followed in order to build a product (design plan) and the product itself. In the following sections, we describe our approach to collaborative analysis based on GCS and how it has been implemented in PlanEdit.

A METHOD BASED ON GENETIC CLASSIFIER SYSTEMS FOR ANALYSIS OF COLLABORATIVE LEARNING ACTIVITIES

In this section, we present our proposal based on the application of AI techniques to process and solution analysis in CSCL systems. As we have commented above, the proposed analysis method is oriented towards those CSCL activities which promote the learning of design by problem solving (in particular following a process of *CPD*) and provide support for collaboration by conversation-based discussion. The analysis method and the tool that implements it have been developed in accordance with the guidelines proposed by Bravo et al. [4]. In particular, our method is oriented towards the configuration and calibration tasks that are found in the abstraction phase of the analysis framework presented by these authors. Next, we will briefly explain the foundations of the proposed analysis method, that is, the GCS, and the adaptation of this technique to our purpose of classifying learning scenarios.

Analysis Method Foundations: Genetic Classifier Systems

A classifier system (CS) is a machine learning system which learns syntactically simple string rules (called classifiers) [30,31]. A classifier system takes its name from its ability to learn how to classify messages from the environment into general sets and is similar in many respects to a control system. Just as a control system uses feedback to *control* or *adapt* its output for an environment, a classifier system uses feedback to *teach* or *adapt* its classifiers for an environment. A classifier system has three main components: (a) rule and message subsystem, (b) apportionment-of-credit subsystem, and (c) a classifier discovery mechanism (primarily the *genetic algorithm*). The concepts and elements that define the operation of a GCS are as follows:

The *rule and message subsystem* of a classifier system is a special kind of production system. A production system is a computational scheme that uses rules as its only algorithmic device. The *rules* are generally of the following form: *if* <*condition*> *then* <*action*>.

A classifier is a production rule with this syntax: <classifier> ::= <condition>:<message>.

A *message* (or *tuple*) within a classifier system is simply a finite-length string over some finite alphabet. If we limit ourselves to a binary alphabet we obtain the following definition: <*message*> *::= {0,1}*. Here the symbol "*::=*" means "*is defined as*." Messages are the basic token of information exchange in a classifier system.

Once a classifier's condition has been matched, that classifier becomes a candidate to post its message to the message list on the next time step. Whether or not the candidate classifier posts its message is determined by the outcome of an activation *auction*, which in turn depends on the evaluation of a classifier's value or weighting. The auction permits suitable classifiers to be selected to post their messages. Although there are a number of ways of doing this, the most prevalent method incorporates what Holland has called a *bucket brigade algorithm*. To participate in the auction, each classifier maintains a record of its net worth, called *strength*. The strength portion of the classifier gives a measure of the rule's past performance in the environment in which it is learning. That is, the higher a classifier's strength the better it has performed and the more likely it will actually be used when the condition matches an *environmental message* and to reproduce when the *genetic algorithm* (GA) is applied.

The messages may match one or more classifiers or string rules. The "#" symbol acts as a wild card or "*don't care*" in the condition. This allows for more general rules. The more "#" symbols, the more general the rule. The measure used to quantify this is called: *specificity.* The *specificity* of a classifier is the number of non "#" symbols in the antecedent.

The *apportionment of credit* subsystem deals with the modification in strength of classifiers. As the classifier system receives messages from the environment, all the classifiers which match one (or more) of the messages compete, by submitting a *bid*, in an *auction* to determine a victorious classifier that will affect the environment. The victorious classifier's modification will be beneficial or detrimental to the environment. With this feedback, the apportionment-of-credit system appropriately uses *reinforcement and punishment* to increase or decrease the strength of the victorious classifier that caused the modifications. Finally, *taxation* is levied on each classifier per iteration and on each classifier that submits a bid during an auction.

Last of all, after a specific number of cycles of execution of the classifier system, an *evolutionary genetic procedure* is started up to discover new classifiers from those with greater strength and to update the population of rules. The present strength of each classifier allows the classifier to compete with the others in the genetic algorithm in order to obtain the best population of rules, that is, those that best define the concept.

The Analysis Method

As we have mentioned above, our analysis method follows the framework for process and solution analysis proposed in Ref. 4. According to this framework, the interaction and collaboration analysis processes are usually carried out in a three-phase life cycle: *observation*, *abstraction*, and *intervention*. The *observation* phase captures raw data that describe the learners' work (proposals of each learner, answers to questions set by the teacher, etc.). The *abstraction* phase calculates indicators that analyze the collaborative learning [32]. These analysis indicators are quantitative or qualitative variables that characterize various aspects of the collaborative learning process (flow of communication, level of coordination, speed of work, quality of solutions, etc.). Finally, the *intervention* phase uses the analysis indicators to improve the collaborative work by means of actions such as advising the learners on how to correct the deficiencies detected by the analysis indicators.

The first stage of our analysis method therefore consists of the identification of indicators for both the process and the product. Each learning activity is described using a set of analysis indicators that characterize the learning process and the products obtained as a result of it. According to Dimitrakopoulou [33], an analysis indicator gives information about either the quality of the individual activity, the mode of collaboration, or the quality of the collaborative product. In the context of the argumentative discussion for the CPD (and based this decision on previous experiences described in Ref. [29]) we consider the following set of analysis indicators. The first three indicators are related to the process, while the last two describe characteristics of the product:

Modeling: Referring to the contributions that make the design plan evolve.

Discussion: Representing the argumentative discussion between the group members to justify the design decisions taken to solve the set problem.

Cohesion: Providing an assessment of the group, indicating if their members have worked collaboratively or individually.

Solution Validity: Indicating whether or not the specified requirements have been met.

Solution Difficulty: A solution is considered difficult when the difficulty level associated with it exceeds a certain threshold, as defined by the evaluator.

