Customizable Computer-based Interaction Analysis for Coaching and Self-regulation in Synchronous CSCL Systems

Jacques Lonchamp

Lonchamp, J. (2008). Customizable Computer-based Interaction Analysis for Coaching and Self-regulation in Synchronous CSCL Systems. *Educational Technology & Society*, 13 (2), 193–205.

Customizable Computer-based Interaction Analysis for Coaching and Self-regulation in Synchronous CSCL Systems

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

ABSTRACT

Computer-based Interaction analysis (IA) is an automatic process which aims at understanding a computer-mediated activity. In a CSCL system, computer-based IA can provide information directly to learners for self assessment and regulation and to tutors for coaching support. This article proposes a customizable computer-based IA approach for a generic synchronous CSCL system, i.e., a system that can be customized for different learning tasks and different ways of performing these tasks. In a generic system, IA mechanisms must also be generic. In the proposed system, called Omega+, a specific submodel specifies the properties and rules that customize the IA process for the learning situation defined by three other submodels (process, interaction, and artifact submodels) that all parameterize the generic kernel. The feasibility of such a generic model-based IA approach is assessed, together with the efficiency of some of its underlying mechanisms and its global acceptance by users.

KEYWORDS

CSCL, Computer-based interaction analysis, Customization, Tutoring, Self-regulation

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 (Resta & Lafferière, 2007). There exist two main approaches for supporting collaborative learning (Jermann et al., 2001). The first one structures the situation in which the collaboration takes place: the task (e.g., with a learning script), the group of learners (e.g., with specific roles), the interaction (e.g., with sentence openers), and the artifacts. The second approach involves structuring the collaboration itself through coaching and self-regulation: as the collaboration progresses the state of interaction is automatically evaluated by the system with respect to a desired state and remedial actions may be proposed. In this article, we focus on *the coaching and self-regulation approach as a complement to the structuring approach in the context of synchronous CSCL environments*, i.e., "same time/different places" or "same time/same place" systems. The coaching and self-regulation approach requires storing the stream of all relevant interaction events, computing *on demand* interaction analysis (IA) indicators that support tutoring activities, and computing *periodically* synthetic visual representations of IA indicators that support learners' self-regulation. The tutor and the system together contribute to guide the learners toward effective collaboration and learning.

The literature describes many computer-based IA indicators. They can be related to the cognitive, social and affective dimensions of interactions. But most of them are dependent of a specific learning activity or a specific learning environment (Dimitracopoulou et al., 2004). Building *customizable* IA tools is recognized as a promising research direction "*that could help to face the problems of restricted validity field of IA tools, the one of low powerfulness of IA output, as well as this of not fulfilment of various users' profiles*" (Dimitracopoulou et al., 2004). The idea of building more generic and customizable mechanisms is not restricted to the IA domain, but *can impact all aspects of CSCL systems*. During its first decade, CSCL researchers have produced a large number of *ad hoc systems* focusing 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., Baker et al., 2003), collaborative design (e.g., Soller et al., 1999), collaborative knowledge construction (e.g., Suthers & Jones, 1997), collaborative modeling (e.g., Baker & Lund, 1996) and collaborative writing (e.g., Jaspers et al., 2001). 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*" (Dimitracopoulou, 2005), "*reconfigurable,*

adaptive, offering collections of affordances and flexible forms of guidance" (Suthers, 2005), "*very flexible and tailorable*" (Lipponen, 2002). Two research streams, following either the *component-based approach* (e.g., De Chiara et al., 2007; Asensio et al., 2004) or *the model-based approach* (e.g., LAMS, 2009; Ronen et al., 2006; Bote-Lorenzo et al., 2004), aim at meeting these expectations. The system described in this article, called Omega+, follows the model-based approach: an *explicit model parameterizes a generic kernel* for flexibly supporting different kinds of collaborative applications (Lonchamp, 2006). 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*). It must be emphasized that building a fully generic synchronous CSCL system is a real scientific challenge whose feasibility is still questionable. COFFee (www.coffee-soft.org), the first open source synchronous CSCL environment which provides basic customization capabilities has been released only a few months ago.

Our experiments with an early version of Omega+ have demonstrated that it is very difficult for a tutor to evaluate and improve the quality of the interactions between participants in particular when several groups of collocated students are working in parallel (while evaluating the artifacts they produce is not harder than in individual paper-based exercises). Some coaching support is strongly required. Supporting learners' self assessment and regulation is important not only in distributed settings when there is no tutor but also as a complement to tutoring activities.

In a generic system, IA mechanisms *must also be generic*, i.e., *model-based* in the case of Omega+. A specific submodel, called the "Effects Model", specifies the *properties and rules that customize the IA process* for the learning situation defined by three other submodels (process, interaction, and artifact submodels) that all parameterize Omega+ generic kernel. The central question is again the concrete feasibility of the approach. Thus, the major part of this article explains the kind of model and mechanisms that can be proposed. But this description also shows to educators concrete examples of high level services which could help them to solve the problems they face when teaching in online environments.

The remainder of the article is organized as follows. The second section depicts Omega+ overall approach for modeling synchronous collaborative knowledge building activities. The third section discusses the main characteristics of the proposal. The central question that is addressed is the *feasibility* of a *generic computer-based IA approach for synchronous CSCL systems*, i.e., automatic IA mechanisms which can be customized for different learning tasks and different ways of performing these tasks. The fourth section aims at evaluating some efficiency aspects of the proposed mechanisms on the one hand, and discussing the global acceptance of the system by users on the other hand.

