Simulation Modelling Practice and Theory 19 (2011) 84–97

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Enhancing synchronous collaboration by using interactive visualisation of modelled traces

Damien Clauzel *, Karim Sehaba, Yannick Prié

Université de Lyon, CNRS, Université Lyon 1, LIRIS, CNRS UMR5205, F-69622, France

ARTICLE INFO

Article history:
Available online 7 July 2010

Keywords:
Modelled trace
Activity modelling
Interactive visualisation
Interaction mode
Collaborative learning environment

ABSTRACT

This article addresses issues related to traces modelling for formally describing human interactions of people engaged in a synchronous collaborative learning activity. The objective is to propose models and tools for representing, transforming, sharing and visualising traces of users' experiences. The traces here represent the users' activities in their interactions with the learning platform. Our proposition is based on reflexive learning defined as the ability to interact with the situation, in order to meet one's own limitations. This work takes place in the ITHACA project which aims at developing an on-line learning platform that uses interaction traces as knowledge sources on, and for, the learners' learning as individuals or groups. In this paper, we propose a general framework for trace management and sharing, a generic model of synchronous collaborative activity based on the notion of interaction modes that we specialized for whiteboard sharing and text chatting, and a conceptual framework for modelling the exploitation of modelled traces, in particular for interactive visualisation on the user side. This article extends our previous work [1] on the instrumented prototypes, by presenting our theorisation of the interactive visualisation of modelled traces.

1. Introduction

Synchronous collaborative learning software is increasingly used in various teaching situations: discussion structuring [2], collaborative design [3,4], construction of knowledge [5], etc. Other environments aim to be generic, such as the Platine [6,7] or the Omeg+ [8] platforms. These environments provide a set of synchronous tools (chat, shared text editor, videoconference, whiteboard, etc.) and regulation mechanisms, such as control of speech turn, advanced referencing and group awareness. However, they do not provide tools for sharing experience and providing feedback to their users, despite the importance of such practices in learning. Indeed, as pointed out in [9], the challenge today is to provide technology-oriented dissemination of practices and experiences for effective collaborative learning.

The objective of our research is to propose models and tools for representation, treatment, sharing and visualisation of interaction traces in the context of a synchronous collaborative activity. The interaction traces are here defined as histories of users' actions collected in real time from their interactions with the software. The approach we advocate is to use the interaction traces as knowledge sources on, and for, the learners' learning as individuals (reflexive learning [10]) or as groups (collaboration, sharing and coordination). Indeed, the visualisation of traces will allow learners to use their own experiences, the results they produced, and the new knowledge they deduced. In this sense, Masciotra [11] stipulates that by such means,

* Corresponding author. Tel.: +33 0 95 089 00 89.
E-mail addresses: Damien.Clauzel@liris.CNRS.fr (D. Clauzel), Karim.Sehaba@liris.CNRS.fr (K. Sehaba), Yannick.Prie@liris.CNRS.fr (Y. Prié).
URLs: http://www.liris.CNRS.fr/membres?idn=dclauzel (D. Clauzel), http://www.liris.CNRS.fr/membres?idn=ksehaba (K. Sehaba), http://www.liris.CNRS.fr/membres?idn=yprie (Y. Prié).

1569-190X/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2010.06.021

a learner can ensure the relevance of his approach or readjust his actions. To assert this, we rely on *reflexive learning* aimed at improving student's competences. Such learning is defined as being directed, or turned back on itself, or self-referential.

We consider in our study two kinds of reflexivity. One is individual, and is the perception that a user has on his own activity. It is used for metacognitive processes that allow to understand strategies that might be used for different tasks, the conditions under which these strategies might be used and the extent to which the strategies are effective. For example, learners can know about different strategies for reading a textbook as well as strategies to check their comprehension. The other kind of reflexivity is group reflexivity through awareness, when members of a group want to have a high-level view on their actions; this is done through multiple sharing of different perceptions.

