Visualizing Shared-Knowledge Awareness in Collaborative Learning Processes

César A. Collazos[1], Luis A. Guerrero[2], Miguel A. Redondo[3], and Crescencio Bravo[3]

[1] IDIS Research Group, Systems Department, FIET, Universidad del Cauca, Colombia
ccollazo@unicauca.edu.co
[2] Department of Computer Science, Universidad de Chile, Chile
luguerre@dcc.uchile.cl
[3] Department of Information Systems and Technologies, School of Computer Engineering, Universidad de Castilla-La Mancha, España
Miguel.Redondo@uclm.es, Crescencio.Bravo@uclm.es

Abstract. SKA (Shared-Knowledge Awareness) refers to the perception about the shared knowledge students have while working in a collaborative learning context. If we understand the shared comprehension of the problem to be solved as a key part of any collaborative learning activity, SKA will be an indispensable aspect to take into account when designing CSCL systems. In this paper, we propose some design guidelines that can help in the process of graphical user interface design for CSCL tools. We have evaluated some CSCL tools according to the proposed design guidelines depicting how these recommendations materialize in the graphical user interface of some CSCL tools.

1 Introduction

In a previous work, Collazos et al. have defined a new type of awareness for groupware systems called Shared-Knowledge Awareness or SKA [5]. SKA corresponds to the perception about the shared knowledge students have in a collaborative learning scenario. In this paper, we are going to present some mechanisms to visualize this kind of awareness in a Computer-Supported Collaborative Learning (CSCL) scenario. In CSCL scenarios, collaborative learning is effective if people succeed in building and maintaining a shared understanding of the problem [8]. For this reason, the shared understanding should be represented and promoted. We are recently working in trying to capture this shared understanding into an awareness mechanism. One of our hypothesis states that this shared understanding could be promoted if people are aware of its current performance during the collaborative activity, i.e., the individual accountability [21] is clear for each member in the context of the group work.

Information visualization that corresponds to the process of analyzing and transforming non-spatial data into an effective visual form is believed to improve our interaction with huge volumes of data [4, 22]. One central point of any successful

J.M. Haake, S.F. Ochoa, and A. Cechich (Eds.): CRIWG 2007, LNCS 4715, pp. 56–71, 2007.

visualization mechanism is the exploration possibility of visual perception principles. These kinds of visualizations could help an increasingly diverse and potentially non-technical community to gain overviews about general patterns and trends and to discover hidden –semantic semantic– structures. Besides, complex visualizations of different viewpoints of thousands of data objects can greatly benefit from storytelling [10]. One of our goals in this work is the use of techniques such as visual perception, story telling, and artistic aspects of visualization design, to visualize the shared-knowledge awareness. This paper describes a set of recommendations we propose in order to provide SKA visualization mechanisms in a CSCL application and shows how these recommendations materialize in the graphical user interface of some CSCL tools.

The paper is organized as follows. Section 2 describes the principal idea of SKA. Section 3 describes some design guidelines based on some questions that are necessary to ask in order to determine the presence/absence of SKA. Section 4 depicts some software tools we have evaluated according to the proposed design guidelines. In Section 5 we discuss some of the benefits of the model proposed and finally some conclusions and further work are described in Section 6.

2 Shared-Knowledge Awareness

SKA is consciousness on the shared knowledge of a group of students that carry out a collaborative learning activity. The shared knowledge is composed of the understanding of several aspects of the collaborative work, including coordination of the activities, communication of the group strategy, monitoring of the process, and the shared comprehension of the problem [5].

In a collaborative situation, every group member has his/her own individual knowledge of the problem to be solved. Part of this individual knowledge is shared by the group. However, in order to improve the collaboration process among group members, it is necessary to define mechanisms to support discussions and so, to acquire a consensus about the shared understanding of the problem. Thus, a good strategy is to enlarge the shared knowledge, because as Dillenbourg mentions, it contributes to an effective collaborative learning process [8].

In order to create this shared knowledge it is necessary to wonder how one may become aware of one’s own knowledge and how the actions people do affect the knowledge of the other members within the group. We need mechanisms for self-controlling and self-monitoring the learning process. If a student is aware of his/her own knowledge and his/her teammate’s knowledge, he/she can make well-founded strategic decisions. These strategic decisions are meta-cognitive decisions when they are explicit and they are communicated to the team members in order to reason on past or future actions. Such reasoning is precisely required by the negotiation involved when the learners wish to agree on decisions [7]. According to Borges and Pino, awareness mechanisms become crucial for group interactions [1]. If people are aware of what is happening around them through social, task, workspace, conceptual and shared knowledge, it is possible to promote interactions among the members of the group. According to Dillenbourg, this increment in the members' interactions could trigger learning mechanisms [8]. Every member of the group should have awareness of what the others are doing and where they are. They should also receive information about any new viewpoint concerning problem solving (e.g., if one of the participants makes a mistake). There will be a sustained communication among participants in order to share a study of the problem and to interchange solution strategies. Therefore, communicated persons are helping to make strategic decisions and change the participants' knowledge about the problem.

When a member of the group expresses his/her opinion in relation to the shared (and public) knowledge, this will be an attempt to synchronize his/her own understanding with the group-accepted version and make the disagreements clear if there are any. Depending on the outcome of this process, there may be further interactions and negotiations until a new meaning or shared understanding is fully accepted by the group. The key aspects of co-construction of knowledge, meaning, and understanding lie on this process interaction among individuals as well as on their shared and individual cognition [15].

