Interaction Analysis Supporting Participants' Selfregulation in a Generic CSCL System

Jacques Lonchamp

Interaction Analysis Supporting Participants' Self-regulation in a Generic CSCL System

Jacques Lonchamp

LORIA-Nancy Université, BP239, 54506, Vandœuvre-lès-Nancy Cedex, France
jloncham@loria.fr

Abstract. Interaction analysis can provide information directly to learners and teachers in order to assess and self-regulate their ongoing activity. Omega+ is a generic CSCL system that uses explicit models as parameters for flexibly supporting different kinds of collaborative applications. This paper describes Omega+ model-based generic approach for supporting participants' self-regulation through interaction analysis. Some quantitative and qualitative results obtained with the proposed approach are discussed.

Keywords: collaborative learning, CSCL, interaction analysis, coaching, self-regulation, generic system, model-based approach.

1 Introduction

Computer Supported Collaborative Learning (CSCL) emphasizes the importance of social processes as an essential ingredient of learning. CSCL has been recognized as a possible way for preparing people to the knowledge society, for achieving deeper learning than traditional methods and for better meeting the expectations of the net generation [1]. During its first decade, CSCL researchers have produced a large number of *ad-hoc systems* focusing at a microscopic level on *particular situations and contexts*, and aiming at triggering *specific learning processes*. It is the case of all early specialized synchronous tools for structured discussion (e.g., [2]), collaborative design (e.g., [3]), knowledge construction (e.g., [4]), modeling (e.g., [5]) and writing (e.g., [6]). Many researchers claim that this first generation of ad-hoc, specialized, and closed tools should be replaced by systems "*richer and appropriate for various collaborative settings, conditions and contexts*" [7], "*reconfigurable*", "*adaptive*", "*offering collections of affordances*" and "*flexible forms of guidance*" [8], "*very flexible and tailorable*" [9]. A promising approach for meeting these expectations is to use an *explicit model* which *parameterizes a generic kernel* for flexibly supporting different kinds of collaborative applications [10]. By providing ad hoc models, teachers can tailor the kernel to their specific needs (*definitional malleability*). Moreover, the behavior of the customized system depends on that, continuously queried, model and can dynamically evolve when the model is modified (*operational malleability*). This technological orientation, implemented in the *Omega+ project*, raises many important conceptual and technological issues.

There exist two complementary approaches for supporting collaborative learning. The first one *structures* the situation in which the collaboration takes place, for instance through process models and interaction models. The second one involves regulating the collaboration itself through *coaching* and *self-regulation* [11]. In this paper we focus on *interaction analysis* that provides information directly to learners and teachers in order to *assess and self-regulate* their ongoing activity. In the specific case of a generic system, such as Omega+, interaction analysis itself has to be generic, i.e., *model-based*. A specific sub-model, called the 'Effect Model', specifies how to measure the effects of collaborative learning. More precisely, it describes the *properties and rules of the interaction analysis and visualization process* for the learning situation defined by the other (process, interaction, and artifact) sub-models which together parameterize the generic kernel.

The remainder of the paper is organized as follows. Section 2 depicts Omega+ overall approach for modeling synchronous collaborative learning activities, i.e., the context of the work. Section 3 discusses the main characteristics of interaction analysis aiming at coaching and self-regulating participants. Section 4 presents Omega+ model-based implementation. Finally, the last section gives some quantitative and qualitative results obtained from a first evaluation study of the current implementation.

2 Modeling Collaborative Learning Activities - Omega+ Approach

The *combination of communication with shared work artifacts* is an important characteristic of synchronous (same-time/same-place or same-time/different-places) CSCL systems [12]. Most of them follow the *dual interaction spaces paradigm* [13], by providing two distinct spaces of interaction. The task space (shared workspace) allows for the collaborative construction and manipulation of shared artifacts that are relevant to the task at hand. The communication space is the place where dialogue-based interaction, mostly textual, takes place. Several recent systems provide multiple tools in the task space for manipulating simultaneously complementary artifacts or multiple views of the same artifact (e.g., [14]).