The principle of our approach is, in a first level (collection phase), to observe and store the user's actions in the form of modelled traces. At a second level (transformation phase), traces of meaningful high level to the user are calculated. These high-level traces can be exploited both:

- *in real time* in order to personalize the environment, to encourage collaboration, to increase adaptability within the learners' team, and to ensure awareness of each learner in learning space, and
- *afterwards* in order to provide a feedback on the learner's experience for quality improvement purposes and to enable learners to revise their action in order to fill gaps.

The work presented in this paper is part of our investigation within the ITHACA project[1] (Interactive Traces for Human Awareness in Collaborative Annotation). This project, by its multidisciplinary nature, aims at proposing models, architecture and tools for both the interactive visualisation of traces of a synchronous collaborative activity and the synchronous collaborative annotation of temporal documents (e.g. synchronous films co-annotation). In terms of application, the project focuses on distance learning of French language. In this article, we focus on aspects related to the interactive visualisation of modelled traces.

The article is organized as follows: Section 2 presents and discusses the theoretical foundation of our work. It consists in showing the contribution of traces reflection for learning. Section 3 presents the general architecture of our system. Section 4 presents our proposition for the characterisation of modelled traces. Section 5 details the model we have proposed to represent, to share and to visualise interaction traces in synchronous systems. The informatics framework for interactive visualisation of modelled traces is presented in Section 6. We present in Section 7 our first results, and finally Section 8 concludes and discusses future perspectives.

2. Reflexivity, awareness in synchronous learning systems

Reflexivity plays a central role in theoretical research on human learning, as shown in several studies (see for example [12]). According to [13], reflexivity is defined as the ability to interact with the situation in order to meet its own cognitive and socio-cognitive limitations. Through reflexivity, individuals can exercise control over their cognitive activity and actions, which allows individual and collective self-assessment and constructive criticism on oneself. In the context of human learning, reflexivity can facilitate appropriation and comprehension of the environment for complex tasks. In collaborative activities, synchronicity is one of the key elements that enable the development of reflexivity. Individual and collective reflexivities (specially needed in learning activities [14]) are used to build group awareness, which in turn reinforces synchronous collaboration [15] among participants.

Using the traces of the learner's activity is an effective way to encourage reflection on the learning process. This type of reflection, consecutive to the task called "reflective follow-up" [16], allows the learner to visualise traces of her actions and leads to awareness allowing meta-cognitive regulation. The difficulty with this approach is to detect, to trace, to model and to represent the meaningful actions of the learner [17]. Sherlock 2 [16] is an example of a system using this kind of reflexive incentives. Plaisant [18] used a system that graphically represents the actions performed by the learner using boxes and arrows. Despres and George [19] have developed a system based on traces allowing the tutor to perceive the status of learners' work. Clauzel et al. [20] have proposed a conceptual framework for tracking a learner's activity and attention in order to assist the user in his work. A reflexive method used in ergonomics is to use traces of the operators (via video) as a tool for construction of new knowledge by making the subject face its activity record. Nevertheless, it appears to us that:

1. studies on the reflexive usage of traces of learners' activity in learning environments do not cover the full extent of metacognitive activities that such traces allow;
2. traces have not been so far used as such for reflexivity in synchronous environments;
3. the systems that have been developed so far are ad hoc and lack the formal modelling of observables and traces, which would on the contrary allow rapid prototyping and exploration of innovative use of traces. To address such issues, we have proposed a general architecture for explicitly managing traces within the so-called Trace-Based Management Systems, which we will apply in the context of synchronous collaborative learning tools.

[1] http://liris.cnrs.fr/ithaca – this project is funded by the French National Research Agency (ANR), it features three labs: LIRIS (http://liris.cnrs.fr/), ICAR (http://icar.univ-lyon2.fr/) and TECFA (http://tecfa.unige.ch/) and the eLycèe company (http://www.eLycee.com/).