3 The Proposed Model

In order to be aware of the shared knowledge in a collaborative activity we propose a set of questions. For constructing this shared knowledge it is necessary to wonder how one may become aware of one's own knowledge and, how the actions people do affect the knowledge of the other members within the group. It is self-control and self-monitoring of the learning process. We propose per every question a set of SKA design guidelines. These guidelines will be denoted by $Q_{i,j}$, where *i* denotes the number of the question and *j* denotes the specific design guideline for that question. Each of the questions we have proposed are based on some of the Gutwin's elements of knowledge contained within a "who, what and where" category of questions asked about workspace events in the present [11].

Question 1: What are the other members of the group doing to complete the task?
In a collaborative activity, every member of the group has a predefined task. Two levels of accountability must be structured into collaborative activities. The group must be accountable for achieving its goals and each member must be accountable for contributing his or her share of the work. Individual accountability exists when the performance of each individual is assessed and the results are given back to the group and the individual in order to ascertain who needs more assistance, support, and encouragement in learning. The purpose of collaborative learning groups is to make each member a stronger individual in his or her own right. Students learn together so that they subsequently can gain greater individual competency [13]. Thus, we propose the following design guidelines:

- Explicit sensor of task advance ($Q_{1,1}$): to include a mechanism that permits members or the group to know the task evolution.

- Explicit sensor of others' collaboration performance ($Q_{1,2}$): to include a mechanism that allows visualizing contributions the group members have made in order to complete the task in an appropriate way. There are mechanisms developed for providing information about the level of participation in a work group. Collaborative work requires people to participate in other member's activities. It is not desirable that any members perform their tasks with little or no concern about what is going on. Therefore, some level of watching is encouraged. For instance, during a group discussion the coordinator and facilitators receive information about the level of participation of the group members and the evolution in the discussion of ideas. The *participameter* and the *contributionmeter* are examples of such artefacts [1].

Question 2: Are the tasks done by others helping to solve the problem?
In order to be effective in a collaborative activity it is not only important to execute the task but the other members of the group execute their task in an improved way. Thus, it is important for people to do the task according to some criteria that allow the achievement of the proposed goals. The following design guideline is considered:

- Explicit sensor of others' collaboration performance ($Q_{2,1}$).

Question 3: What do other members know about the topic? What do other members need to know about the topic?
There are important cognitive activities and interpersonal dynamics that can only occur when students promote each other's learning. This includes orally explaining of how to solve problems, teaching one's knowledge to others, checking for understanding, discussing concepts being learned, and connecting present with past learning. Each of those activities can be structured into group task directions and procedures. Doing these tasks help to ensure that collaborative learning groups have both an academic support system (every student has someone who is committed to helping him or her to learn) and a personal support system (every student has someone who is committed to him or her as a person) [13]. Mechanisms as conceptual maps or storytelling can be used to support these aspects. It is widely accepted that concept maps can help students to effectively externalize their knowledge in a domain, and evoke and support meta-cognitive activities [16]. Conceptual mapping has been one of the most referenced in the literature [24]. Moreover, the use of concept maps seems to be useful in supporting knowledge management, which is a very important concern in societies rapidly expanding their knowledge resources. This question includes the following design guidelines:

- Explicit sensor of task advance ($Q_{3,1}$)
- Representation of received information ($Q_{3,2}$)
- Mechanism for classifying received information ($Q_{3,3}$)
- Others' user profiles (only if the profile defines part of the topic, i.e., if the profile denotes specific characteristics of the task executed by the group member) ($Q_{3,4}$)

Question 4: How can I help other students to complete the task?
In a collaborative activity, it is assumed that each of the group members is self-interested. That is, each member has his/her own preferences and desires about how the world should be better. We could represent a member's preferences by means of utility functions which assign an indicator to every outcome, showing how good the outcome is for each member. If each member obtains the best utility then the whole collaborative activity would obtain the best utility. Each member can share information relevant to sub-problems and tasks for helping to increase the member's utilities. This information may be shared proactively (one member shares information because he/she believes the other will be interested in it) or reactively (one member shares information in response to a request). This question is related to the following design guidelines:

- Representation of the received information ($Q_{4,1}$)
- Mechanism for classifying the received information ($Q_{4,2}$)
- Others' user profiles (only if the profile defines part of the topic or includes user's expertise) ($Q_{4,3}$)

Question 5: What did other members of the group learn from me?
Positive interdependence is successfully structured when group members perceive they are linked with each other in a way that one cannot succeed unless everyone succeeds. Group goals and tasks, therefore, must be designed and communicated to students in ways that make them believe they sink or swim together. When positive interdependences are solidly structured, it highlights that (a) each group member's efforts are required and indispensable for group success and (b) each group member has a unique contribution to make to the joint effort because of his or her resources and/or role and task responsibilities. This creates a commitment to the success of group members as well as one's own and it is the heart of collaborative learning. If there are no positive interdependences, there is no collaboration [13]. In that way, it is important to do the task in a better way, and to teach other members of the group the performed activity. For this question, we propose:

- Explicit sensor of self-collaboration performance ($Q_{5,1}$)
- Representation of the information received ($Q_{5,2}$)
- Mechanism for classifying the information sent ($Q_{5,3}$)

Question 6: Where are the other members of the group?
In a collaborative activity it is very important to know where the group members are (unless they were working face-to-face) in order to assign a new task or to communicate in a better way. Accordingly, we consider:

- Mechanism to highlight others' last contribution ($Q_{6,1}$)
- Mechanism to restart the actual state of the task (only asynchronous CSCL interfaces) ($Q_{6,2}$)

4 Experimented Tools

In this section, we describe some software tools based on collaborative activities we have designed including mechanism to provide SKA among group members. The design guidelines explained in the previous section are taken into account in order to develop the software tools we have experimented.