Modeling Synchronous Collaborative Knowledge Building Activities - Omega+ Approach

An important requirement for effective synchronous collaborative learning is the *combination of communication with shared work artifacts* (Suthers & Xu, 2002). Most synchronous CSCL systems follow the *dual interaction spaces (DIS) paradigm* (Dillenbourg & Traum, 2002), by providing two distinct spaces of interaction. The *task space* allows for the collaborative construction and manipulation of shared artifacts. The *communication space* is the place where dialogue-based interaction, mostly textual, takes place. Several recent DIS systems provide multi tools task spaces for manipulating simultaneously complementary artifacts or perspectives (e.g., De Chiara et al., 2007). Figure 1 summarizes Omega+ conceptual approach of collaborative learning based on artifact mediation which extends the model proposed by Miao (2000) with such multiple views. "Representation" and "exploration" information flows can be related to Nonaka and Takeuchi's (1995) "externalization" and "internalization" processes. Knowledge in individuals' minds and the information which is held in the shared artifact can each be defined as being in "conflict" or "coherent" state. The term "conflict" is used to describe all situations that can trigger a reaction from learners, including inconsistency, incompleteness, contradiction, and impreciseness. At the individual level, learners construct new knowledge by integrating new information into their own cognitive structure. Conflicts occur when the new information contradicts existing knowledge. Learners must therefore solve this cognitive dissonance (Festinger, 1957) by constructing a new cognitive structure. At the group level, conflicts occur when one or more learners disagree with existing information in the shared workspace (inter or intra-view conflict). In this case, learners in the group might negotiate together to construct a new consensual state. This dialogue can trigger learning mechanisms.

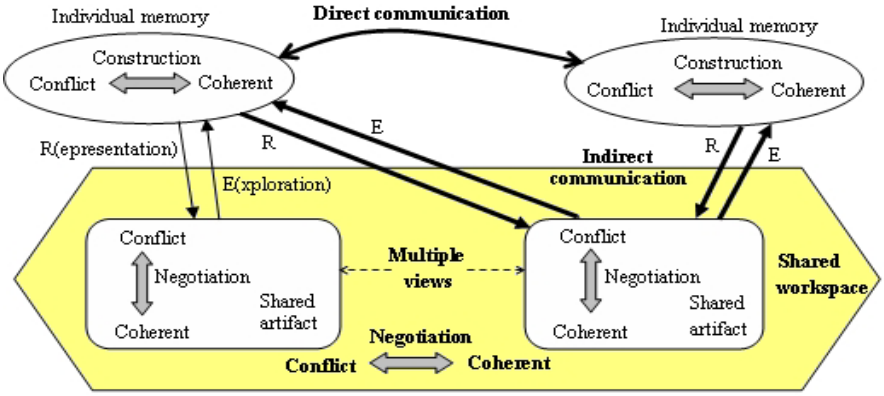


Figure 1. Omega+ conceptual approach

In a non-trivial CSCL application, the learning task is structured into a *process* including a sequence of phases where collaborative learning based on artifact mediation takes place. 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 to the same task in the same room, and "split phases", where participants are divided into parallel sub groups performing different tasks in different rooms. The corresponding process model, often called "macro-script" in the CSCL field (Tchounikine, 2008), is a plan which can be modified during its execution. It 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 another existing phase ("Jump"). As depicted by Figure 2, each phase type is characterized by a set of role types, a set of tool types for constructing the artifacts and a *floor-control policy* (FCP) at the environment level (Lonchamp, 2007a) that defines who can manipulate which space ("who has the floor in which space"). Application-specific *interaction protocols* can be defined through a set of application-specific roles, a set of typed messages (speech acts) and a set of adjacency pairs (Clark & Schaefer, 1989) specifying how message types are related (e.g., a question is followed by answers) and which role can speak first. *Application-specific FCPs* can use application-specific interaction protocols (see the "based_on" relationship in Figure 2) for controlling who has 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 2).

More generally, all important concept types (roles, tools, protocols, and FCPs) are specialized into *predefined* and *application-specific* subtypes. This reflects the fact that Omega+ provides both "hard-coded" features (that cannot be changed) and model-based features that are customizable and evolvable by users. More details about the models are given in (Lonchamp, 2006). Three submodels are highlighted in Figure 2, 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 "Effects Model". This submodel, at the heart of the generic IA approach, will be described in the following sections. The four submodels serve as four *parameters* for the generic Omega+ kernel. By providing *ad hoc* submodels, users can tailor the kernel to their specific needs (*definitional malleability*). Moreover, the behavior of the customized system depends on these continuously queried models and can dynamically evolve when a model is modified (*operational malleability*). System customization can be performed in different ways, adapted to the skills and needs of different categories of users: by reusing existing combinations of submodels, by building new combinations of existing submodels (i.e., following a very high-level configuration process), by creating or customizing submodels through high-level visual languages or low-level specification languages, including programming languages.