In a non-trivial CSCL application, the learning task is structured into a *process* including a sequence of collaborative phases. Within each phase participants can play different *roles* which constraint how they can act (in the task space) and how they can talk (in the communication space). In Omega+, a process is a sequence of phases, taking place into rooms: 'simple phases', where all participants collaborate on the same task in the same room, and 'split phases', where participants are divided into parallel sub groups performing different tasks in different rooms. A process model (machine-interpretable script) is a plan which does not prescribe the execution of phases exactly in the specified order. Participants playing the predefined 'Room Operator' role have two buttons for selecting the next phase to execute, either by following the plan (Next) or by selecting any other existing phase (Jump). Each phase type is mainly characterized by a set of role types, a set of tool types and a *floor-control policy* (FCP) at the environment level [15]. In Omega+, application-specific *interaction protocols* are defined by a set of application-related roles, a set of typed messages (speech acts), and a set of adjacency pairs [16] specifying how messages types are related (e.g., question-answer) and which role can speak first. *Application-specific FCPs* can use application protocols (see the 'based_on' relationship in Figure 1) for controlling the floor either in the communication space only or in both the communication space and the task space (see the 'impact' relationship between FCPs and tools in Figure 1). Figure 1 summarizes Omega+ overall conceptual model.

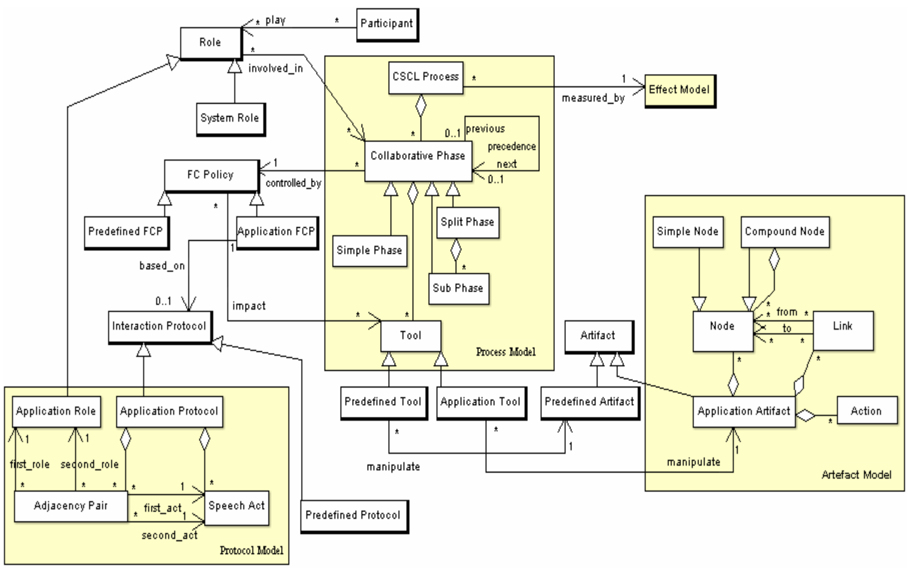
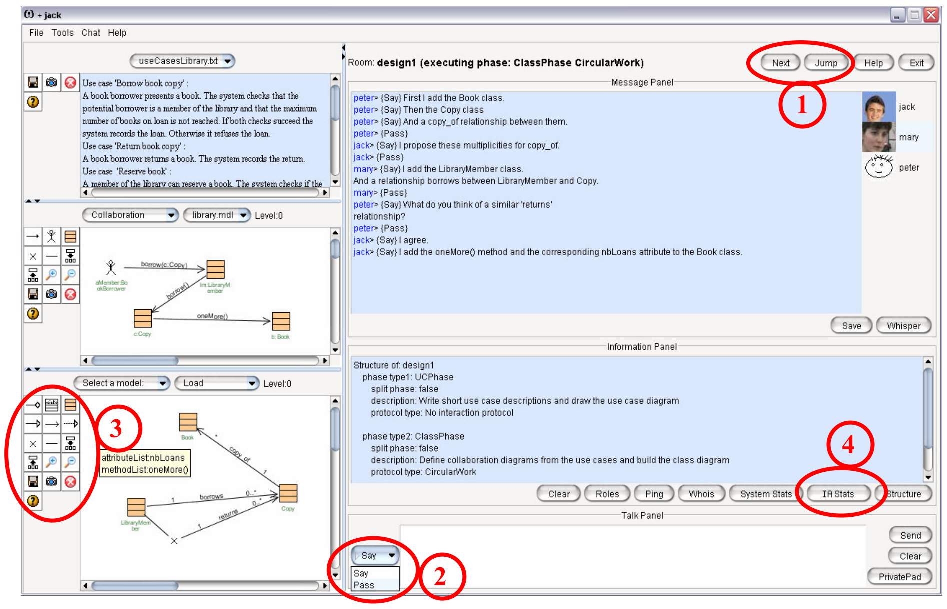