4.1 TeamQuest

This game is a labyrinth with obstacles [6]. The players of a team must reach a goal by satisfying sub-goals in each of the game stages. Each player is identified with a role image and name. The screen has three well-defined areas: game, communication and information (Fig. 1). The game area has four quadrants (each one assigned to a player who has the *doer* role; the other players are collaborators for that quadrant). In

Fig. 1. TeamQuest user interface. This is the main interface of the designed software tool which incorporates the Shared-Knowledge Awareness mechanisms.

a quadrant, the *doer* must move an avatar from the initial position to the *cave* that allows entering the next quadrant. In the way, the *doer* must circumvent all obstacles and traps in the map (which are not visible to all players). Moreover, the *doer* must pick an item useful to reach the final destination. The user interface has many elements showing awareness: the *doer*'s icon, score bars, items that were picked up in each quadrant, etc. (see Fig. 1). Table 1 depicts the design elements of SKA that appear in this software tool.

Table 1. Shared-Knowledge Awareness mechanisms present in the game

Aspect	Design Element	Comments
$Q_{1,1}$	Avatar position in the game board (Fig. 1-a)	Collaborator: is the coordinator moving the avatar by skipping my traps?
$Q_{1,2}$	Dialogues boxes to receive messages (Fig. 1-c)	*Doer*: are the other members of the team sending me their traps? Collaborator: is the *doer* asking me something? Everyone: are the others' messages helping me to understand the game strategy?
$Q_{2,1}$	Group energy (Fig. 1-d)	Collaborator: after ending every player's turn, is the group energy better or not?
$Q_{3,1}$	Group energy (Fig. 1-d)	*Doer*: after ending my turn, does the group score increment or decrement?
$Q_{3,2}$	Dialogue boxes to receive messages (Fig. 1-c)	*Doer*: needs information about others' traps adjacent to the current avatar Collaborator: needs to know coordinator questions Everybody: the game strategy can be understood by analyzing the received information
$Q_{3,3}$	Dialogue boxes to receive messages (Fig. 1-c)	Separate dialogue boxes for each player (each participant identified with a different avatar)
$Q_{3,4}$	Avatar position in the game board (Fig. 1-a,f)	Every quadrant in the board is associated with the avatar of the participant that coordinates it. Therefore the avatar position defines the current profile of the other players (*Doer* or Collaborator) Collaborator: knowing who is the current *doer* helps to focalize answering his/her questions
$Q_{4,1}$	Dialogue boxes to receive messages (Fig. 1-c)	*Doer*: he/she has to read others' traps to skip them Collaborator: he/she has to read coordinator's questions to help him/her
$Q_{4,2}$	Dialogue boxes to receive messages (Fig. 1-c)	Separate dialogue boxes for each player (each participant is identified with an avatar)
$Q_{4,3}$	Avatar position in the game board (Fig. 1-a,f)	Every quadrant in the board is associated with the avatar of the participant that coordinates it. Therefore the avatar position defines the current profile of the other players (*Doer* or Collaborator) Collaborator: knowing who is the current coordinator helps to focalize answering his/her questions
$Q_{5,1}$	Avatar position in the game board (Fig. 1-a,f) Individual energy (Fig. 1-g)	Collaborator: after sending a message to the *doer* with his/her traps adjacent to the avatar, the coordinator skipped them. Everyone: my self-performance is reflected in the individual energy
$Q_{5,2}$	Dialogue boxes to receive messages (Fig. 1-c)	Everyone: observing others' questions and Comments can help him/her to perceive his/her performance
$Q_{5,3}$	Does not apply	This element is not present in the tool
$Q_{6,1}$	Dialogue boxes to receive messages (Fig. 1-c) Group energy (Fig. 1-d)	Everyone: others' messages are listed chronologically, therefore it is possible to see the last contributions of other players Collaborator: after seeing the mouse changing from one quadrant to another, the last contribution of the coordinator is reflected in the increasing or decreasing of the group energy
$Q_{6,2}$	Does not apply	This element is not present in the tool

4.2 DomoSim-TPC

DomoSim-TPC is a CSCL system for the learning of domotical design of models that should satisfy a specification. This system incorporates tools for organization, authoring, model building and simulation, communication and coordination, and assessment [17]. The learning activities consist of two phases: (a) *Collaborative Planning of Design*, and (b) *Detailed Design and Simulation*. The first phase is a reflexive task supported by asynchronous tools [19]. The second phase is an interactive task supported by synchronous tools [2]. In this study, we focus on the tools supporting the tasks of the first phase. DomoSimTPC has a specific workspace for the Collaborative Planning of Design. In this workspace, there are tools for individual planning of design (individual elaboration), argumentative discussion about the design strategies (collaboration) and organization of results (or artefacts).

The tool for individual planning of design is called PlanEdit [19]. A plan is specified as a partially ordered set of generic actions for the construction of a model. PlanEdit allows the learner to interactively define design strategies represented by mean of design plans and adapts to the strategy that the learner follows. Fig. 2 shows the user interface of PlanEdit. This is structured in separates areas: the problem formulation, the list of tasks to carry out to solve each sub-problem (a problem can be organized in sub-problems), the icon bars representing design actions (or domain operators), the sequence of design actions already planned, the current action under construction and a set of buttons dedicated to support several general functions.