Table 1 shows these indicators and the types of values they can take. The values are expressed in linguistic terms. The indicator's type is also indicated, showing whether it is related to the collaboration process or to the product. The values of

Table 1 Indicators That Characterize the Process and Product of a Collaborative Learning Activity

Indicator	Type	Values
Solution validity	Product	Not Valid (NV)/Valid Incorrect (VI)/Valid Redundant (VR)/Valid Correct (VC)/Valid Well Constructed (VWC)
Solution difficulty	Product	Yes (Y)/No (N)
Modeling	Process	Low (L)/Medium (M)/High (H)
Discussion	Process	Low (L)/Medium (M)/High (H)
Cohesion	Product	Low (L)/Medium (M)/High (H)

these indicators are obtained by executing the subsystems to monitor and analyze learning activities in PlanEdit. This corresponds to the *observation* and *abstraction* phases of the analysis framework described in Ref. 4.

The next stage consists in defining an initial set of *classification rules* for collaborative activities which will allow for the characterization of significant situations. These rules can be contributed by an expert, obtained by applying a non-supervised learning process or identified by means of a statistical study that analyses the correlation between the indicators. The classification validity of the initial set of rules is not relevant, since this set will be refined and adapted to the new situations that can emerge during the use of the CSCL environment. The next stage of the method develops the process of adapting the rule system to the CSCL environment by means of the GCS. The details of this algorithm can be consulted in Ref. 34. We adapt the concepts handled by the GCS to our scenario as shown in Figure 3. The fundamental elements of this adaptation are as follows.

As explained above, *tuples* are used to define situations or *events* (or *messages*) as well as the precedents of the classification rules. In our case they are formed by the values that the analysis indicators take (*Solution Validity*, *Solution Difficulty*, *Modeling*, *Discussion*, and *Cohesion*). The values of these indicators are obtained by executing the subsystems to monitor and analyze learning activities in PlanEdit (Fig. 3A). In Figure 3B we can see an example tuple. In this example, the value of each element in the tuple is (*Valid Correct*, *Yes*, *High*, *Medium*, *High*) or, in code form (*VC*, *Y*, *H*, *M*, *H*).

A *classifier* (c_i) is each of the rules that allow us to define a concept C (in this case, a possible classification or characterization of a learning setting or a collaborative learning activity). Each classifier consists of an assignment of possible values for each of the indicators shown in Table 1 (Fig. 3C). The symbol # is used to indicate that a value is not set. Each concept is expressed formally in predicate logic in the form $c_1 \vee c_2 \vee \ldots c_r : C$. Therefore, an example of a classification rule could be:

$$(\text{Valid Correct} \vee \text{Valid Well Constructed}, \#, \#, \text{High} \vee \text{Medium}, \text{High} \vee \text{Medium}) : \text{Effective Learning}.$$

Which can be written in code form as:

$$(\text{VC} \vee \text{VWC}, \#, \#, \text{H} \vee \text{M}, \text{H} \vee \text{M}) : \text{Effective Learning}$$

This rule allows us to define a scenario of *effective learning*. An expert considers that this kind of scenario has been produced when the solution obtained is *valid correct* (*VC*) or *valid well constructed* (*VWC*) and the process is characterized by a *medium* (*M*) or *high* (*H*) level of *discussion* and *cohesion* in the group. As an action of the rule the expert can define a warning message to be displayed when situations of this kind are detected. For example:

$$(\text{VC}, \#, \#, \text{H}, \text{H}) : \text{Welldone. Carryon.}$$

The *Michigan Approach* [Holland] is used for coding the classifiers. This notation includes fixed-size classifiers,

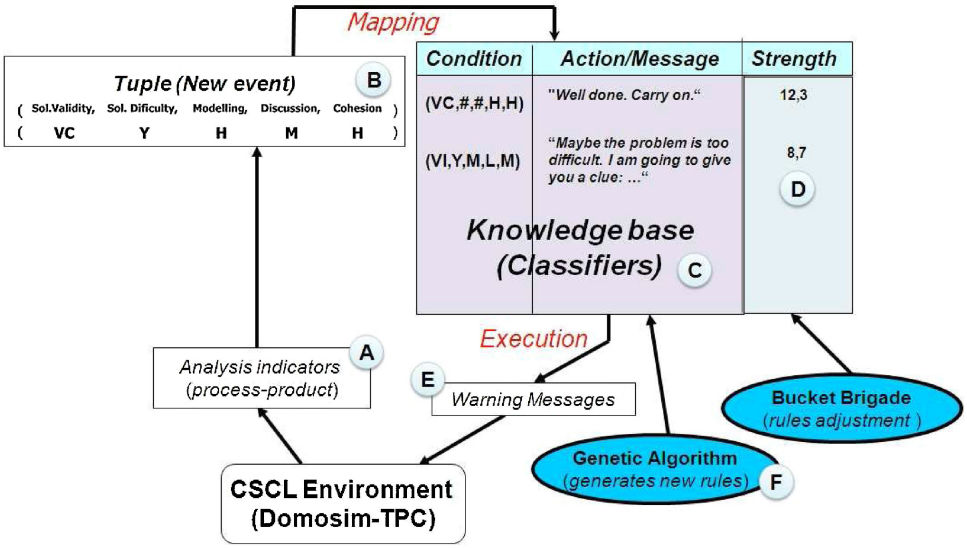


Figure 3 Overall schema of the elements involved in the execution of the proposed analysis method. (A) Analysis Indicators (process-product); (B) Tuple (New event); (C) Knowledge base (Classifiers); (D) Strength; (E) Warning Messages; (F) Genetic Algorithm. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

which facilitates the computational processing (Fig. 3C). The example above can be expressed as follows:

$$(\mathrm{VC}, \#, \#, \mathrm{H}, \mathrm{H}) \vee (\mathrm{VWC}, \#, \#, M, M) \vee \ldots : \mathrm{EffectiveLearning}$$

Each classifier has an associated *strength* which measures its ability to define a concept (Fig. 3D).

Every time a situation is classified by rules from the *knowledge base*, the system will respond with a warning message to attract the learners' attention and to guide them towards a better next step in the problem solving process (Fig. 3E).

Finally, after a specific number of learning activities, a *genetic algorithm* is started in order to update the population of rules (Fig. 3F). In our case the *selection function* is made by drawing lots and the *replacement* is made by *crowding factor*, since we are not interested in finding the best rule but in obtaining a complete population that carries out the best situation recognition work possible.

To illustrate the operation of the GCS, we present a simple example. Let us consider that the system must learn to determine whether or not the social process of solution development is following a path that is considered effective. We will assume that, in a given instant, the system has the classifier population shown in Table 2.