Fig. 1. Omega+ conceptual model

It is worth noting that all important concept types (roles, tools, protocols, FCPs) are specialized into *predefined* and *application-specific* sub types. This reflects the fact that Omega+ provides both hard-coded mechanisms and model-based features which are customizable and evolvable by its users. Three sub-models are highlighted in Figure 1, corresponding to the *process dimension*, the *interaction dimension* and the *artifact dimension* of collaborative learning activities. A fourth one, specifying how individual and group performance can be characterized corresponds to the entity called 'Effect Model'. This sub-model will be studied in greater details in the following sections. Concretely, these four sub-models serve as four *parameters* for the generic Omega+ kernel. This approach makes possible to build the activity representation in different ways, adapted to the skills and needs of different categories of users (researchers, technologists, early adopters, regular teachers): just reusing existing combinations of models, building new combinations of existing sub-models (i.e., following a very high level configuration process), defining or customizing sub-models through high-level visual languages, or through low-level specification languages (including programming languages).

Figure 2 shows Omega+ system customized for supporting a collaborative process of object-oriented design. The communication space on the right includes a protocol model driven chat and an information panel. As Jack plays the Room Operator role the Next and Jump buttons are available to him. The task space on the left may contain up to three tools as requested by the process model definition. Tools are either predefined editors (shared text editor, shared whiteboard) or shared graphical editors customized by artifact (meta-) models. In Figure 2 the task space includes a read-only text viewer (its colored background shows that interaction is not possible) displaying use cases created during a previous collaborative phase and two instances of Omega+ generic visual editor customized with UML collaboration and class diagram meta-models. The model-driven 'Circular Work' protocol controls the floor in both spaces in a round robin fashion. A participant can explicitly pass the floor to the next user with a 'Pass' message (see the combo box on the bottom of the talk panel). Omega+ also provides several dedicated mechanisms for supporting learners at the cognitive and meta cognitive level, such as sticky elements ('sticky annotated snapshots', 'sticky notes' and persistent pointers) for *referencing purposes* [17] and a tool for *collaboratively replaying* any episode of the ongoing knowledge building process (the 'collaborative history browser').



Use case 'Borrow book copy' :
A book borrower presents a book. The system checks that the potential borrower is a member of the library and that the maximum number of books on loan is not reached. If both checks succeed the system records the loan. Otherwise it refuses the loan.
Use case 'Return book copy' :
A book borrower returns a book. The system records the return.
Use case 'Reserve book' :
A member of the library can reserve a book. The system checks if the

Room: **design1 (executing phase: ClassPhase CircularWork)**

```
peter> {Say} First I add the Book class.
peter> {Say} Then the Copy class
peter> {Say} And a copy_of relationship between them.
peter> {Pass}
jack> {Say} I propose these multiplicities for copy_of.
jack> {Pass}
mary> {Say} I add the LibraryMember class.
And a relationship borrows between LibraryMember and Copy.
mary> {Pass}
peter> {Say} What do you think of a similar 'returns'
relationship?
peter> {Pass}
jack> {Say} I agree.
jack> {Say} I add the oneMore() method and the corresponding nbLoans attribute to the Book class.
```

```
Structure of: design1
  phase type1: UCPhase
    split phase: false
    description: Write short use case descriptions and draw the use case diagram
    protocol type: No interaction protocol

  phase type2: ClassPhase
    split phase: false
    description: Define collaboration diagrams from the use cases and build the class diagram
    protocol type: CircularWork
```

Fig. 2. Omega+ client reflecting process (1), protocol (2), artifact (3) and effect (4) sub-models

3 Interaction Analysis Supporting Participants' Self-regulation

The previous section has illustrated learners' support by structuring the collaborative learning process, its artifacts and interaction protocols. The remainder of the paper focuses on the second approach which involves structuring the collaboration itself through *coaching and self-regulation* [11]. It requires storing continuously the stream of actions and messages, counting them in a predefined set of low level variables, *computing on demand* from these low level variables a set of interaction indicators for supporting the coaching process, and *displaying periodically* a set of synthetic visual representations supporting learners' self-regulation. Indicators for the coaching process are freely configurable in the 'Effect model' (see section 4.3). For the self-regulation process, indicators well fitted to the dual interaction spaces paradigm and reflecting the diversity of indicators proposed in the CSCL field [18] have been selected. Some of them are task-dependent and their precise definition is specified in the 'Effect Model'.