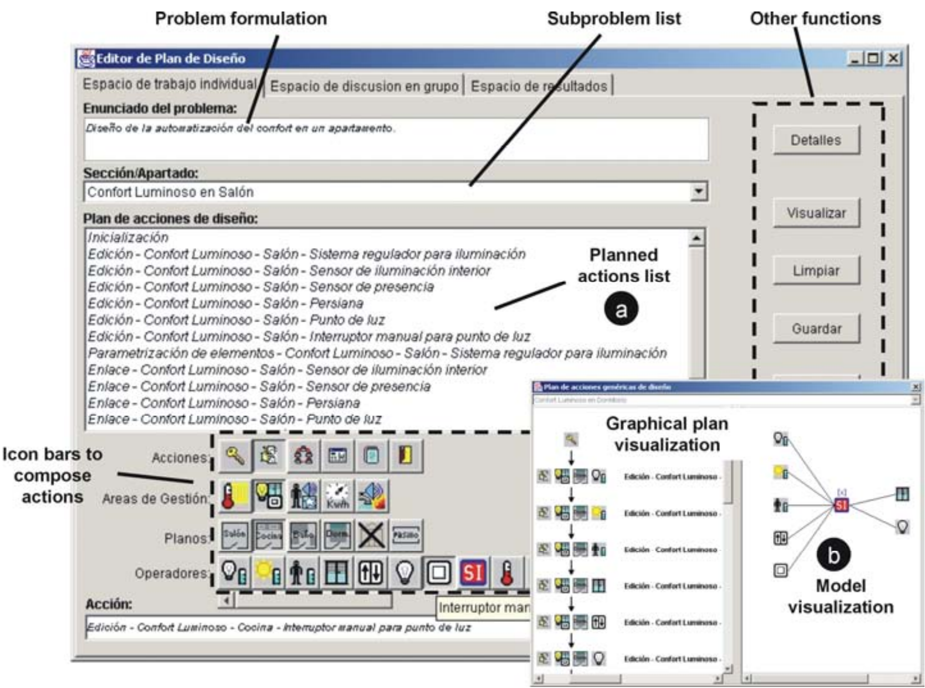

Fig. 2. PlanEdit user interface (in Spanish)

When a student has built a design plan, he/she has to present it to the rest of the group members arguing and justifying his/her design decisions. This is supported by the tool for argumentative discussion about the design strategies [18]. This tool structures the communication by means of a model inspired in the Conversation Based on Topics [20]; it hierarchically represents the evolution of the knowledge constructed in group and uses expert knowledge to guide the process. The objective of this guidance is to lead the group towards a solution in agreement. Fig. 3 shows the user interface of the argumentation tool. In the centre the hierarchical structure is located in the form of a reversed tree used to organize the contributions and to represent the evolution of the knowledge. This structure is called *Scheme of the Discussion and Argumentation Process* (SDAP). On the right, several interaction buttons are presented. The content of the selected contribution can be visualized with the corresponding button. This visualization depends on the kind of contribution. Mainly, they can be design plans or text messages.

In order to give a process-independent support to the organization, presentation and access to the final solution (or artefacts) elaborated and agreed by the group a *Table of Contents* is used. This table shows the epigraphs corresponding to the sub-problems in which the proposed problem was organized. These epigraphs are represented using two different icons. The first one (a *cross*) indicates that the problem does not have a solution in agreement yet and, therefore, it does not have a result associated. The second one (a *tick*) is used when the epigraph has already a solution in agreement.

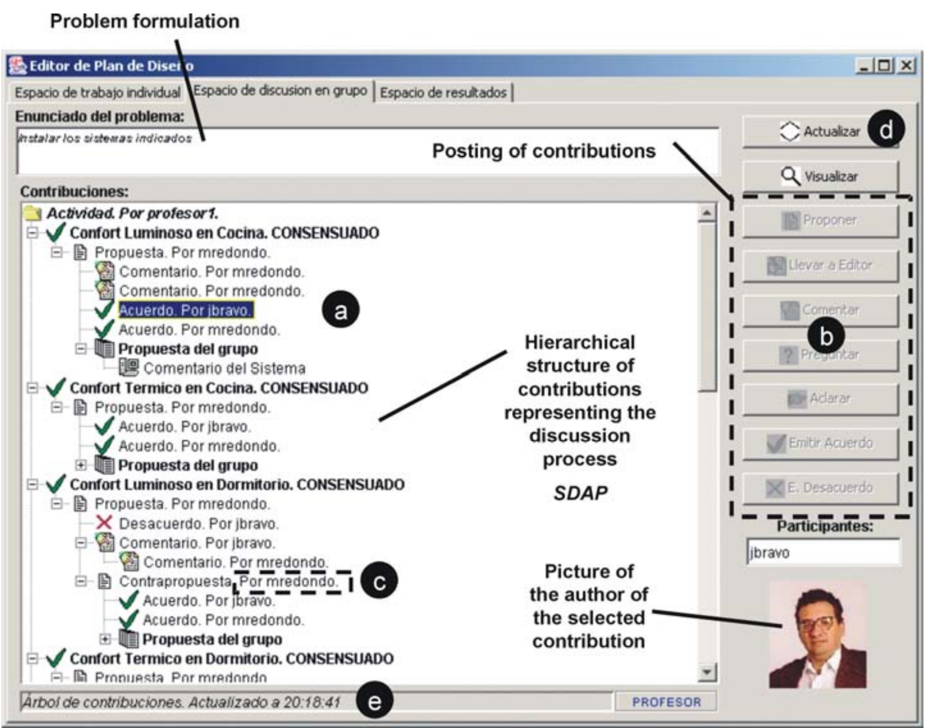

Fig. 3. User interface of the discussion and argumentation tool of PlanEdit (in Spanish)