In a certain instant t the system detects that the current situation is characterized by a tuple (*message*) of the following type (*VC*, *Y*, *H*, *M*, *H*). This description of the current situation indicates that the partial solution found so far is correct (*VC*) and difficult (*Yes*), existing an amount of modeling that can be considered high (*H*), a medium degree of discussion or argumentation (*M*), and a high group-cohesion (*H*). As we can see, the identification of the current scenario (the arrival of the new message) triggers the activation of the classifiers q_1, q_2, and q_4. These classifiers will compete to send their messages via an auction process. With the matching classifier pool determined, the auction commences. Each classifier participating in the auction submits a bid that is a function of the classifier's strength (second column in Table 2) and specificity. Only the winning classifier's bid is paid, so only the winning classifier has its strength decreased by the same amount as the winning bid.

The proposed analysis method, which is automatic, can be complemented by the contributions that an *expert* (or *trainer*), acting as an evaluator, can make, largely with regards to the classification of significant settings. The expert's intervention helps to penalize or fortify the various classification rules that the method can generate. When the winning classifier has a beneficial effect upon the environment, the trainer sends positive feedback, causing the winning classifier's strength to increase. Conversely, a detrimental effect leads to punishment. Since the winning classifier's strength decreases when it wins the auction and pays its bid, whilst punishment is implicit whenever a reward is not given. In addition, an adjunct strength reduction may occur. If the trainer has the ability to rank environmental effects, then the rewards and punishments can be scaled appropriately.

Table 2 Sample of Valid Classifiers

Rule	Strength
q_1 = (#, #, #, #, H): Effective Learning	22.1
q_2 = (VC, #, #, M, #): Effective Learning	12.3
q_3 = (#, #, L, #, L): Non-Effective Learning	9.3
q_4 = (VC, Y, #, #, #): Effective Learning	4.2

A TOOL FOR SUPPORTING COLLABORATIVE LEARNING ACTIVITY ANALYSIS METHOD

In this next section, we provide some details about the tool developed to implement our method and the architecture that describes how we have incorporated this analysis tool into our initial CSCL environment (DomoSim-TPC and PlanEdit tool).

The Analysis Tool

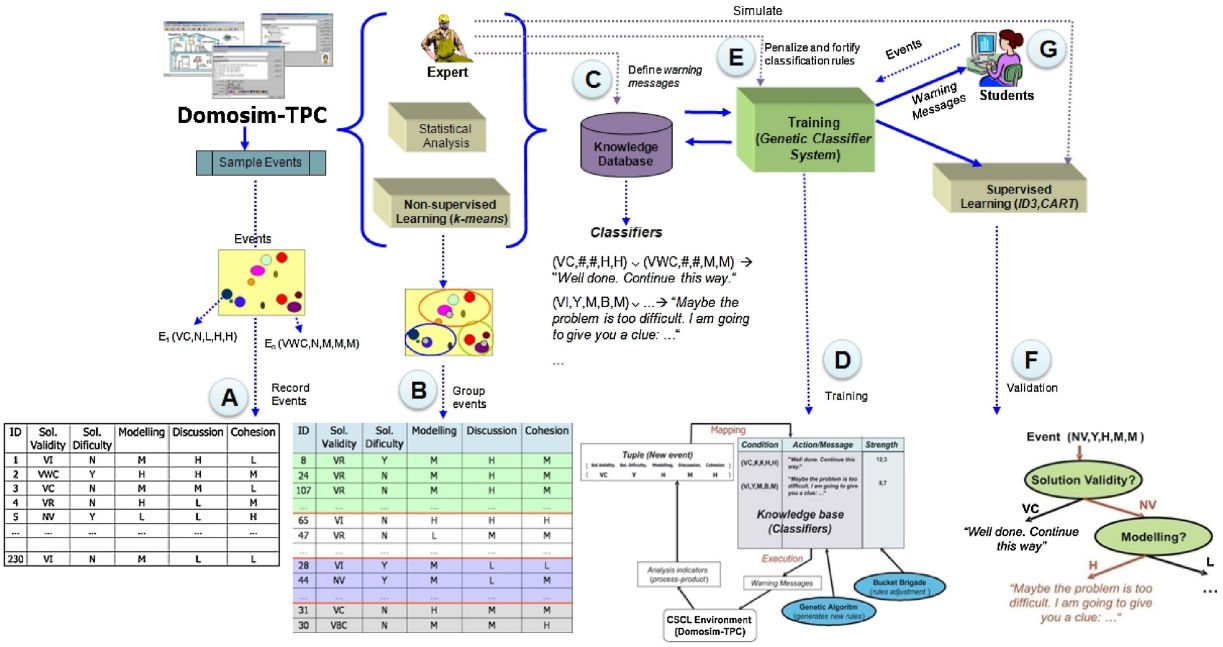
This section presents the tool for process-product analysis. In Figure 4 we can see the process that the analysis tool supports. It is made up of several stages.

The first stage (Fig. 4A) consists of recording of events from the CSCL system (in our case, DomoSim-TPC). Each event is formed by the values taken by the aforementioned analysis indicators (*Solution Validity*, *Solution Difficulty*, *Modeling*, *Discussion*, and *Cohesion*). These events are recorded in a database. Using this initial set of events we can obtain the initial rule base. This can be created from a statistical correlation analysis, from the opinion of an expert (who would enter the rules according to his/her experience), or from a non-supervised learning algorithm (which groups events and infers classification rules) applied to an initial set of learning activities (Fig. 4B). The tool we have developed implements a clustering algorithm, which uses the *k-means* technique [35] to obtain the initial set of rules. In this way, we obtain the initial set of classifiers (specifically, the antecedents of the classification rules).

Once the grouping of events (antecedents of the classification rules) has been done, the expert can define the action to be triggered, in this case, the warning messages to show when a scenario is classified (Fig. 4C). Once the initial set of classifiers has been defined, the GCS, described in the The Analysis Method Section, can operate (Fig. 4D). As mentioned above, the expert intervenes in the training of the classification rules, penalizing and fortifying them (Fig. 4E).