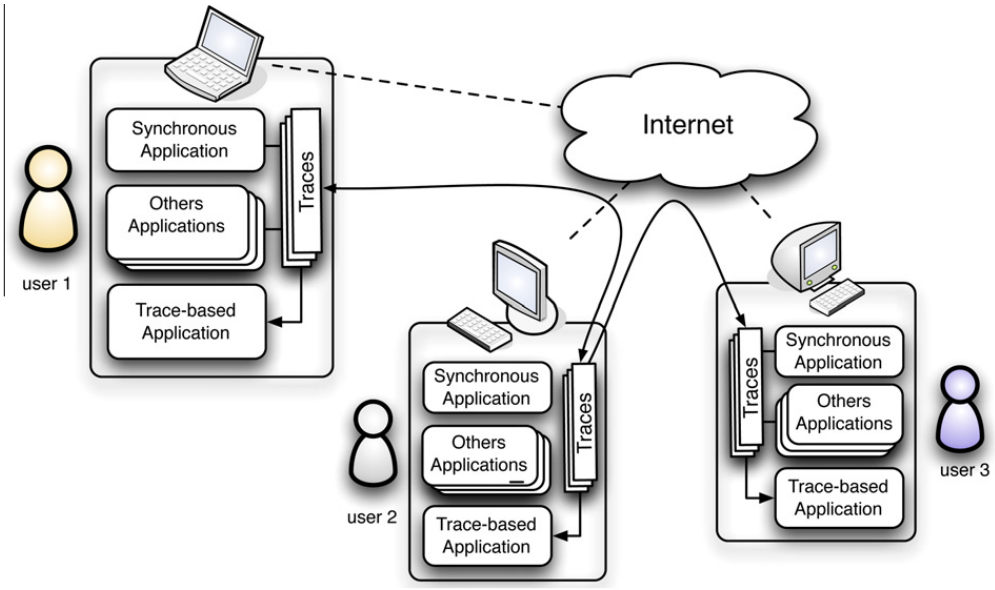

Fig. 1. Traces for individual and collective use.

As a way to facilitate the learners' monitoring in collaborative learning systems, modelled traces can be used to formally describe the learners' interactions with their computer systems and with other members of their activity group. This allows a tutor to know the functioning of his learners, how they progress on an individual and a collective base, and to do precise pedagogical interventions [21]. The models we present further down this article describe our way of formally describing generic collaborative learning activities, illustrated by an example model with its corresponding naive implementation.

3. General architecture for synchronous collaborative traces

3.1. Traces in synchronous collaboration

Our research team[2] has been working on traces for several years, building applications and studying various usages [22,23]. As illustrated in Fig. 1, our approach supposes that (1) some of the user's interactions with her applications are traced and (2) personal traces can be further reused within the so-called trace-based applications, providing individual services such as:

- *interactive visualisation*: the user can explore, query, annotate one's trace, for instance, for direct activity reflexivity (online), or for exploring one's past history (offline);
- *trace-based assistance*: for instance, the adaptation of the learning scenario.

The synchronous tools always already offer a basic native online group reflexivity that is related to the very scope of such tools (e.g. if somebody is writing on the whiteboard, what he writes is *intended* to be shared). There also exists a second kind of reflexivity, related to the extension of the application with "parallel" activity indicators (e.g. when Skype tells the user that "John is typing", it adds a sense of awareness of what John is doing apart from the chat message that will likely arrive on the screen). We want to go beyond this second kind of group reflexivity, by considering traces *as such* and *apart* from the main synchronous application. This will allow us to extend the use of one synchronous tool (1) with activity related to the tool itself (e.g. muting the sound can be part of the shared trace); (2) with activity related to other tools, be they asynchronous (e.g. sharing the use of a word processor during the session) or synchronous (e.g. extending one's whiteboard trace with part of one's visioconference activity).