Table 2. Shared-Knowledge Awareness mechanisms present in DomoSim-TPC

Aspect	Design Element	Comments
$Q_{1,1}$	Schema of the Discussion and Argumentation Process (SDAP) (Fig. 3-a)	The SDAP is a board (reversed tree) which is always displaying the task evolution. That is, this displays the posted contributions and their relations.
$Q_{1,2}$	Panel containing the list of planned design actions (Fig. 2-a)	The plan can also be graphically displayed (Fig. 2-b) in order to facilitate the interpretation of the model and to decide if it is finished.
$Q_{2,1}$	SDAP (Fig. 3-a) and the panel containing buttons which are used to post contributions (Fig. 2-b).	When a user proposes a model, other users can post critics and comments, and suggest improvements. All the users can visualize what the others propose, although restrictions can be established.
$Q_{3,1}$	SDAP (Fig. 3-b) and Table of Contents.	While the discussion process is carried out every user can observe what the rest of members have done (posted) and everyone can post new contributions. In the Table of Contents finished sub-problems and the constructed artefacts (plans) can be observed and reviewed.
$Q_{3,2}$	SDAP (Fig. 3-b)	The posted information is always represented, organized and structured in the SDAP. When new information is posted, it is necessary to relate it to another previous one contained in the SDAP.
$Q_{3,3}$	SDAP (Fig. 3-b and Fig. 3-c)	Labels are associated to the contributions indicating who has been the author of each one (Fig. 3-c).
$Q_{3,4}$	SDAP (Fig. 3-b)	The information posted by each user is visible and accessible to the rest of users from the SDAP. However, in the definition of the learning activity, restrictions can be established.
$Q_{4,1}$	SDAP (Fig. 3-b)	The posted information is always represented, organized and structured in the SDAP. When new information is posted, it is necessary to relate it to a previous one contained in the SDAP.
$Q_{4,2}$	SDAP (Fig. 3-b)	Labels are associated to the contributions indicating who has been the author of each one (Fig. 3-c).
$Q_{4,3}$	SDAP (Fig. 3-b)	The posted information is always represented, organized and structured in the SDAP. When new information is posted, it is necessary to relate it to a previous one contained in the SDAP.
$Q_{5,1}$	SDAP (Fig. 3-b)	Every contribution containing a design model can be moved to the individual workspace (Fig. 3-a) to be analyzed.
$Q_{5,2}$	SDAP (Fig. 3-b)	Everyone: observing others' questions and comments can help him/her to perceive his/her performance
$Q_{5,3}$	SDAP (Fig. 3-b)	Posted information as well as received information is represented in the SDAP.
$Q_{6,1}$	SDAP (Fig. 3-b)	The contributions are organized in a hierarchical structure in the form of a reversed tree. The leaves of the tree are the last posted contributions, but it is not possible to know when they were posted.
$Q_{6,2}$	Button *Actualizar* (Update) of the panel in the discussion tool (Fig. 3-d). Indicator of when the last update was carried out (Fig. 3-e). Mechanism of automatic and periodic update	The tool automatically and periodically updates the information and shows when it was the last update. Also, the user can request an update at any time.

The results of the Table of Contents serve as a starting point for the following phase of Detailed Design and Simulation in group. This phase is outside the scope of this work. Table 2 depicts the design elements of SKA that appear in the tools supporting Collaborative Planning of Design in the DomoSim-TPC system.

4.3 COLLECE

COLLECE (COLLaborative Edition, Compilation and Execution of programs) [3] allows distributed programmers to edit a program or code fragment, to compile it and to run it collaboratively. The system provides a shared workspace that supports an explicit collaboration protocol: first, the students create a program using a shared text editor; then, they are able to compile the program, receiving a list of compilation errors; finally, they can execute the program provided a compiled program is available. To support such protocol, the main user interface of the system includes four main areas (Fig. 4): the edition area at the top (Fig. 4-a), the console in the middle (Fig. 4-b), the chat at the bottom (Fig. 4-c), and the session panel on the right (Fig. 4-d). The console shows both the compilation errors and the execution outcome. In addition, a system function allows the users to consult the compilation statistics, so that the students are aware of their more frequently made mistakes (Fig. 4-e).