3.1 Interaction Indicators for Participants' Self-regulation

In a system following the dual interaction spaces paradigm, learners can interact in two ways. They can communicate directly, by using the chat tool, or indirectly, by building the shared artifacts. The *balance between conversation events and action events* is interesting for measuring a correct usage of both modalities by learners. Pure action without dialogue and pure dialogue without any action are not likely to occur. *Participation indicators*, such as the number of produced messages and the number of tool actions, are important because collaboration cannot occur within a group unless there is roughly equal participation among its participants [19]. If some participants do the main part of the work while others barely contribute, then the group is not truly collaborating. Indicators about the *communication style* can also be helpful. It is the case of *the average size of produced messages*, because it is important that learners make efforts for externalizing their ideas and thoughts [20] in an elaborated form. For chat interaction, it is important *to distinguish between on-task and off-task messages*. This indicator is obviously task-dependent. Off-task messages are useful for specific purposes, such as socialization, but should be restricted in quantity for keeping the learners focused on the constructive task at hand. Interaction requires that group members *actively respond to one another, react and change their minds as the interaction progresses* [19]. The most straightforward approach for measuring interaction is to track *event patterns* which reflect specific 'interaction episodes': for instance, two learners who successively modify the same element (or directly linked elements) of a graph-based artifact. A second possible approach is to *track explicit referencing mechanisms usage* such as participants' names referencing, chat line referencing, sticky elements creation and referencing, and so on. A third approach is to track *event patterns* showing an individual activity aiming at *facilitating collaboration* ('facilitation episode'). For instance, when the same learner changes something on a shared artifact and immediately after or before sends a task-related message with the chat tool, hopefully containing some explanation. Pattern definitions should be customizable for each specific task.

3.2 Interaction Indicators Presentation

For coaching purposes, teachers should be able to access *on demand* to *a detailed representation* of the selected indicators. In particular, *stacked bar charts* are important for contrasting the values for the different learners (for the current phase or for the whole learning process) and *time series* are important for showing the temporal evolution of the values (with a customizable time interval). For self-regulation, learners should receive *periodically* a more *synthetic view*. This view can encompass a *visual evaluation of each criterion*, a kind of *global score* for motivating the participants and *guidance* on what they must do for improving their performance.

4. Omega+ Generic Implementation

The first two subsections discuss message classification and pattern recognition which are complex issues in a generic context. Coaching indicators definition and the monitoring window generation are then discussed.

4.1 Message Classification

For classifying messages into on-task and off-task categories we use a *Naive Bayes Classifier (NBC)* [21]. NBC approach is one of the most effective probabilistic approaches currently known for text categorization. The method is considered naive due to its assumption that every word in the document is conditionally independent from the position of the other words given the category of the document. The classifier learns a set of probabilities from the training data during the learning phase. It then uses these probabilities and the Bayes theorem to classify any new documents. First, an estimate of the probability of the new document belonging to each class given its vector representation is calculated. Then, the class with the highest probability is chosen as a predicted categorization. Omega+ NBC *learns the task vocabulary* by analyzing several sources. First, when the server starts, one (or several) file(s), explicitly referenced into the 'Effect Model' are processed. These files can contain for instance a textual description of a given diagram type (e.g., <OnTaskFile file="usecase.txt" />) or a summary of the instructions given to the learners. The classifier also analyzes all the files loaded into the text board (containing for instance the problem description submitted to the learners), all the meta-model files which serve as parameters for the generic diagram editor (giving in particular the names of all the concepts), and all the models created with the customized diagram editors (giving in particular the names of all nodes and links created by the learners). Omega+ NBC also includes a 'stemming' phase and a 'stopwords' removal phase. Stemming is the process by which words are reduced to their root forms [22]. For example, suffixes are removed, such as "-ing" and "-s", such that "digging" and "dig" become the same word. Stopwords are words that occur frequently in the language, such as "a", "and" and "the" (http://www.snowball.tartarus.org/algorithms/english/stop.txt). Because of their frequent occurrence, they may not add any additional information to aid classification, assuming a uniform distribution over all classes. English and French languages are supported.