When the knowledge database has been trained, the method allows the operation of the analysis tool to be validated and simulated (Fig. 4F). To do this, classification algorithms can be used (e.g., the ID3 and CART) [36]. The expert can introduce new events (definition of new scenarios) and observe the behavior of the classifier system and the outcome. When the expert has checked the performance of the analysis tool, this can be included in the initial system. We suggest this be done using an architectural proposal, as described in the Integration Into DomoSim-TPC Environment Section. In this way, the students interact with the system producing new events and receiving the warning messages defined by the expert (Fig. 4G). The GCS, implemented by the analysis tool, allows the knowledge database to adapt and evolve in order to respond better to new events or learning situations.

Figure 5 shows the main areas that make up the analysis tool's user interface. The analysis can be carried out during the course of experiences as well as after them. The user can choose from a set of predefined classifiers, or he/she can create a new set of classifiers (Fig. 5A). In the latter case, the top right-hand area of the user interface (Fig. 5B) will be enabled, allowing the



ID	Sol. Validity	Sol. Dificulty	Modelling	Discussion	Cohesion
1	VI	N	M	H	L
2	VWC	Y	H	H	M
3	VC	N	M	M	L
4	VR	N	H	L	M
5	NV	Y	L	L	H
...	...	...	...	...	...
230	VI	N	M	L	L

ID	Sol. Validity	Sol. Dificulty	Modelling	Discussion	Cohesion
8	VR	Y	M	H	M
24	VR	N	M	H	M
107	VR	N	M	H	M
...	...	...	...	...	...
65	VI	N	H	H	H
47	VR	N	L	M	M
...	...	...	...	...	...
28	VI	Y	M	L	L
44	NV	Y	M	L	M
...	...	...	...	...	...
31	VC	N	H	M	M
30	VBC	N	M	M	H

Figure 4 General operating scheme of the collaborative learning activities analysis tool. (A) Record events; (B) Group events; (C) Define warning messages; (D) Training; (E) Personalize and fortify classification rules; (F) Validation; (G) Students using the system. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

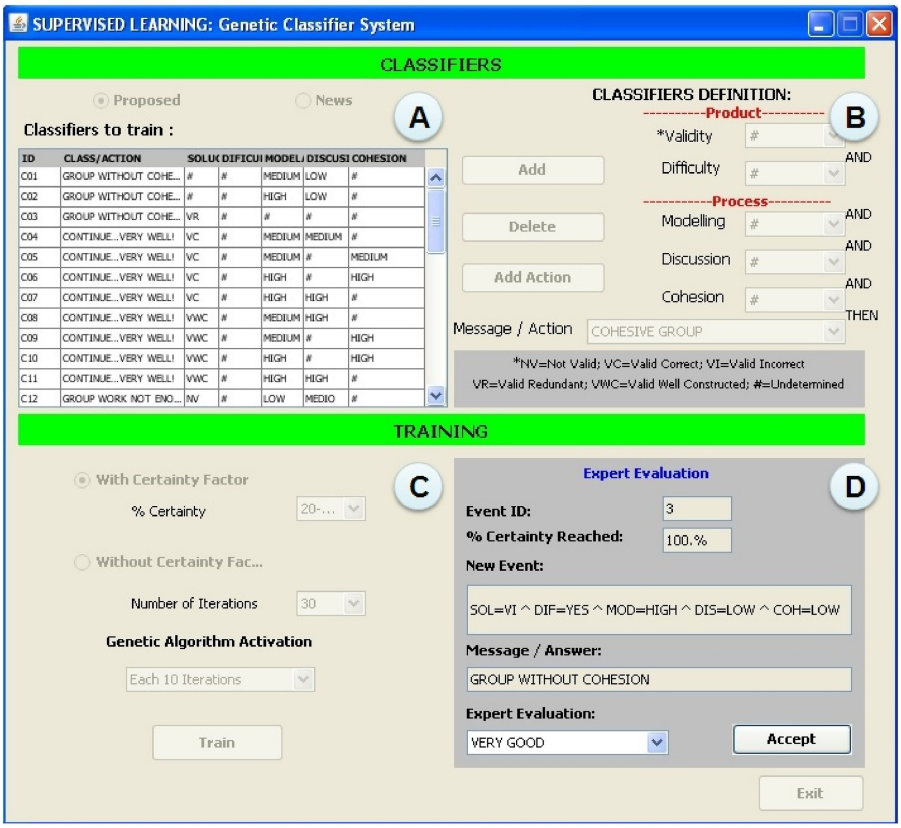


Figure 5 User interface of the application, allowing the expert knowledge based to be trained to classify collaborative learning experiences. (A) Classifiers to train; (B) New classifiers definition; (C) Parameters of the training algorithm; (D) Expert Evaluation. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

user (the expert) to define the values of the indicators for each rule and the action to be triggered when the classifier is activated. The tool includes a number of predefined warning messages. New warning messages can be defined by clicking the button labeled *Add*. In addition, the action to be triggered can also be the execution of an operation. The expert or evaluator can add as many initial classifiers as he/she wishes. The greater the number of classifiers, the greater the proportion of events covered by the classifier and the more precise the identification of situations (events).

Once the classifiers have been defined, the next step is the training and adjustment of the knowledge base rules. Before training, certain parameters of the training algorithm must be defined (Fig. 5C), such as the stop condition for the algorithm, which can be the fulfillment of a specific certainty factor, expressed as a percentage. Also, the frequency with which the genetic algorithm is to run can be set, and is expressed as a number of iterations. Once the parameters have been defined, the training of the rules can begin (*Train* button). When the option of training with certainty factor is selected, the bottom right-hand area of the user interface (Fig. 5D) is enabled. In this case, the events as well as the actions they produce are shown to the expert (trainer) so that he/she can express his/her assessment of the Event-Action pair.

When the knowledge base has been trained, the tool allows two classification algorithms (ID3 and CART) to be executed, in order to validate and observe the behavior of the classifier system. Figure 6 shows a screenshot of the tool when a warning message has been generated as a result of an event being identified.