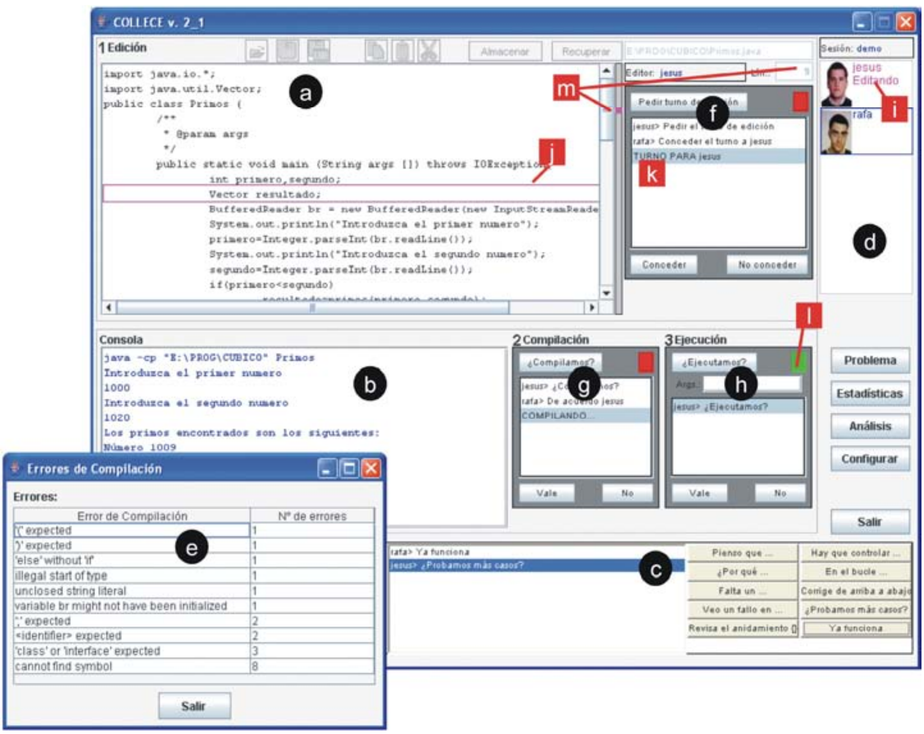

Fig. 4. Main COLLECE user interface which incorporates the Shared-Knowledge Awareness mechanisms (in Spanish)

Communication and coordination tools are available in the workspace for use when required. Communication is materialized by means of a structured chat, i.e. a chat with sentence openers (Fig. 4-c). Coordination processes are modeled with a simple protocol of actions extracted from language. In order to regulate the editor floor control, we identified the acts *Request the edition turn*, *Give* and *Don't give*. With these acts, a student can request the edition turn and his/her fellow students can express his/her agreement or disagreement (Fig. 4-f). When all the users in the group agree, the assignment is made. Similar acts are used for coordinating when to compile (Fig. 4-g) and when to execute the program (Fig. 4-h). These coordination tools support multiple proposals, that is to say, proposals coming from more than one user, and, as a result, lists are required to contain the historical proposals, enabling a user to select the proposal to which he/she wants to respond from such lists.

Besides this support, awareness support is available so that the users can easily perceive and gain knowledge of the interaction carried out by other people in the shared space [11]. The COLLECE awareness is materialized by means of a number of elements: session panel (Fig. 4-d); user's state (*editing*, *compiling* or *executing*) (Fig. 4-i); tele-pointers (Fig. 4-j); lists of interactions (Fig. 4-k); semaphores (Fig. 4-l); beeps, when actions occur; and other mechanisms (Fig. 4-m). Table 2 shows the design elements of SKA that are present in COLLECE. The COLLECE components most related to each design element are shown.

Table 3. Shared-Knowledge Awareness mechanisms present in COLLECE

Aspect	Design Element	Comments
$Q_{1,1}$	Edition area indicator (Fig. 4-m) Console (Fig. 4-b)	The edition area indicator shows the task advance indirectly, since it shows the edition position (the hypothesis is that the higher the position is, the longer the program and the more advanced the task are). The console showing the compilation errors also represents the task advance: the number of errors reduces as the task advances.
$Q_{1,2}$	Chat (Fig. 4-c) Lists of interactions (Fig. 4-k) Semaphores (Fig. 4-l)	The chat includes the users' performance at communication level. The different lists of interactions show the users participation textually; specifically, visual semaphores highlight active behaviours requesting agreements for carrying out shared tasks.
$Q_{2,1}$	Shared editor (Fig.4-a) Execution console (Fig. 4-b) Data input for execution	The program created with the shared editor means a solution to a problem. This program is manipulated, in coordinated turns, by the students. Thus, they help each other in solving the problem. The execution console is a shared space where the program is validated as a solution. The area for data input –this is not shown in the example of Fig. 3– contributes to the understanding of that validation.
$Q_{3,1}$	Compilation errors (Fig. 4-e) Shared editor (Fig. 4-a) Execution console (Fig. 4-b)	At level of task advance sensor, the compilation errors represent knowledge that students need to manage. On the same way, the program in the editor and the execution feedback involve a cognitive level that allows the students to validate the knowledge they have applied in solving the problem.
$Q_{3,2}$	Shared editor (Fig.4-a) Structured chat (Fig. 4-c) Lists of interactions (Fig. 4-k)	A number of components represent in different ways the information received involving knowledge about the topic: the programming instructions in the editor, the textual messages about the programming tasks in the chat, etc.