4.3 Patterns Recognition

In this first implementation, a very simple language is provided for specifying interaction patterns: <InteractionPattern actors="aaa" tooltype1="xxx" tooltype2=

"yyy" tools= "zzz" condition="ccc" maxtime="mmm" />, where: actors = same | different; toolType1 = chat | diagrammer | whiteboard | text board; toolType2 = chat | diagrammer | whiteboard | text board; tools = same | different; condition = none | <a_condition_name>; maxtime = n | any.

The proposed language is *extensible* as it is possible to create a condition for a particular task by writing a dedicated method having the same name than the condition in the 'InteractionAnalysis' class of Omega+ server. For instance, <InteractionPattern actors="different" tooltype1="Diagrammer" tooltype2= "Diagrammer" tools="same" condition="sameobject" maxtime="60000" /> defines a pattern specifying that two different learners modify during the same minute the same object in the same diagram. <InteractionPattern actors="same" tooltype1= "Diagrammer" tooltype2="Chat" tools="different" condition="ontask" maxtime= "30000" /> defines a pattern specifying that the same learner acts on a diagram and sends an on-task message during the next 30 seconds.

Omega+ server stores the message/action history and checks all the patterns *each time an element is inserted* by testing all elements belonging to the time interval. The synthetic view ('monitoring window') is generated for each participant with a frequency also specified in the Effect Model as a multiple of the time series delta (<MonitoringDelta nbTimeSeriesDelta="4" />).

Suthers [23] emphasizes that interpreting actions in a shared workspace in terms of their domain-specific semantics is difficult and illustrates the danger of taking a superficial approach when mapping domain level actions to intentions. It is hypothesised that our simple pattern definition language *is sufficient for roughly estimating the amount of collaborative episodes*. It is not intended to provide a precise quantification but to *trigger guidance* in the case of a very low amount of interaction.

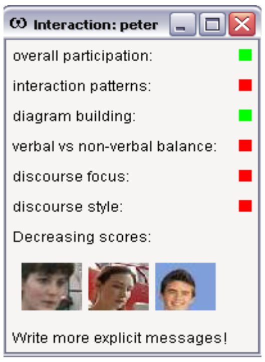
4.2 Coaching Indicators Specification

The 'Effect Model' defines some general parameters, such as the time interval between measures for time series (<TimeSeriesDelta ms="30000" />). It also defines the characteristics of graphical representations used for coaching purposes: name, informal description, type (bar chart, time series), value labels, and expressions defining how values are computed from the predefined set of low-level variables. For instance, <Diagram name="MeanMessLengthSeries" descr="Time series of the mean length of chat messages" type="TimeSeries" labels="length" exprs="ratio: sizeMess nbMess"/>, <Diagram name="MessVSInteractionChart" descr="Bar chart of the number of chat messages vs. other interactions" type="BarChart" labels="chat messages, other interactions" exprs="nbMess, sum: nbWhite nbTextb nbDiag" />.

4.4 Monitoring Window Generation

A guidance message is generated when the ratio between the value of an actor and the average value is under a given threshold. A set of rules in the 'Effect model' specify these thresholds and messages that must be generated. For instance, the rule associated to the discourse focus indicator is <Rule name= "DiscourseFocus"

threshold="0.35" message="Your discourse should be more focussed on the task!" />. A negative value indicates that the indicator is deactivated. Figure 3 shows the monitoring window generated for Peter. In this example, all indicators are activated. Some interaction parameters are well rated (two green squares) while others are weak (four red squares). Peter has the lower global score and receives guidance about his discourse style because this indicator has the worse score ('Write more explicit messages!').



overall participation: 🟩
interaction patterns: 🟥
diagram building: 🟩
verbal vs non-verbal balance: 🟥
discourse focus: 🟥
discourse style: 🟥
Decreasing scores:

Write more explicit messages!