Integration Into DomoSim-TPC Environment

The architectural model for the entire developed system (DomoSim-TPC and our analysis tool) is shown in Figure 7. This figure shows how the different pieces of the system have been organized. The figure shows a multi-tier client/server architecture, optimized to provide good performance and efficiency. The system has three tiers, namely:

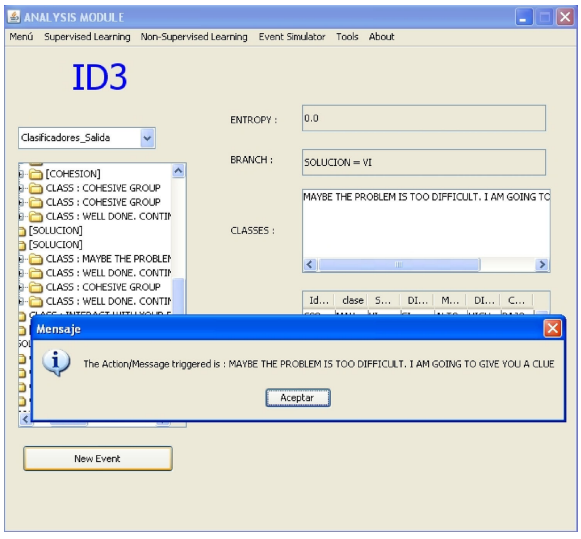


Figure 6 User interface of the classifier system validation tool (using ID3 algorithm). [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

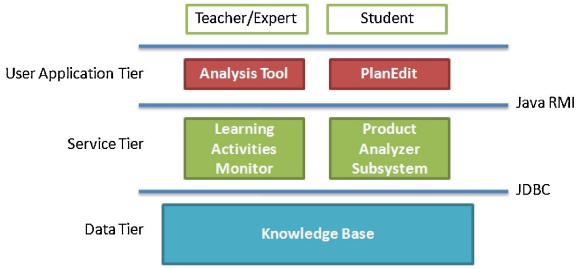


Figure 7 Multi-tier client/server architecture of the developed system. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

User Application Tier: containing the applications that the users (teachers, experts and students) can run on their computers.

Service Tier: containing the Learning Activities Monitor and the Product Analyzer Subsystem.

Data Tier: where all of the data generated from student interactions and communication, student products, expert plan specifications, the messages generated by the Analysis Tool, etc. are stored.

The entire system has been implemented in the Java programming language and using Java-related technologies. As we can see in Figure 7, communication between the *User Application Tier* and the *Service Tier* is performed by means of a Java RMI (*Remote Method Invocation*) connection. Furthermore, all of the generated data are stored in a database. This storage uses remote database access through the API JDBC (*Java Database Connection*). The chosen database management system was MySQL.

The links and the direction of communication between the different pieces of the architecture are shown in Figure 8. On the right-hand side we have the group learning system, DomoSim-TPC, with which the students carry out the various domotical design assignments collaboratively. This system includes two main tools: PlanEdit, providing support for the CPD during an assignment; and a Discussion and Argumentation Space, in which the students justify, question and explain the decisions they make whilst building the solutions.

During the course of a learning session, DomoSim-TPC notifies the *Learning Activities Monitor* (at the top of Fig. 8) about the user interaction in the work group, and notifies the *Product Analyzer Subsystem* (at the bottom of Fig. 8) about the product design events. The *Learning Activities Monitor* gathers all of the interactions and stores them in the *Knowledge Base* for possible subsequent analysis (in the middle of Fig. 8). A similar situation occurs for the *Analyzer Subsystem*, implemented in order to analyze the products built by learners and to store the different versions of the products in the *Knowledge Base*. In due course, if the *Analysis Tool* (on the left-hand side of Fig. 8) is launched, both the *Learning Activities Monitor* and the *Product Analyzer Subsystem* will notify it of the corresponding events. The Analysis Tool allows the teacher to analyze the interaction and visualize the data and the message in the work group by applying the proposed analysis method shown in the fourth section. In addition, this *Analysis Tool* is also able to react to certain situations detected by the GCS (at the top left of Fig. 8), sending the corresponding message to DomoSim-TPC, which will notify the users appropriately.

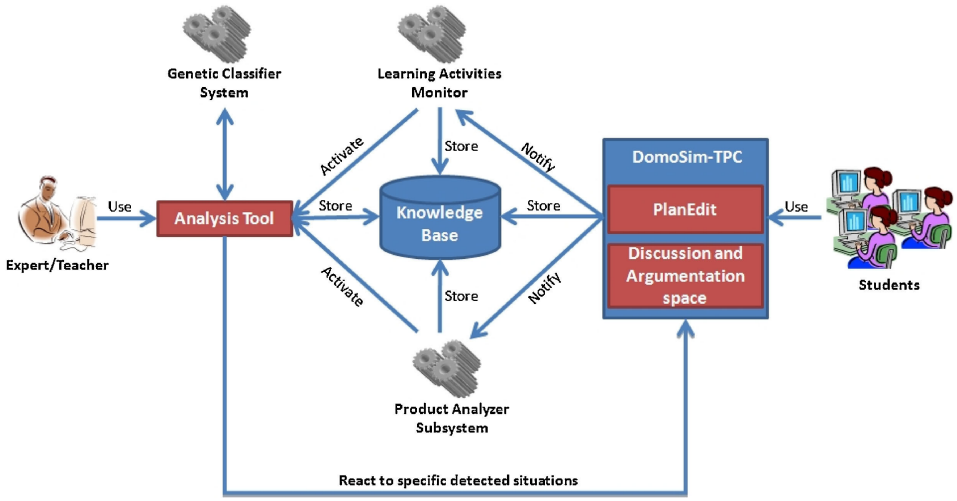


Figure 8 Architectural model for the developed system. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

STUDY

In order to explore the suitability of the proposed method and the developed tool, the designed GCS was applied to real-life collaborative learning activities. To this end, we selected some teachers and students from secondary education (post-compulsory secondary education) and set up a supervised session for them with the DomoSim-TPC system. In the following subsections we set out all of the information related to this session.

Objective

We designed a study in order to get some experimental data about the performance of our approach for process and solution analysis within a CSCL environment with support for asynchronous collaboration. To do so, we followed the template for research goal definition proposed by Wohlin et al. [37]. The goal of the experiment was defined as:

- *Analyze* the performance of the GCS analysis subsystem,
- for the purpose of evaluating,
- *with respect to* learning and application modes,[1]
- *from the point of view of* teachers, evaluators, and analysis designers,
- *in the context of* CPD activities in CSCL environments based on argumentative discussion.