Figure 3 shows Omega+ client customized for a collaborative process for learning object-oriented analysis with the UML language. 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. The task space on the left may contain up to three tools as requested by the process model definition. Tools are either predefined editors

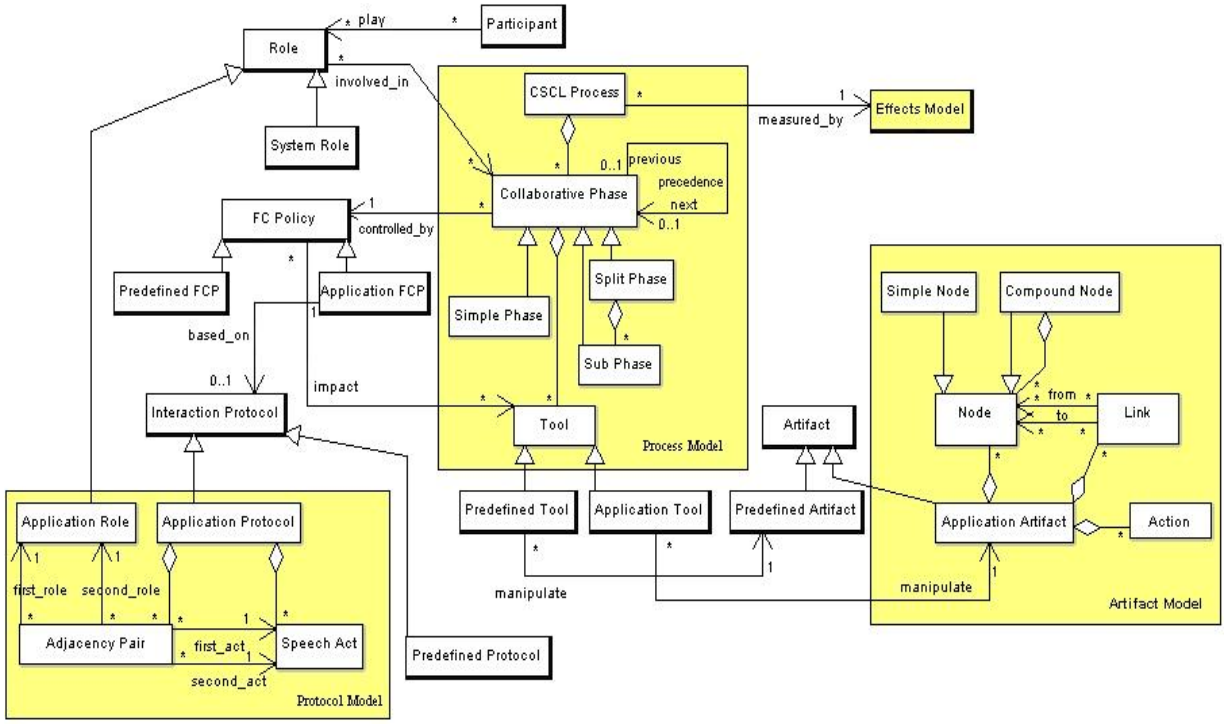


Figure 2. Omega+ conceptual model

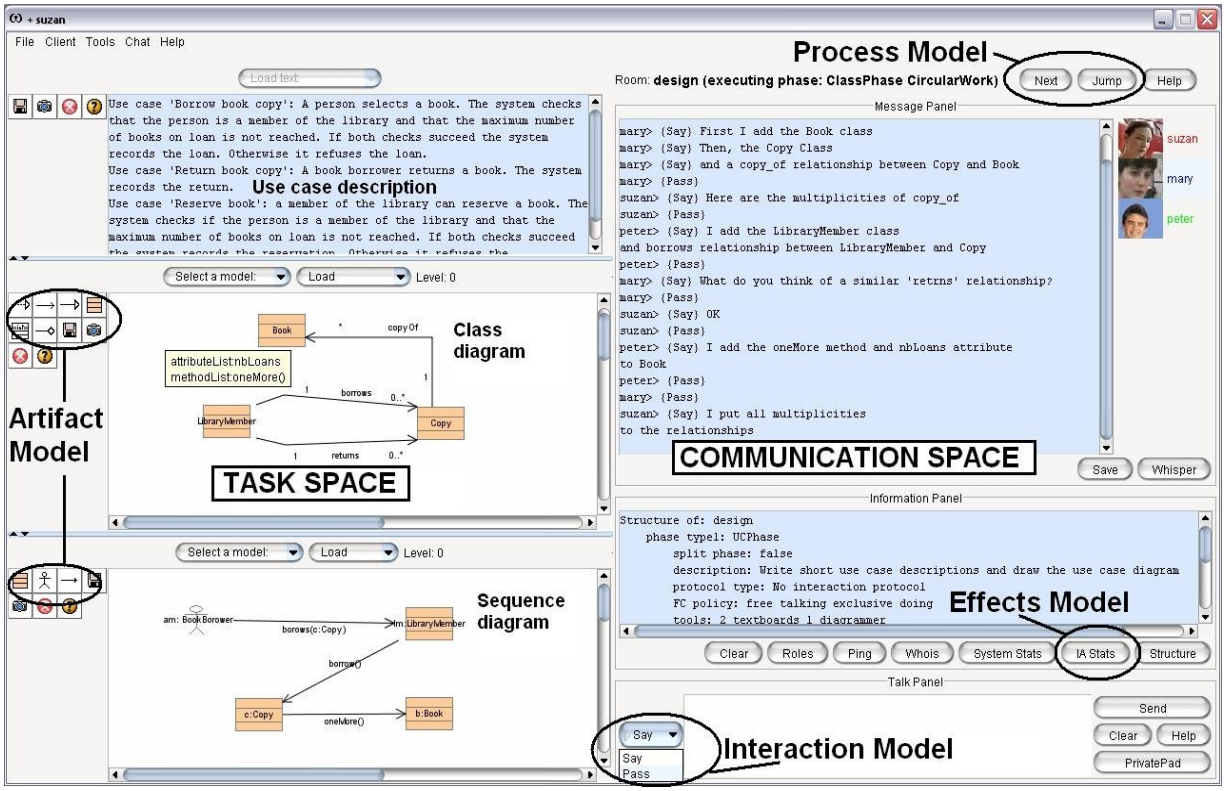


Figure 3. Omega+ client customized by the different submodels

(shared text editor, shared whiteboard) or shared graphical editors customized by artifact models. The task space on the left of Figure 3 includes: (1) A read-only text viewer − the colored background shows that interaction is not possible − which displays use cases (describing "who can do what" with the system under analysis) created during a previous collaborative phase. (2) Two instances of Omega+ generic visual editor customized with the UML class diagram model (describing the structure of the system under analysis) and the UML collaboration diagram model (describing how the components collaborate for every system operation). During this phase students collaborate for building these two diagrams. The model-driven "Circular Work" protocol controls the floor in both spaces in a circular order. A learner can explicitly "pass the floor" to the next one in the circle with a "Pass" message (see the drop-down list at the bottom of the communication space).

Omega+ provides several additional mechanisms for supporting learners at the cognitive and meta cognitive level, like sticky elements ("sticky annotated snapshots", "sticky notes" and "persistent pointers") for *referencing purposes* (Lonchamp, 2007b) and a tool for *collaborative session history browsing* (Lonchamp, 2009).

Computer-based IA Supporting Participants' self-regulation and Tutors' coaching

The definition of IA indicators is based on the following reasoning. Collaboration is not sufficient *per se* for producing learning effects. Specific kinds of *knowledge-productive interactions* are necessary like explaining and justifying opinions or reasoning, asking each other questions, reflecting upon knowledge (Dillenbourg, 1999). It is very difficult to characterize in a generic way such complex interactions and to detect them automatically during a collaborative session. It would require a precise understanding by the machine of all exchanged messages. A more realistic goal might be to characterize and detect *specific "ingredients" of these knowledge-productive interactions* such as actions with accompanying on-task messages or inter-subjective reactions. It is the approach that is explored in this article.

We distinguish between two kinds of variables that are used for computing IA indicators: (1) *Predefined task-independent variables* which count simple events at the interface, like the number of messages, or measure direct properties associated to these events, like the message size. Task-independent indicators based on these variables have been frequently proposed and implemented (Dimitracopoulou et al., 2004). In Omega+, these indicators are defined in the Effects Model by a calculus formula and a specification of their presentation (e.g., time series or bar chart). (2) *Task-dependent variables* which are elaborated by *complex customizable mechanisms* such as text classifiers or pattern recognizers. Omega+ provides several of these generic mechanisms whose customization information is also specified into the Effects Model. The following subsection discusses a representative set of task-independent and task-dependent indicators well adapted to the DIS paradigm. In any learning situation teachers can select predefined task-independent indicators from a library, create new ones by combining task-independent variables, and customize task-dependent indicators.