Fig. 3. Omega+ monitoring window

5. A First Evaluation Study

The goal of the evaluation study was twofold: first, analyze the efficiency of the more complex indicators (Bayesian classification and patterns recognition), and secondly carry out interviews to know how learners evaluate the approach. The study has been performed with 24 French students enrolled in a second-year university course in computer science. Small groups of three students, randomly assigned to the groups, received small case descriptions and were asked to build UML use case diagrams or UML Class Diagrams during 30 to 45 minutes length collaborative sessions with Omega+. Students were collocated (in the same classroom) but were not allowed to speak. Omega+ client was configured with a read-only text-board for the case description, a customized shared diagram editor and a chat tool. Students had free access to the communication space and to the task space and no specific process was enforced. Specific control information was written in the log file: the classification of each message by the Bayesian classifier, each recognized pattern associated to its triggering event, and all monitoring values periodically computed for each learner.

5.1 Message Classification

Table 1 gives the results obtained by the Bayesian classifier after a learning phase using a less than one page text file describing the UML formalism. The decision of the classifier ('Classified' column) is compared with the decision of a researcher who has analyzed the messages in the log file after the session ('Analyzed' column). The automatic classification into on-task and off-task messages has an accuracy of 82%. It is sufficient for characterizing students who are not focussed on their task. Errors have multiple causes which are difficult to eliminate. Here are some examples of false off-task messages and false on-task message (translated from French):

- *improper word usage*: in the message "I place the two functions I have said", the word 'function' is used instead of 'use case' or 'case' for a component in a use case diagram and the message is not recognized as being on-task,
- *non explicit reference*: the message "I put them" is not recognized as being on-task because no distinctive word is found,
- *word improperly recognized*: in the message "I am pretty happy of my reorganisation", "reorganisation" is stemmed into "organisation" which is part of the actor concept definition in a use case diagram "an actor is a person, an organisation or a system (…)"; the message is incorrectly classified as an on-task message.

Misspellings, compounding, abbreviations and initialisms ("answ" for answer), reduplications ("heeellllooo"), and frivolous spellings of interjections ("okey") are other well known difficulties [24]. Students were asked to avoid 'chatspeak' and to spell and punctuate correctly.

Table 1. Classification evaluation

Category	Classified	Analyzed	%
a	On task	On task	37
b	On task	Off task	8
c	Off task	On task	10
d	Off task	Off task	45

Evaluation

accuracy = (a + d) / (a + b + c + d) = 82%
error = (b + c) / (a + b + c + d) = 18 %

5.2 Patterns Recognition

As expected, pattern-based indicators only provide rough evaluations. For example, in the log file excerpt of Figure 4 the same rule (defined by <InteractionPattern actors="different" tooltype1="Diagrammer" tooltype2= "Diagrammer" tools="same" condition="sameobject" max time="60000" />) fires twice in the last two lines. In the first case, the learner with the pseudo name 'titi' has created a node at the first line (action numbered 52) and has given to this node the name 'acessoir' which includes a typo (it should be 'accessoire' in French, action 54). This typo has been corrected by 'tata' (action 64) 28 seconds after. This is a reasonable example of collaboration with a student who analyzes and reacts to the action of another student. In the second case, 'toto' has added several properties to the node 'titi3'. The rule was triggered only because 'titi' has moved the same node 26 seconds before when he/she was reorganizing several elements in the graph (actions numbered 56-60). In this case, there is *no real semantic relationship between the two episodes*. A possible improvement could be to test in the method associated to the 'sameobject' condition a Boolean matrix specifying for all couples of actions of the diagram editor if they should be considered or not for triggering the rule (it would be true for 'addVertex' and 'newName' actions and false for 'move' and 'newProperties' actions).

```
févr. 06 15:10:39 in ex1 titi performs a diagram action: Diagrammer0 52:addVertex:Classe:titi5:
févr. 06 15:10:47 in ex1 toto performs a diagram action: Diagrammer0 53:newProperties: numPermis
            Conduire||:toto2:|:
févr. 06 15:10:49 in ex1 titi performs a diagram action: Diagrammer0 54:newName:accessoir:titi5:|:
févr. 06 15:10:54 in ex1 toto performs a diagram action: Diagrammer0 55:newName:Numéro:titi3:|:
févr. 06 15:10:56 in ex1 titi performs a diagram action: Diagrammer0 56:move:336:258:titi5:
févr. 06 15:10:56 in ex1 titi performs a diagram action: Diagrammer0 57:move:195:52:toto0:
févr. 06 15:10:57 in ex1 titi performs a diagram action: Diagrammer0 58:move:196:72:toto0:
févr. 06 15:11:01 in ex1 titi performs a diagram action: Diagrammer0 59:move:190:240:titi3:
févr. 06 15:11:02 in ex1 titi performs a diagram action: Diagrammer0 60:move:302:245:titi5:
févr. 06 15:11:07 in ex1 tata performs a diagram action: Diagrammer0 61:addVertex:Classe:tata0:
févr. 06 15:11:10 in ex1 tata performs a diagram action: Diagrammer0 62:newName:camion:tata0:|:
févr. 06 15:11:13 in ex1 tata performs a diagram action: Diagrammer0  63:move:379:306:tata0:
févr. 06 15:11:18 in ex1 tata has triggered rule 2 in the following action
févr. 06 15:11:18 in ex1 tata performs a diagram action: Diagrammer0  64:newName:accessoire:titi5:|:
févr. 06 15:11:27 in ex1 toto has triggered rule 2 in the following action
févr. 06 15:11:27 in ex1 toto performs a diagram action: Diagrammer0 65:newProperties: code,nom,
            nbArtistes,durée||:titi3:|:
```