Therefore, the purpose of this experiment is to empirically explore the proposed method's capacity to identify learning scenarios.

[1]The *learning mode performance* measures how well the classifier system is learning how to perform the correct behavior in an environment. The *application mode performance* measures the performance of the learned classifier system in handling problems from the same domain (but different problems) from which it was taught.

Method

We invited some secondary education teachers to participate in the experiment. These teachers randomly chose 12 beginner students of domotics (home automation) who had no previous knowledge of domotical design. These students represented 60% of an entire professional training class.

The experiment was split into several stages. In the first stage, we asked the teachers to specify five domotical design assignments of progressively increasing complexity and size. These assignments consisted of solving a number of problems. In total, five domotical design problems were set, the first being the easiest and the fifth being the most difficult.

In the second stage, we asked the students to answer some questions about design procedures (*pre-test*). The results of this test showed us that their level of knowledge about design, particularly where highly complex problems were concerned, was low. Next, the teachers proceeded to give some master classes, covering theoretical concepts relating to the elements and design procedures used in domotics.

Next, in the third stage, the 12 students were split into six pairs. In organizing the groups, the teachers followed a procedure similar to that used for lab sessions, putting a list of groups up and allowing the learners to form the groups themselves. Also, the size of the groups (two students per group) matched that of the usual lab sessions. Following the advice of the teachers in the experiment, we decided that it would be better to maintain the same traditional structure and organization. As such, the only thing that was different for our experiment was that our system was being used, so the events that the system detected and reacted to were obtained under as normal conditions as possible.

Then, in the fourth stage, the six pairs of students were asked to work with the CSCL environment to solve some design problems. We defined five sessions, of approximately 40 min. In each session, one domotical design problem was set for the groups. The problems were solved using the DomoSim-TPC environment and the PlanEdit tool. These problems are described in more detail in Ref. 29. During the course of the

Table 3 Some Tuples Characterizing Learning Scenarios (Recorded During the Experiment)

Tuples (Sol. Validity, Sol. Difficulty, Modeling, Discussion, Cohesion)
(VI, N, L, M, M)
(VBC, N, M, M, M)
(VC, N, M, H, H)
(VC, Y, H, M, H)
(NV, Y, H, B, B)
. . .

learning sessions, all events were registered and stored for off-line post-processing in order to test our method. During the sessions, of 10 min each, the *Learning Activities Monitor* and the *Product Analyzer Subsystem* of DomoSim-TPC calculated and recorded the values of the analysis indicators. We therefore ended up with a database containing 120 tuples, that is to say, the characterization of 120 real-life learning scenarios. A sample of the obtained tuples is shown in Table 3.

In order to validate the performance of the analysis method we carried out a comparative study of the percentage of events identified between the set of inference rules obtained by applying a clustering process and that obtained by applying the implemented GCS. To create the initial set of inference rules we used 35 of the 120 recorded tuples. The rest of the tuples were used to test the identification ability of each set of rules.

In the first stage of the comparative study, and using the data obtained from the experiment (35 of the 120 tuples), we carried out a clustering process. As mentioned above, the implemented tool allows a *k*-means algorithm to be applied. As a result of the execution of the *k*-means algorithm, five well-differentiated event groupings were obtained. The tuples for two of the groupings are shown in Table 4. The first group is defined by collaborative learning situations in which the obtained solution can be considered *valid correct* (*VC*) or *valid well constructed* (*VWC*), the *solution difficulty* is not relevant, and *modeling*, *discussion* and *cohesion* are considered to be *medium* (*M*) or *high* (*H*). The second group includes situations in which the obtained solution is *not valid* (*NV*), the solution is considered difficult (*Y*), the level of *modeling* is *low* (*L*) or *medium* and both *discussion* and *cohesion* are considered *low* (*L*). These two tuples characterize two sets of possible learning activity situations. The five identified tuples (or clusters) allow for the grouping of the 120 learning situations obtained during the use of the DomoSim-TPC system and characterize and classify them.

In the second stage of the study, we used the five clusters identified in the first stage as an initial set of GCS classification rules. The five discovered groupings were presented to the experts (teachers), who then named the groupings and provided suitable warning messages which could be triggered by the system and shown to the students as each event (or tuple/message defining a learning scenario) is identified. A number

Table 4 Two Groupings Obtained by Applying a *k*-Means Algorithm

Tuples (groupings) (Solution Validity, Solution Difficulty, Modeling, Discussion, Cohesion)
(VC ∨ VWC, #, H ∨ M, H ∨ M, H ∨ M)
(NV, Y, L ∨ M, L, L)

Table 5 Some Classifier Rules

Condition (Solution Validity, Solution Difficulty, Modeling, Discussion, Cohesion)	Message
(VC ∨ VWC, #, #, H ∨ M, H ∨ M)	Well done. Carry on
(NV, Y, L ∨ M, L, L)	Maybe the problem is too difficult. I am going to give you a clue
(NV, #, L, M, L)	You are not focusing on the problem

of classifier rules and the warning messages for the defined group are shown in Table 5. For example, when the tool detects that the solution does not satisfy the specified requirements (i.e., the solution is considered not valid, *NV*), the level of discussion is considered to be medium (*M*), but the contributions carried out do not result in the design plan evolving (the level of modeling is considered low, *L*) and the members of the group are not working together (the level of cohesion level is low, *L*), the intervention message might be "*You are not focusing on the problem*." In this case, the students may be using the communication tool for some other purpose, but they are not working together to create a solution.

Once the knowledge database had been completed, we wanted to draw some conclusions about the performance of our GCS analysis subsystem. We simulated and triggered the occurrence of new events (using the 85 events not used previously). We analyzed the behavior of the analysis subsystem and, in particular, the percentage of event identification. We compared these results with those obtained using the inference rules provided by the *k*-means technique.

Results and Discussion

The result was that by applying a classifier system based exclusively on the groupings made by a non-supervised learning *k*-means algorithm to the aforementioned learning activity, the system was able to detect 56.2% of the new events that occurred. When carrying out the same classification using the proposed GCS with training not based on certainty factors and with 30 iterations, an event identification factor of 70% was obtained. This factor was achieved in the first phase (after the presentation of 30 new events). In successive phases this factor improved, finally reaching 95.1%. As the system evolves, the ability to give a correct answer to new events improves considerably, keeping the number of initial classifiers unchanged. We can therefore observe that the application of the genetic algorithm significantly improves the system's ability to identify situations. In this way, the analysis tool can adapt to the evolving group work, being able to contribute useful information in order to motivate the participants or to encourage specific kinds of situations.