$Q_{3,3}$	Lists of interactions (Fig. 4-k) Semaphores (Fig. 4-l) Structured chat (Fig. 4-c)	Information about the coordination processes is organized in the lists of interactions. Semaphores help in identifying the open coordination processes. On the other hand, the chat classifies discussion information, especially because the chat structures the conversation by means of the sentence openers.
$Q_{3,4}$	Edition area indicator (Fig. 4-m)	The program line in which the editor user is working represents specific program instructions and algorithm fragments of his/her property (since he/she writes them), from which a user profile would be generated.
$Q_{4,1}$	Structured chat (Fig. 4-c) Lists of interactions (Fig. 4-k)	At communication level, the chat manages information students interchange to solve the problem; at coordination level, the lists of interactions collect coordination actions for leading the tasks
$Q_{4,2}$	Lists of interactions (Fig. 4-k), Structured chat (Fig. 4-c)	Both the lists of interactions and the chat classify the users' actions since they include the user's name.
$Q_{4,3}$	User's state (Fig. 4-i)	The user's state represents the task the leader is carrying out. This is the way that user employs to complete the global task under the others' supervision.
$Q_{5,1}$	Shared editor (Fig. 4-a) Structured chat (Fig. 4-c) Lists of interactions (Fig. 4-k)	The link between the edition and the communication, which allows discussing while editing the program code, promotes positive interdependence. Additionally, the lists of interactions mean sensors of collaborative performance.
$Q_{5,2}$	Shared editor (Fig. 4-a) Structured chat (Fig.4-c) Lists of interactions (Fig. 4-k)	The shared editor shows pieces of information (sequences of instructions) representing learning fragments. Again, the chat and the lists of interactions provide a textual representation of the discussion and coordination processes.
$Q_{5,3}$	Shared editor (Fig.4-a) Tele-pointer (Fig. 4-j) Structured chat (Fig. 4-c) Lists of interactions (Fig. 4-k)	The users' names in the different components and the tele-pointer colour are used to identify the information sent concerning learning.
$Q_{6,1}$	Shared editor (Fig. 4-a) Tele-pointer (Fig. 4-j) Edition area indicator (Fig. 4-m) Structured chat (Fig. 4-c) Lists of interactions (Fig. 4-k) Data input for execution	The COLLECE support for awareness provides rich information about where the last contribution took place.
$Q_{6,2}$	Does not apply	This element is not present in the tool.

5 Discussion

Effective collaboration requires students to engage in task-related, meta-cognitive, and socio-communicative activities. Most CSCL environments support only task-related and meta-cognitive activities. However, few CSCL environments attempt to support socio-communicative activities. One way to support socio-communicative activities may be through visualization of the social aspects of collaboration [9]. Visualization uses software tools and different representations to guide argumentative knowledge construction. Interfaces with different representational aids such as graphs, matrices or texts were found, which have different effects on CSCL [23]. Software tools may visualize the argumentation of learners [14]. For instance, diagrammatic representations visualize how arguments are related to each other and thus facilitate and guide the learners' awareness of the argumentative discourse [12].

In the model we have presented, we include some mechanisms to visualize what the group members are doing in order to determine if what one member is doing is in the correct way to reach the final goal. This information needs to be assimilated by the entire group in order to provide a better collaboration. One of our hypothesis states that this shared understanding could be promoted if people are aware of their current performance during the collaborative activity. The guidelines discussed previously permit group members to know what the other group members know about a certain topic for determining what information change should be tracked and displayed to participants, and what perspectives of viewing this information are relevant to the end user. All the design guidelines we have proposed are explained in the context of the designed software tools.

In a CSCL scenario, these kinds of changes will provide information to the students about the possible learning people have achieved. It is expected that these new tools will provide the students with useful awareness information. This awareness information may trigger explicit communication between students, which facilitates coordination of collaborative activities. This, in turn, may facilitate group processes and lead to better group products and better evaluations of group processes by the students. It is also expected that these new tools will help the teacher to better guide the students' group processes. In spite that the software tools analyzed correspond to different domains and functioning ways (synchronous and asynchronous), we have observed that these guidelines are taken into account. DomoSim-TPC has an asynchronous style and uses a complex structure (inverted tree) to represent the shared knowledge in a structured manner. In this representation, we have observed that is necessary to include some references about the exact time when participants are doing some modifications ($Q_{6,1}$). The other software tools are synchronous and fundamentally use text boxes to show and share messages, as the mechanism to contribute the shared knowledge construction, and some indicators to express that the user has the responsibility to do something (semaphores) and the quality of the task (*Group Score*). The same situation occurs in COLLECE through the cursor position in the edition text area. This last software tool has many elements to express awareness not only about the knowledge but also about the actions and interactions the group members are doing. Some of these elements (for example, the control panel that receives information about who is working every time) still do not have correspondence with the guidelines proposed previously. These aspects suggest us to include more elements in further work.

6 Conclusions and Further Work

Shared-Knowledge Awareness and the respective visualization elements can be used in many group work contexts. For instance, many modern organizations can be seen as network organizations. Participants of these network organizations are individuals as well as other organizations, which collaborate in distributed teams. In some knowledge-intensive domains, such as service engineering, consulting, communities of practice, etc., project teams are composed by participants from different disciplines and organizations that unite their special competencies to match project necessities. Problems occur when participants try to establish a shared knowledge repository and knowledge management mechanisms. A striking point here is the missing awareness of the team members, which leads to the ignorance of the competencies the organization has. One way to improve transparency over the activities of the organization is having SKA mechanisms to be integrated into groupware systems that are mainly used for collaboration needs. The main idea presented in this paper is the use of awareness visualization systems as part of the knowledge management tools, which permit visualizing the shared knowledge to every member of the group in order to help him/her to complete the task in a more effective way.

Of course, in CSCL environments the Shared-Knowledge Awareness is crucial. The group members need to build and maintain a shared comprehension of both the problem and the required tasks. Visualization mechanisms for this kind of awareness will help the group to work in a more efficient manner. Systematic methods for utilization of these mechanisms and the visualization elements are necessary. These methods can help the engineer to design and build new tools supporting learning and group work.