For this, we consider as illustrated in Fig. 1 that a user can share and stream his traces to the trace bases of other users, who are then able to use shared traces plus their own personal traces within their trace-based applications. It becomes possible that the user will be aware of the activity of his group's members and to situate his activity within the group. Sharing of traces can also be symmetric or asymmetric, depending on the activity or the status of the users. For instance, user 1 and user 2 can fully exchange their activities as peers, while user 3 as a tutor could be aware of user 2 activity as a pupil, the reciprocity being false.

[2] SILEX (Supporting Interaction and Learning by Experience): https://liris.cnrs.fr/equipes?id=44&set_language=en.

3.2. Trace-Based Management Systems

The general goal of our team being to make traces first-class citizens of computer systems (as for instance files are), we had to define precisely what traces are and how they were to be manipulated. For that, in [24] we defined the notion of *Trace-Based Management Systems* (TBMS) as systems devoted to the management of *modelled traces*.

A modelled trace is a trace explicitly associated with its *trace model*. A trace model is an ontology that describes the vocabulary of the trace. A trace results from the observation of the interactions between a user and her system, it has a temporal extension related to the time of the observation. A trace is composed of *observed elements* (or *obsels*) representing the interaction between the user and the system. Each obsel has a set of attributes/values that are related to the temporal extension of the trace (e.g. it can be related to an instant or a temporal interval). As shown in Fig. 2, a trace can contain relations between obsels (e.g. T_4). A trace model is then a set of *observed element types* and *relations types*, as M_1 and M_2 respectively describe the obsels of T_1 and T_2.

Modelled traces are managed by *Trace-Based Management System* (TBMS). The process of *collecting* is that of creating a first modelled trace – called *primary trace* – from several sources. The traces can be used in various ways (visualisation, assistance, adaptation, etc.) within dedicated applications. These applications can take advantage of two main services provided by a TBMS. A trace-querying service is dedicated to retrieve traces from the trace base according to various criteria. More interesting is the *transformation service*, whose role is to operate transformations on traces. Indeed primary traces originating from the collecting may not have the right abstraction level for the target application (e.g. one wants to visualise a high-level trace showing the realization of "answering an exercise" instead of the low-level, primary trace describing "using a web browser"), or there may be traces from several applications that should be considered together, etc. The TBMS can then transform one or several traces according to a transformation τ resulting in a new trace in the base. Fig. 2 shows a primary trace T_1, transformed *by selection* into T_2 according to τ_1 and T_3 according to τ_2. T_2 is transformed *by rewriting* into T_3 according to τ_3. A transformation *by fusion* (see Fig. 3) consists in copying all the obsels of two or more traces into a new one.

A complete formalization of our metamodel proposal for traces models, traces, queries and transformations can be found with precise semantics in [25]. We are currently developing an open-source TBMS that implements such metamodel.

3.3. Synchronous collaborative traces

So as to adapt to the synchronous collaborative framework of the ITHACA project context, and to the uses we foresaw, we somewhat extended the notion of TBMS (see Fig. 3). At the architectural level, if users do have a personal TBMS for managing their own traces, they should also be able to manage other's shared traces. Inter-TBMS communication is then needed so as to be able from one side to share traces, and from the other side to collect shared traces. At the metamodel level, we also needed to be able to manage in one single trace base personal traces and other's traces. For that we introduced the notion