Participation indicators, like the number of chat messages and the number of task actions, are important because collaboration cannot occur within a group unless there is roughly equal participation among its members (Ingram & Hathorn, 2004). If some participants do the main part of the work while others barely contribute, then the group is not truly collaborating. The *balance between conversation events and action events* is interesting for measuring a correct usage of both modalities by learners in a DIS system. Pure action without dialogue and pure dialogue without any action are not likely to occur. *Indicators about the communication style* can also be helpful. It is the case of the average size of produced messages because learners should externalize their ideas and thoughts in an elaborated form. It is also important to distinguish between *on-task* and *off-task messages*. 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. For counting on-task and off-task messages a customizable message classification mechanism is required. Moreover, interaction requires that group members *actively respond to one another, react and change their minds as the interaction progresses* (Ingram & Hathorn, 2004). The most straightforward approach for measuring interaction is to track event patterns that reflect significant *interaction episodes*: for instance, two learners who successively modify the same element (or closely linked elements) of the same shared artifact (Bollen et al., 2008). A second possible approach is to track *explicit referencing mechanisms usage* such as participants' names referencing, message numbers referencing, sticky elements creation and referencing. A third approach is to track event patterns showing an individual activity aiming at facilitating collaboration ("facilitation" episodes). For instance, when a learner changes something on a shared artifact and immediately after sends a task-related message,

hopefully containing some explanation. A customizable mechanism is required for interaction and facilitation pattern recognition. Table 1 summarizes this representative set of IA indicators. All of them are implemented in the current prototype.

Table 1. A representative set of IA indicators for DIS systems

IA Indicator	Definition	Type
Participation	Number of chat messages	TI (Task-independent)
	Number of task actions	TI
Discourse focus	Balance between on-task and off-task messages	TD (Task-dependent)
Discourse style	Average size of messages	TI
Conversation/action balance	Balance between messages and actions	TI
Collaborative interaction	Number of explicit referencing actions	TI
	Number of interaction patterns	TD
Collaborative attitude	Number of "facilitation" patterns	TD

Omega+ Generic Implementation

Global Architecture

Omega+ supports both tutors and learners. For tutors, a selection of IA indicators is displayed on demand thanks to the "IA Stats" button (see Figure 3). Indicators can be presented as (stacked) time series for visualizing their temporal evolution and (stacked) bar charts for comparing values of different learners (see Figure 4a). The Effects Model specifies some general parameters, such as the time interval between measures for time series, and the characteristics of the selected indicators: name, informal description, type (bar chart, time series), value labels, and calculus expressions. The examples below are given in the XML format of the Effects Model which can be easily decrypted.