Acknowledgements

This work was partially funded by Spanish Ministry of Science and Tech., CICYT Project ADACO (TIN 2004-08000-C03-03), Colciencias (Colombia) Proj. Nº 4128-14-18008 & 030-2005, and Fondecyt (Chile) grant Nº 1040952 and 11060467.

References

[1] Borges, M., Pino, J.: Awareness mechanisms for coordination in asynchronous CSCW. In: Proceedings of WITS’99, Charlotte, North Carolina, pp. 69–74 (1999)

[2] Bravo, C., Redondo, M.A., Ortega, M., Verdejo, F.: Collaborative Distributed Environments for Learning Design Tasks by Means of Modelling and Simulation. Journal of Network and Computer Applications 29(4), 321–342 (2006)

[3] Bravo, C., Duque, R., Gallardo, J., García, J., García, P.: A Groupware System for Distributed Collaborative Programming: Usability Issues and Lessons Learned. In: Proceedings of International Workshop on Tool Support and Requirements Management in Distributed Projects (REMIDI’07) (2007) (in press)

[4] Card, S., Mackinlay, J., Shneiderman, B. (eds.): Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann, San Francisco (1999)

[5] Collazos, C., Guerrero, L.A., Pino, J., Ochoa, S.: Introducing Knowledge-Shared Awareness. In: Proceedings of IASTED International Conference: Information and Knowledge Sharing (IKS 2002), St. Thomas, Virgin Islands, USA, pp. 13–18 (2002)

[6] Collazos, C., Guerrero, L.A., Pino, J., Ochoa, S.: Evaluating Collaborative Learning Processes. In: Haake, J.M., Pino, J.A. (eds.) CRIWG 2002. LNCS, vol. 2440, pp. 203–221. Springer, Heidelberg (2002)

[7] Dillenbourg, P., Self, J.: Designing human-computer collaborative learning. In: O'Malley, C.C (ed.) Computer Supported Collaborative Learning, Springer, Heidelberg (1995)
[8] Dillenbourg, P.: Some technical implications of the distributed cognition approach on the design of interactive learning environments. Journal of Artificial Intelligence in Education 7(2), 161–180 (1996)
[9] Erkens, G.: Dynamics of coordination in collaboration. In: van der linden, J., Renshaw, P. (eds.) Dialogic learning: Shifting perspectives to learning, instruction, and teaching, pp. 191–216. Kluwer Academic Publishers, Dordrecht (2004)
[10] Gershon, N., Ward, P.: What Storytelling Can Do for Information Visualization. Communications of the ACM 44(8) (2001)
[11] Gutwin, C.: Workspace Awareness in Real-Time Groupware Environments. Ph.D. thesis, Department of Computer Science, University of Calgary, Canada (1997)
[12] Hoppe, H.U., Gaßner, K., Mühlenbrock, M., Tewissen, F.: Distributed visual language environments for cooperation and learning. Journal Group Decision and Negotiation 9(3), 205–220 (2000)
[13] Johnson, D.W., Johnson, R.T., Holubec, E.J.: Cooperation in the Classroom, 6th edn. Interaction Book Company (1993)
[14] Kirschner, P., Buckingham Shum, S.J., Carr, C.S.: Visualizing argumentation. In: oftware tools for collaborative and educational sense making, Kluwer Academic Publishers, Dordrecht (2003)
[15] Lally, V.: Analyzing teaching and learning interactions in a networked collaborative learning environment: issues and work in progress. In: Procc. Of Euro CSCL'01 (2001)
[16] Novak, J.D.: Concept mapping: A useful tool for science education. Journal of Research in Science Teaching 27(10), 937–949 (1990)
[17] Redondo, M.A., Bravo, C.: DomoSim-TPC: Collaborative Problem Solving to Support the Learning of Domotical Design. Journal Computer Applications in Engineering Education 4(1), 9–19 (2006)
[18] Redondo, M.A., Bravo, C., Ortega, M.: Contextualized Argumentative Discussion for Design Learning in Group. In: Navarro, R., Lorés, J. (eds.) HCI related papers of Interacción 2004, pp. 317–328. Springer, Heidelberg (2006)
[19] Redondo, M.A., Bravo, C., Ortega, M., Verdejo, F.: Providing adaptation and guidance for design learning by problem solving. The DomoSim-TPC approach. Computers and Education 48(4), 642–657 (2007)
[20] Savery, J., Duffy, T.: Problem based learning: An instructional model and its constructivist framework. In: Wilson, B. (ed.) Constructivist learning environments: Case studies in instructional design, pp. 135–148. Prentice Hall, Englewwod Cliffs (1996)
[21] Slavin, R.: Using Student Team Learning. Center for Social Organization of Schools, Johns Hopkins University (1980)
[22] Spence, B.: Information Visualization. Addison-Wesley, Reading (2001)
[23] Suthers, D.D., Hundhausen, C.D.: Learning by constructing collaborative representations: An empirical comparison of three alternatives. In: Dillenbourg, P., Eurelings, A., Hakkarainen, K. (eds.) European perspectives on computer-supported collaborative learning, University of Maastricht, pp. 577–592 (2001)
[24] Wong, M.: The concept map: A tool for learning science concepts and understanding cognitive processes of preservice teachers – A Case Study. University of Malaysia Sarawak, Malaysia (1997)