We can see, therefore, how we have obtained a very high percentage of event identification and a proven ability to evolve when searching for suitable answers, reaching identification factors.

CONCLUSIONS

In this paper, we have presented a method for analyzing the collaboration efforts of different groups of learners to solve

the same problem using CSCL systems. Unlike traditional approaches, which focus on analyzing only the group discussion process, the method presented here generates rules to infer indicators that analyze both the group work process and the solutions that are built collaboratively to solve the problem. Unlike CSCL systems which use AI techniques (fuzzy logic, Bayesian networks, etc.) to automate a specific collaborative analysis process, techniques based on GCS can adapt the generation of rules to new collaborative situations and can therefore be used in different CSCL systems. Moreover, the method takes advantage of the inferred analysis indicators to actively intervene in the collaboration with messages that help students to solve the problems detected by the analysis.

A tool has been developed in order to implement the proposed method, based on GCS. This tool not only allows for the use of classifiers made available as a consequence of previous learning activities or clustering processes but also permits new classifiers to be defined by an expert in the field. In the case of defining new classifiers, the tool facilitates the training of the knowledge base by the specification of a very diverse range of parameters which allow the evaluator to direct and limit the entire process. In addition, the behavior of the different classifiers in response to the introduction of certain events into the system (in the form of classification rules, added by the expert) can be monitored. Therefore, there is support for the definition of new criteria which, according to the defined parameters, makes the tool adaptable to different situations and makes it possible to monitor learners' behavior when different events occur.

In order to evaluate the method and the tool, the designed GCS was applied to real-life collaborative learning activities. These activities consisted of solving problems in the domain of domotics, making use of a CSCL environment that supported the exchange of proposals and arguments between group members in order to build design plans. The data obtained from these activities have shown very high percentages of event identification and the ability to evolve in the search for suitable answers, reaching identification factors superior to 90%. The results obtained from these experiments show a high level of performance.

In addition, the flexibility and adaptability of the designed method, which we have observed during the course of this study, allow us to outline and explore its future application to other kinds of CSCL activities, studying in depth the correction of situations which deviate from effective learning in the course of a collaboration process.

ACKNOWLEDGMENT

This study has been partially supported by the Junta de Comunidades de Castilla—La Mancha within the AULA-T (PBI08-0069) and iCoLab projects.

REFERENCES

[1] T. Koschmann, CSCL: Theory and practice of an emerging paradigm. Lawrence Erlbaum, Mahwah, NJ, 1996.

[2] P. Dillenbourg, M. Baker, A. Blaye, and C. O'Malley, The evolution of research on collaborative learning, In: E Espada and P Reiman (Eds.), Learning in humans and machine: Towards an interdisciplinary learning science. Elsevier, Oxford, 1996, pp 189–211.

[3] P. Jenlink and A. A. Carr, Conversation as a medium for change in education, Educ Technol 36 (1996), 31–38.

[4] C. Bravo, M. Á. Redondo, M. F. Verdejo, and M. Ortega, A framework for process-solution analysis in collaborative learning environments, Int J Hum Comput Study 66 (2008), 812–832.

[5] K. Lund, M. Baker, and M. Baron, Modelling dialog and beliefs as a basis for generating guidance in a cscl environment, Proceedings of the Third International Conference on Intelligent Tutoring Systems, Springer-Verlag, 1996.

[6] M. Baker and K. Lund, Promoting reflective interactions in a computer-supported collaborative learning environment, J Comput Assist Learn 13 (1997), 175–193.

[7] A. Soller and A. Lesgold, Knowledge acquisition for adaptive collaborative learning environments, American Association for Artificial Intelligence Fall Symposium, 2000.

[8] M. M. McManus and R. M. Aiken, Monitoring computer-based collaborative problem solving, J Artif Intellect Educ 6 (1995), 307–336.

[9] B. Barros and M. F. Verdejo, Analyzing student interaction processes in order to improve collaboration: The degree approach, Int J Artif Intell Educ 11 (2000), 221–241.

[10] X. d. P. Puente, New method using wikis and forums to evaluate individual contributions in cooperative work while promoting experiential learning: Results from preliminary experience, Proceedings of the 2007 International Symposium on Wikis, ACM, 2007, 87–92.

[11] M. Paolucci, D. Suthers, and A. Weiner, Automated advice-giving strategies for scientific inquiry, Third International Conference on Intelligent Tutoring Systems (ITS'1996), Montréal, Canada, Springer-Verlag, 1996, 372–381.

[12] P. Suraweera and A. Mitrovic, An intelligent tutoring system for entity relationship modelling, Int J Artif Intell Educ 14 (2004), 375–417.

[13] N. Baghaei, A. Mitrovic, and W. Irwin, Supporting collaborative learning and problem solving in a constraint-based CSCL environment for UML class diagrams, Comput Support Collaborative Learn 2 (2007), 159–190.

[14] T. Daradoumis, A. Martinez Mones, and F. Xhafa, A layered framework for evaluating on-line collaborative learning interactions, Int J Hum Comput Study 64 (2006), 622–635.

[15] E. Gómez-Sánchez, M. L. Bote-Lorenzo, I. M. Jorrín-Abellán, G. Vega-Gorgojo, J. I. Asensio-Pérez, and Y. Dimitriadis, Conceptual framework for design, technological support and evaluation of collaborative learning, Int J Eng Educ 25 (2009), 557–568.

[16] A. Soller, J. Wiebe, and A. Lesgold, A machine learning approach to assessing knowledge sharing during collaborative learning activities, Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community, International Society of the Learning Sciences, 2002, pp 128–137.

[17] M. d. l. A. Constantino-Gonzalez, D. Suthers, and J. Escamilla de los Santos, Coaching web-based collaborative learning based on problem solution differences and participation, Int J Artif Intell Educ 13 (2003), 263–299.