Examples:

<TimeSeriesDelta ms="30000" />

<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" />

(sizeMess, nbMess, nbWhite, nbTextb, nbDiag are predefined task-independent variables computed by Omega+ generic kernel)

For learners, a "monitoring window" is periodically displayed (see Figure 4b). This window contains visual representations of the selected indicators (a green/red square represents a "good/bad value"), a global ranking based on all the indicators for motivating the participants and an explicit guidance/advice message associated to the indicator having the worse value. The Effects model contains a rule for each indicator which specifies the label, the associated message, and the threshold for deciding if the value is "good" or "bad".

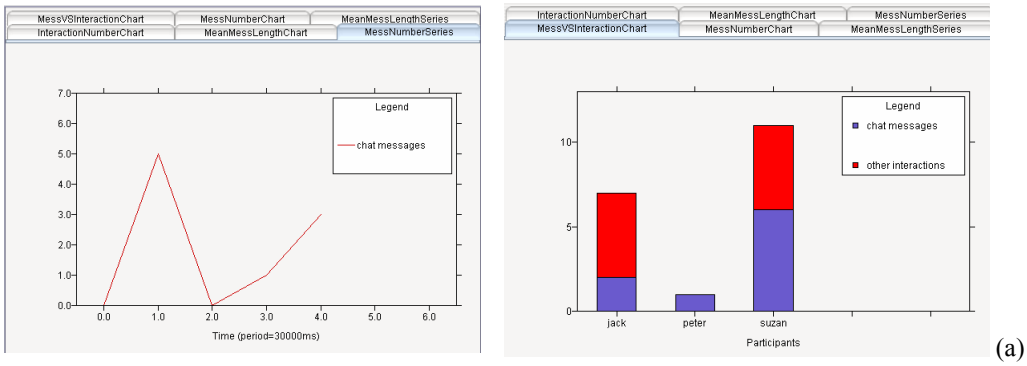
Example: the rule associated to the discourse focus indicator is

<Rule name= "DiscourseFocus" threshold="0.35" message="Your discourse should be more focused on the task!" />.

0.35 is the percentage of the mean value that must be at least attained. For each indicator, this value can be adjusted empirically to achieve the desired behaviour. If the threshold is set to a negative value the indicator is disabled.

The monitoring window is generated for each participant with a frequency specified in the Effects Model as a multiple of the time series delta (<MonitoringDelta nbTimeSeriesDelta="4" />). Figure 4b shows the monitoring window generated for Peter. Some indicators are well rated (two green squares) while others are weak (four red squares). Peter has the lower global score and receives an advice message about his discourse style ("Write more explicit messages!").

The next two subsections focus on the implementation of the customizable mechanisms that compute the task-dependent variables.



(a)

(b)

Figure 4. Coaching support (a) and self-regulation support (b)

Message Classification

A *Naive Bayes Classifier (NBC)* (Mitchell, 1997) is used for classifying messages into "on-task" and "off-task" categories. The NBC approach is one of the most effective probabilistic approaches 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. The classifier learns a set of probabilities from training data during a machine learning phase. Unlike other techniques like Latent Semantics Analysis, a large amount of textual input is not required. These probabilities and the Bayes theorem are used to classify any document. First, an estimate of the probability of this 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 files before or during the learning session. First, when the session starts, one (or several) file(s), explicitly referenced into the Effects Model are processed (e.g., <OnTaskFile file="usecase.txt" />). These files can contain for instance a textual description of a given diagram type or a summary of the instructions given to the learners. The classifier also analyzes during the session all the files loaded into the text board (e.g., the problem description submitted to the learners), all the meta-model files which serve as parameters for the generic diagram editor (they give in particular the names of all the concepts), and all the models created with the customized diagram editors (they give the names of all nodes and links created by the learners). Omega+ NBC also includes a "stemming" phase and a "stop words" removal phase. Stemming is the process by which words are reduced to their root forms (Lovins, 1968). For example, suffixes are removed, such as "-ing" and "-s", such that "digging" and "dig" become the same word. Stop words are words that occur frequently in the language, like "a", "and", "the" (e.g., 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 stemming algorithms and stop words files are provided in Omega+.

Pattern Recognition

In the current implementation, a very simple language is provided for specifying interaction (and facilitation) patterns. In the Effects Model, each pattern is defined by an XML expression:

<InteractionPattern actors="aaa" tooltype1="xxx" tooltype2= "yyy" tools= "ttt" condition="ccc" maxtime="mmm" />,

where (the vertical bar denotes a choice): aaa and ttt = same | different, xxx and yyy = chat | diagrammer | whiteboard | text board, ccc = none | a_condition_name, mmm = n | any.

This language is *extensible* as it is possible to create a condition for a particular situation by programming in java a dedicated filtering method having the same name than the condition in the "InteractionAnalysis" class of Omega+ generic kernel.

Examples:

<InteractionPattern actors="different" tooltype1="Diagrammer" tooltype2= "Diagrammer" tools="same" condition="sameobject" maxtime="60000" />

defines an interaction pattern where two different learners modify during the same minute the same object in the same diagrammer.