Fig. 4. Excerpt of a log file with rule triggering

5.3. Other Indicators

The other indicators, mainly based on the number of messages and actions, are easy to implement. The results show that some deeper analysis could be envisioned. For example, in the session summarized in Table 2, different 'profiles' would be easy to detect with student1 mainly talking, student2 mainly constructing the shared artifact and student3 mainly improving the graph layout by moving nodes and edges.

Table 2. Participation analysis

Action	Student 1	Student 2	Student 3	Total
Node creation	3	**7**	3	13
Link creation	3	**13**	3	19
Node or link movement	39	53	**137**	229
Chat contribution	**20**	15	15	50
Total	65	88	158	311
Action/minute	2,8	3,8	6,9	4,5

5.4. Evaluation of the Approach

From the students' point of view, *personalized advices generation* appears to be the most effective way of pushing information to them periodically. Most students recognize that they do not take the time for analyzing in detail the analytic part of the monitoring window. The overall ranking is perceived as a kind of 'high score' that can increase their motivation to actively participate.

After the presentation of the system, the regulation approach generates much more debate and *controversy* than the structuring approach. Constraints at the process, protocol or artifact level are well accepted as pedagogical constraints, while monitoring rules are strongly rejected by a majority of students and we noticed many attempts to fight against the rules, for instance with students sending non-sense messages for impacting the on-task/off-task indicator:

```
févr. 06 15:20:38 in ex1 tata says: is the weather good in Madrid? (il fait beau à Madrid ?)
févr. 06 15:20:46 in ex1 titi says: lol
```

6. Conclusion

Omega+ is one of the few fully generic system that support synchronous collaborative learning activities. A survey of the state of the art was already given in [10].

This paper focuses on interaction analysis that provides information directly to students and teachers in order to assess and self-regulate the ongoing activity, and describes the generic model-based approach proposed in Omega+. A specific model, called the 'Effect Model', specifies *how interaction analysis is customized for the specific learning situation* defined by the other models that parameterize the generic system. Interaction analysis mostly relies on a *generic* machine learning algorithm (NBC) and *ad-hoc patterns specification* and *recognition*. Efficiency and acceptance results presented in the previous section are encouraging, even if several aspects require further investigation and improvement such as the pattern definition language, heuristics for generating guidance messages, participation analysis, etc.

It is also sometimes objected that most indicators are about *collaboration* and not about *learning*. It is well established now that collaboration is not sufficient *per se* for producing learning outcomes. Specific kinds of knowledge-productive interactions such as *asking each other questions, explaining and justifying opinions and reasoning, reflecting upon knowledge* are necessary to foster learning. It is the reason why some of the proposed indicators measure *customizable ingredients that are required by these productive interactions* such as on-task messages, actions with accompanying textual utterances, and 'uptaken acts' [23].

References

1. Resta, P., Laferrière, T.: Technology in Support of Collaborative Learning. Educational Psychology Review, 19, 65—83 (2007)
2. Baker, M.J., Quignard, M., Lund, K., Séjourné, A.: Computer-supported collaborative learning in the space of debate. In: Proceedings of International Conference on Computer Supported Collaborative Learning (CSCL), Bergen, Norway, pp. 11—20 (2003)
3. Soller, A., Linton, F., Goodman, B., Lesgold, A.: Toward intelligent analysis and support of collaborative learning interaction. In: Proceedings of the Int. Conf. on Artificial Intelligence in Education, Amsterdam, Netherlands. IOS Press, pp. 75—82 (1999)
4. Suthers, D., & Jones, D.: An architecture for intelligent collaborative educational systems. In: Proceedings of Int. Conf. on artificial intelligence in education, Amsterdam, Netherlands. IOS Press, pp. 55—62 (1997).
5. Baker, M.J., & Lund, K.: Flexibly structuring the interaction in a CSCL environment. In: Proceedings of Euro AIED, Lisbon, Portugal. Edições Colibri, pp. 401—407 (1996).