[18] M. Mühlenbrock and U. Hoppe, Computer supported interaction analysis of group problem solving, Conference on Computer Support for Collaborative Learning, Palo Alto, California, 1999, pp 398–405.

[19] T. Read, B. Barros, E. Bárcena, and J. Pancorbo, Coalescing individual and collaborative learning to model user linguistic competences, User Model User Adapted Interact 16 (2006), 349–376.

[20] C. J. Butz, S. Hua, and R. B. Maguire, A web-based Bayesian intelligent tutoring system for computer programming, Web Intell Agent Syst 4 (2006), 77–97.

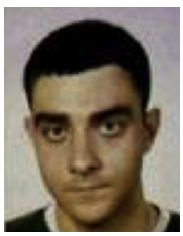
[21] O. Velez-Langs and A. de Antonio, Seeking adaptation in intelligent tutoring systems: A learning classifier systems proposal, International Educational Technology Conference 2005, 100–105.

[22] E. Plantinga, Student modelling using a genetic algorithm, Doctoral Dissertation, Faculty of Mathematics & Natural Sciences, University of Groningen (2003).
[23] M.-J. Huang, H.-S. Huang, and M.-Y. Chen, Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach, Expert Syst Appl 33 (2007), 551–564.
[24] M. A. Redondo, Collaborative planning of design in simulation environments for distance learning, ProQuest Information and Learning USA, 2006.
[25] M. A. Redondo, C. Bravo, and M. Ortega, Contextualized argumentative discussion for design learning in group, In: R Navarro and J Lorés (Eds.), HCI related papers of Interacción 2004. Springer-Verlag, Dordrecht, The Netherlands, 2006, pp 317–328.
[26] M. A. Redondo and C. Bravo, DomoSim-TPC: Collaborative problem solving to support the learning of domotical design, Comput Appl Eng Educ 4 (2006), 9–19.
[27] M. A. Redondo, C. Bravo, M. Ortega, and M. F. Verdejo, PlanEdit: An adaptative problem solving tool for design, Adaptative Hypermedia and Adaptative Web Systems. Springer-Verlag, Heidelberg, Germany, 2002, pp 560–563.
[28] M. A. Redondo, C. Bravo, J. Bravo, and M. Ortega, Applying fuzzy logic to analyze collaborative learning experiences, J U S Distance Learn Assoc 17 (2003), 19–28.
[29] M. A. Redondo, C. Bravo, M. Ortega, and M. F. Verdejo, Providing adaptation and guidance for design learning by problem solving. The DomoSim-TPC approach, Comput Educ 48 (2007), 642–657.
[30] J. H. Holland and J. S. Reitman, Cognitive systems based on adaptive algorithms, In: DA Waterman and F Hayes-Roth (Eds.), Pattern-directed inference systems. Academic Press, NY, 1978.
[31] D. E. Goldberg, Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Reading, MA, 1989.
[32] A. Dimitrakopoulou, State of the art of interaction analysis for metacognitive support & diagnosis, Kaleidoscope Network of Excellence, Deliverable D.31.1.1, 2005.
[33] A. Dimitrakopoulou, State of the art on interaction analysis: Interaction analysis indicators. Kaleidoscope Network of Excellence. Interaction & Collaboration Analysis' Supporting Teachers and Students' Self-Regulation. Jointly Executed Integrated Research Project, Deliverable D.26.1, 2004.
[34] A. I. Molina, M. A. Redondo, and M. Ortega, Un método semiautomático basado en algoritmos genéticos para el análisis de experiencias de aprendizaje colaborativo, JISBD'03—Taller de Hipermedia Colaborativa y Adaptativa, Alicante, 2003.
[35] J. A. Hartigan, Clustering algorithms. John Wiley & Sons, New York, NY, 1975.
[36] T. M. Mitchell, Machine learning. McGraw-Hill, New York, NY, 1997.
[37] C. Wohlin, P. Runeson, M. Höst, M. Ohlson, B. Regnell, and A. Wesslén, Experimentation in software engineering: An introduction. Kluwer Academic Publishers, Norwell, MA, 2000.

BIOGRAPHIES

Ana I. Molina received her Computer Science (2002) degree and her PhD (2007) from University of Castilla – La Mancha (Spain). She then joined the Escuela Superior de Informática (College of Computer Science and Engineering) of University of Castilla – La Mancha. In addition to teaching, her main interests are in the field of New Information Technologies applied to Collaborative Learning and Computer-Human Interaction.

Francisco Jurado is associate professor at the Escuela Superior de Informática (College of Computer Science and Engineering) of University of Castilla – La Mancha. His research interests include Intelligent Tutoring Systems for learning to program, heterogeneous distributed eLearning systems, eLearning standards and computer supported collaborative environments. He received his Computer Science degree in 2005 and his PhD degree in Computer Science in 2010 from University of Castilla-La Mancha.



Rafael Duque is assistant professor in the Department of Mathematics, Statistics and Computer Science at the University of Cantabria (Spain). He received his MSc in Computer Science (2005) and PhD (2010) from the University of Castilla-La Mancha (Spain). His research interests include computer-support for Collaborative Learning and Cooperative Work and Computer-Human Interaction.

Miguel A. Redondo has a PhD in Computer Science (2002) and is associate professor at the Escuela Superior de Informática (College of Computer Science and Engineering) of University of Castilla – La Mancha. His research interests focuses on the fields of new Information Technologies applied to Computers in Education and Computer-Human Interaction.

Crescencio Bravo is associate professor of Computer Systems and Languages at the Escuela Superior de Informática (College of Computer Science and Engineering) of University of Castilla – La Mancha. He has focused his research in the development of structured mechanisms to support collaborative learning and of computational methods to analyze the collaboration process and the resulting products. He has also investigated the generation of intelligent advice to students during modeling tasks in problem solving settings.

Manuel Ortega is Full Professor at the Escuela Superior de Informática (College of Computer Science and Engineering) at the University of Castilla – La Mancha (Spain). He received his BSc (1982) and PhD (1990) degrees from Universidad Autónoma de Barcelona (Spain). He is the director of the *Computer Human Interaction and Collaboration* Research Group and is the Editor of the Iberoamerican Journal on Computers on Education IE Comunicaciones. His main interests are in the field of New Information Technologies applied to collaborative learning and Computer-Human Interaction.