<InteractionPattern actors="same" tooltype1= "Diagrammer" tooltype2="Chat" tools="different" condition="ontask" maxtime= "30000" />

defines a facilitation pattern where the same learner acts on a diagram and sends an on-task message during the next 30 seconds.

Omega+ server keeps the complete history of chat messages and tool actions and checks all the patterns each time an event is inserted by filtering (with the condition method) the events belonging to the specified time interval (maxtime).

A Preliminary Evaluation Study

The evaluation 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, have received small case descriptions and were asked to build UML use case diagrams and UML class diagrams during 30 to 45 minutes length collaborative sessions with Omega+. Students were collocated (all in the same classroom) but could not speak for avoiding disturbing the other groups. It was hypothesized that Omega+ will stimulate task-related interactions. 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 has been recorded in the session 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.

During this evaluation study three different questions were addressed: the efficiency of the customizable mechanisms (classifier and pattern recognizer), the evaluation of the monitoring support by users and the global acceptance of the approach. The following subsections discuss each aspect in turn.

Message Classification Efficiency

The log file shows all the messages classified as "on task" messages by the automatic classifier.

Example:

janv. 30 15:25:27 in ex1 tata says: j'ai mis l'employé (I have put the employee)

janv. 30 15:25:27 on_task message

Table 2 gives the results obtained by the Bayesian classifier after a learning phase using a less than one page text file describing the two formalisms which are manipulated. The decisions of the classifier ("Classified" column) are compared with the decisions of a researcher who has analyzed the messages in the log file after the session ("Analyzed" column). For instance, 37% of messages are classified "on task" by both the classifier and the researcher. The accuracy was measured at 82% which is very satisfying with a so small machine learning phase. It

should be sufficient for characterizing *students who are not focused on the task* and for characterizing *accompanying explanations of actions*. Errors (18%) 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 mentioned", the word "function" is used instead of "use case" or "case" for 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 the graph reorganization", "reorganization" is stemmed into "organization" which is part of the actor concept definition in a use case diagram "an actor is a person, an organization or a system (…)"; the message is incorrectly classified as an "on-task" message.

Misspellings, compounding, abbreviations and initialisms ("answ" for answer), frivolous spellings ("okey") are other well-known difficulties (Anjewierden et al., 2007). Students were asked to avoid "chatspeak" and to spell and punctuate correctly.

Table 2. Message 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 %

Pattern Recognition Efficiency

The efficiency of the pattern recognition mechanism mainly depends on the efficiency of the filtering condition. If the library of predefined conditions does not contain the adequate method, users can customize an existing condition or write a new one from scratch in java. For each pattern, we have qualitatively analyzed all the cases where the pattern was recognized to assess the efficiency of the filtering condition. As statistical results would only reflect the efficiency of a specific set of filtering conditions the remaining of the section just discusses an example. Figure 4 gives an excerpt of a log file where 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. Each line of the log file includes the date and time, the name of the session ("ex1" in the excerpt), the name of the learner, the action type (e.g., "says" for a chat contribution, "performs a diagram action" for a diagram editor contribution). For a tool contribution, the line includes the tool type and number (e.g., "Diagrammer0") and a tool-dependent action description. This description starts with an action number, followed by an action type (e.g., "addVertex", "newProperties", "move"). In diagrams vertex have an internal identifier composed by the learner's name followed by a sequence number: "addVertex:Classe:titi5:" means that user titi has created a class ("Classe" in French) vertex identified by "titi5".

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: numPermisConduire||: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:|:

Figure 4. Excerpt of a log file with rule triggering

The two cases of pattern recognition may be analyzed in the following way. 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 reacts to the action of another student. In the second case, toto has added several properties to the node "titi3" (action 65). The rule was triggered because titi has moved the same node 26 seconds before, when he was changing the graph layout (actions numbered 56-60). In this case, *there is no semantic relationship between the two events* and it is hard to say that the two students collaborate. For improving the pattern recognition process it could be possible 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). Unfortunately, this kind of matrix cannot be fully generic: for instance, the position of a node is significant in a few artifacts and not significant in most others.

Users' evaluation

The monitoring window has been evaluated through qualitative interviews of learners. Personalized advice messages are considered as the most effective way of pushing information to them periodically. The global ranking is interpreted as a kind of "high score" which can increase the motivation to actively participate. At the opposite, most students find too complicated the analytic part which displays all the indicators.

For the coaching support, teachers have suggested a deeper analysis of users' participation because some learners can have a high participation score without a big impact on group performance. For example, in the session summarized in Table 3, one can observe that student1 is mainly talking, student2 is actively constructing the shared artifact and student3 spends much time for improving the graph layout by moving its nodes and edges. This will be further discussed in the conclusion.

Table 3. 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

User Acceptance

Globally, the *regulation approach generates much more debate and controversy than the structuring approach*. Structured processes, interactions, and artifacts are well accepted by students like classical pedagogical constraints. They do not require deep explanations. At the opposite, computer-based IA is criticized by some students as a "Big Brother" approach. The log file shows that some participants fight against the rules by sending nonsense messages for impacting the on-task/off-task indicator.
Example:

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

We can thus advise teachers to give precise explanations about the objectives and the implementation of the IA support. In blended learning settings we plan to provide a global supervision tool to the tutor where the information about all groups working in parallel will be centralized. The information will be verified by the tutor by browsing the session history and transmitted by him orally or electronically to the learners. It could be a way to *take benefit from automatic IA monitoring without these acceptance problems*.