6. Jaspers, J., Erkens, G., Kanselaar, G.: COSAR: Collaborative writing of argumentative texts. In: Proceedings of the Int. Conf. on Advanced Learning Technologies, Piscataway, NJ. IEEE Press, pp. 269—272 (2001)
7. Dimitracopoulou, A.: Designing Collaborative Learning Systems: Current Trends & Future Research Agenda. In: Proceedings of the International Conference on Computer Supported Collaborative Learning (CSCL), Taipei, Taiwan, pp. 115—124 (2005).
8. Suthers, D.: Technology Affordances for Intersubjective Learning: A Thematic Agenda for CSCL. In: Proceedings of the International Conference on Computer Supported Collaborative Learning (CSCL), Taipei, Taiwan, pp. 662—671 (2005)
9. Lipponen, L.: Exploring foundations for computer-supported collaborative learning. In: Proceedings of the International Conference on Computer Supported Collaborative Learning (CSCL), Boulder, Colorado, pp. 72—81 (2002)
10. Lonchamp, J.: Supporting synchronous collaborative learning: A generic, multi-dimensional model. International Journal of CSCL, 1, 2, 247—276, (2006)
11. Jermann, P.: Computer Support for Interaction Regulation in Collaborative Problem-Solving. Doctoral Dissertation, University of Geneva (2004)
12. D. Suthers, J. Xu, J.: Kukakuka: An Online Environment for Artifact-Centered Discourse. In: Proceedings of 11th WWW Conference, pp. 472—480 (2002)
13. Dillenbourg, P., Traum, D.: Sharing solutions: persistence and grounding in multi-modal collaborative problem solving. Journal of the Learning Sciences, 15, 1, 121—151 (2006)
14. De Chiara, R., Di Matteo, A., Manno, I., Scarano, V.: CoFFEE: Cooperative Face2Face Educational Environment. In: Proceedings of the 3rd International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2007), New York, USA (2007)
15. Lonchamp, J.: Floor Control in Complex Synchronous CSCL Systems. In: Proceedings of the International Conference on Web Information Systems and Technology (WebIST), Barcelona, Spain (2007)
16. Clark, H., Schaefer, E.: Contributing to Discourse. Cognitive Science, 13, 259--294 (1989)
17. Lonchamp, J.: Linking Conversation and Task Objects in Complex Synchronous CSCL environments. In: Proceedings of the International Conference on Web Information Systems and Technology (WebIST), Barcelona, Spain (2007)
18. Dimitracopoulou, A. et al.: State of the Art on Interaction Analysis: "Interaction Analysis Indicators". ICALTS Project Deliverable: D.26.1 (2004) http://telearn.noe-kaleidoscope.org.
19. Ingram, A.L., Hathorn, L.G.: Methods for Analyzing Collaboration in Online Communications. In: Roberts, T.S. (ed.) Online collaborative learning: theory and practice, pp. 215—241. Idea Group Inc, Hershey (2004)
20. Nonaka, I., Takeushi, H.: The knowledge-creating company: How Japanese companies create the dynamics of innovation, pp. 61—85. New York, Oxford University Press (1995)
21. Mitchell, T.: Machine Learning. McGraw Hill (1997)
22. Lovins, J.B.: Development of a stemming algorithm. Mechanical Translation and Computational Linguistics, 11, 22—31 (1968)
23. Suthers, D.: Implications of Shared Representations for Computational Modeling. In: Proceeding of the 2nd International Workshop on Designing Computational Models of Collaborative Learning Interaction, Maceió, Brazil, pp. 42—52 (2004)
24. Anjewierden, A., Kollöffel, B., Hulshof, C.: Towards educational data mining: Using data mining methods for automated chat analysis to understand and support inquiry learning processes. In: International Workshop on Applying Data Mining in e-Learning, Crete, pp. 27—36 (2007)