Conclusion

This article focuses on computer-based IA which provides information directly to learners, in order to assess and self-regulate their activity, and to tutors, for coaching the participants. This approach can complement the more classical approach that structures the situation in which collaborative learning takes place. A generic model-based approach is proposed in the context of synchronous collaborative learning activities. A specific submodel, called the "Effects Model", specifies how IA has to be customized for the specific learning situation defined by the three other submodels that parameterize Omega+ generic environment kernel. This work demonstrates that a generic solution is feasible. It is possible with the proposed approach to customize the monitoring and coaching support to the situated learning task either by selecting predefined indicators or by creating *ad hoc* indicators, some of them relying on customizable mechanisms for machine learning and pattern recognition. We are not aware of any other generic computer-based IA approach.

The results of the preliminary study are encouraging in terms of efficiency of the customizable mechanisms and system acceptance. It is important for computer-based IA but also, more generally, in the perspective of building the flexible and tailorable systems the CSCL community is expecting for the future. The model-based generic approach is appealing in that it provides different ways to adapt the generic system to a given context at different levels of expertise: by just selecting and reusing existing models, by creating or customizing models through high-level visual languages or by enriching the underlying mechanisms through low-level specification languages, including programming languages. Therefore, all kinds of users, i.e., educators, instructional designers, and technologists, can contribute to system tailoring. However, the actual effects of the proposed IA mechanisms on teaching and learning remain unknown for a large part. For a generic environment like Omega+ it is very difficult to reach absolute conclusions because experiment results are always dependent of a specific learning situation and system customization.

An interesting research perspective is to study the relationships between *automatic computer-based IA* during the learning process and *manual IA by human experts* after the end of the process. They have different purposes (guiding vs. explaining) and follow different approaches. In the same way, a car dashboard displays critical information for *guiding* drivers (e.g., speed, engine temperature, oil pressure) which is very different from the information that human experts need for *explaining* car crashes (e.g., topological, psychological, medical parameters). However, *a high-level interpretation of how learners behave at a socio-cognitive level,* resulting from a careful analysis of conversations and actions, could be of a great help for better customizing computer-based IA indicators. For instance, a participation indicator will be more accurate if it mainly focuses on the *most important actions* whose characterization is dependent from such a socio-cognitive analysis. We have recently proposed a *generic framework for post-analyzing synchronous collaborative learning processes* (Lonchamp, 2009) and we plan to study how this could *improve the customization process* that is required by the technical approach described in this article.

References

Anjewierden, A., Kollöffel, B. & Hulshof, C. (2007). Towards educational data mining: Using data mining methods for automated chat analysis to understand and support inquiry learning processes. In Romero, C., Pechenizkiy, M., Calders, T., Viola, S. & Van Assche, F. (Eds.) *International Workshop on Applying Data Mining in e-Learning* (pp. 27-36). Retrieved October 27, 2009, from http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-305/

Asensio, J., Dimitriadis, Y., Heredia, M., Martinez, A., Alvarez, F., Blasco, M. & Osuna, C. (2004). Collaborative Learning Patterns: Assisting the Development of Component-Based CSCL Applications. In Cremonesi, P. (Ed.) *Proceedings of 12th Euromicro Conference on Parallel, Distributed and Network-Based Processing* (pp. 218-224). New York: IEEE Press.

Baker, M.J., Quignard, M., Lund, K. & Séjourné, A. (2003). Computer-supported collaborative learning in the space of debate. In Wasson, B., Ludvigsen, S. & Hoppe, U. (Eds.) *Proceedings of 5thInternational Conference on CSCL* (pp. 11-20). Dordrecht: Kluwer Academic Publishers.

Baker, M.J. & Lund, K. (1996). Flexibly structuring the interaction in a CSCL environment. In Brna, P., Paiva, A. & Self, J. (Eds.), *Proceedings of the Euro-AIED Conference* (pp. 401-407).Lisbon: Edições Colibri.

Bollen, L., Giemza, A. & Hoppe, U. (2008). Flexible analysis of user actions in heterogeneous distributed learning environments. In Dillenbourg, P. & Specht, M. (Eds.) *Proceedings of Third European Conference on Technology Enhanced Learning* (pp. 62-73). LNCS 5192, Berlin: Springer.

Bote-Lorenzo, M. L., Hernández-Leo, D., Dimitriadis, Y., Asensio-Pérez, J. I., Gómez-Sánchez, E., Vega-Gorgojo, G. & Vaquero-González, L. M. (2004). Towards reusability and tailorability in collaborative learning systems Using IMS-LD and Grid Services. *Advanced Technology for Learning, 1*(3), 129-138.

Clark, H. & Schaefer, E. (1989). Contributing to Discourse. *Cognitive Science, 13*, 259-294.

De Chiara, R., Di Matteo, A., Manno, I. & Scarano, V. (2007). CoFFEE: Cooperative Face2Face Educational Environment. In Grudin, J. & Kellogg, W. (Eds.) *Proceedings of the 3rdInternational Conference on Collaborative Computing: Networking, Applications and Worksharing* (pp. 243-252). New York: IEEE Press.

Dillenbourg, P. & Traum, D. (2006). Sharing solutions: persistence and grounding in multi-modal collaborative problem solving. *Journal of the Learning Sciences*, *15*(1), 121-151.

Dillenbourg, P. (1999). What do you mean by collaborative learning? In P. Dillenbourg (Ed) *Collaborative learning: Cognitive and Computational Approaches* (pp. 1-19). Oxford: Elsevier.

Dimitracopoulou, A. (2005). Designing Collaborative Learning Systems: Current Trends & Future Research Agenda. In Koschmann, T., Suthers, D. & Chan, T.W. (Eds.) *Proceedings of the 6thInternational Conference on CSCL* (pp. 115-124). Mahwah, NJ: Lawrence Erlbaum Associates.

Dimitracopoulou, A. et al. (2004). *State of the Art on Interaction Analysis: Interaction Analysis Indicators*. ICALTS Project Deliverable: D.26.1. Retrieved October 27, 2009, from http://www.rhodes.aegean.gr/ltee/kaleidoscope-icalts/Publications/D1 State of the Art Version_1_3 ICALTS_Kal NoE.pdf

Festinger, L. (1957). *A Theory of Cognitive Dissonance*. Stanford, CA: Stanford University Press.

Ingram, A.L. & Hathorn, L.G. (2004). Methods for Analyzing Collaboration in Online Communications. In Roberts, T.S. (Ed.) *Online collaborative learning: theory and practice* (pp. 215-241). Hershey: Idea Group Inc.

Jaspers, J., Erkens, G. & Kanselaar, G. (2001). COSAR: Collaborative writing of argumentative texts. In Okamoto, K, Hartley, R., Kinshuk & Klus, J.P. (Eds.) *Proceedings of the First International Conference on Advanced Learning Technologies* (pp. 269-272). Piscataway, NJ: IEEE Press.

Jermann, P., Soller, A. & Muehlenbrock, M. (2001). From mirroring to guiding: A review of the state of art technology for supporting collaborative learning. In Dillenbourg, P., Eurelings, A. & Hakkarainen, K. (Eds.) *Proceedings of the First European Conference on CSCL* (pp. 324-331). Retrieved October 27, 2009, from http://www.ll.unimass.nl/euro-cscl/Papers/197.pdf

LAMS (2009). Retrieved October 27, 2009, from http://www.lamsinternational.com

Lipponen, L. (2002). Exploring foundations for computer-supported collaborative learning. In Stahl, G. (Ed.) *Proceedings of the 4thInternational Conference on CSCL* (pp. 72-81). Mahwah, NJ: Lawrence Erlbaum Associates.

Lonchamp, J. (2009). A three-level analysis of collaborative learning in dual interaction spaces. *International Journal of CSCL*, *4*(3), 289-317.

Lonchamp, J. (2007a). Floor Control in Complex Synchronous CSCL Systems. In Cordeiro, J. & Hammoudi, S. (Eds.) *Proceedings of the 3rd International Conference on Web Information Systems and Technology* (pp. 397-402). Setubal, Portugal: INSTICC Press.

Lonchamp, J. (2007b). Linking Conversation and Task Objects in Complex Synchronous CSCL environments. In Cordeiro, J. & Hammoudi, S. (Eds.) *Proceedings of the 3rd International Conference on Web Information Systems and Technology* (pp. 281-288). Setubal, Portugal: INSTICC Press.

Lonchamp, J. (2006). Supporting synchronous collaborative learning: A generic, multi-dimensional model. *International Journal of CSCL*, *1*(2), 247-276.

Lovins, J.B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, *11*, 22-31.

Miao, Y. (2000) *Design and Implementation of a Collaborative Virtual Problem-Based Learning Environment*. Doctoral Dissertation, Technischen Universität Darmstadt.

Mitchell, T. (1997). *Machine Learning*. New York: McGraw Hill.

Nonaka, I. & Takeushi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation.* New York: Oxford University Press.

Resta, P. & Laferrière, T. (2007). Technology in Support of Collaborative Learning. *Educational Psychology Review*, *19*, 65-83.

Ronen, M., Kohen-Vacs, M. & Raz-Fogel, N (2006). Adopt & adapt: structuring, sharing and reusing asynchronous collaborative pedagogy. In S. Barab, K. Hay & D. Hickey (Eds.) *Proceedings of 7th International Conference of the Learning Sciences* (pp. 599-605). London: Routledge.

Soller, A., Linton, F., Goodman, B. & Lesgold, A. (1999). Toward intelligent analysis and support of collaborative learning interaction. In Lajoie, S.P. & Vivet, M. (Eds.) *Proceedings of the 9th International Conference on Artificial Intelligence in Education* (pp. 75-82). Amsterdam: IOS Press.

Suthers, D. (2005). Technology Affordances for Intersubjective Learning: A Thematic Agenda for CSCL. In Koschmann, T., Suthers, D. & Chan, T.W. (Eds.) *Proceedings of the 6th International Conference on CSCL* (pp. 662-671). Mahwah, NJ: Lawrence Erlbaum Associates.

Suthers, D. & Xu, J. (2002). Kukakuka: An Online Environment for Artifact-Centered Discourse. In Lassner, D (Ed.) *Proceedings of 11th WWW Conference*. Retrieved October 27, 2009, from http://www2002.org/CDROM /alternate/252/index.html

Suthers, D. & Jones, D. (1997). An architecture for intelligent collaborative educational systems. In du Boulay, B. & Mizoguchi, R. (Eds.) *Proceedings of 8th World Conference on Artificial Intelligence in Education* (pp. 55-62). Amsterdam: IOS Press.

Tchounikine, P. (2008). Operationalizing macro-scripts in CSCL technological settings. *International Journal of CSCL*, *3*(2), 193-233.