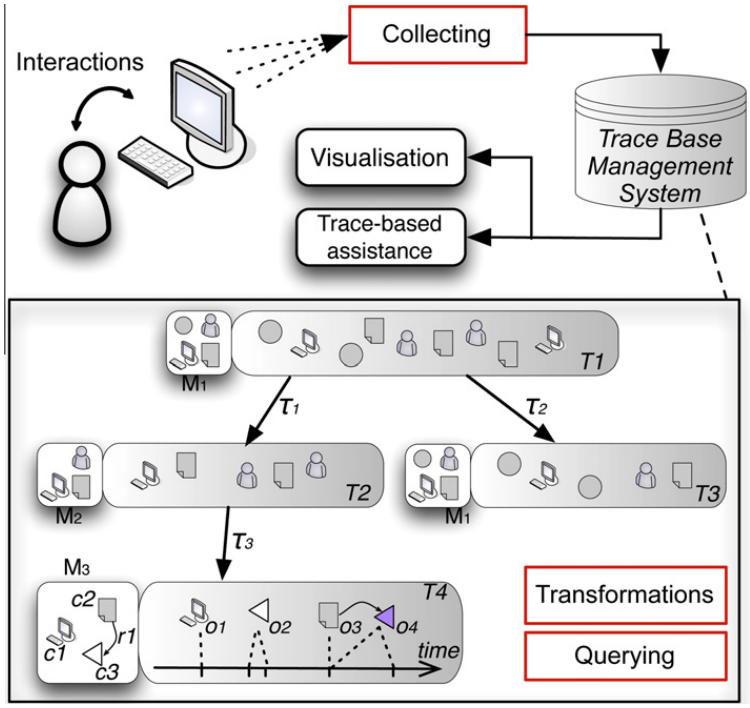

Fig. 2. A Trace-Based Management System framework for using modelled traces. The example trace $T3$ is somewhat detailed: the trace model M_3 contains three obsels types (c_1,c_2,c_3) and one relation type (r_1). T_3 contains four obsels: o_1 and o_3 are related to instants, while o_1 and o_4 are related to intervals. There is a relation between o_3 and o_4. A TBMS offers query and transformation services.

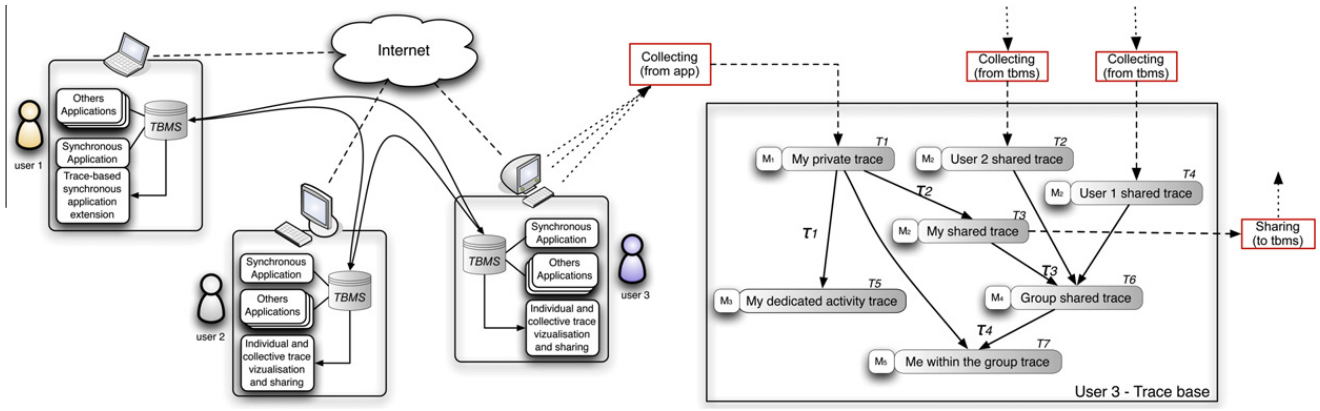

Fig. 3. Left: general architecture for individual and collective trace visualisation and sharing. Note that user 2 and user 3 have a separate trace-based application, while user 1 has a more integrated trace-based extension of the synchronous application. Right: the trace base of user 3. This base contains the primary trace T_1 of *user*3, which is abstracted/rewritten by τ_1 into the trace T_5 that is more adapted to the representation of a user 3 high-level activity. T_1 is also modified into T_3 by selecting obsels that user 3 wants to share. User 1 and user 3 have shared their traces, and a fusion transformation τ_3 is used to build a common "group trace". User 3 can then use τ_4 so as to build a trace adapted to the visualisation of her activity within that of the group.

of the *subject* of a modelled trace, who is the user that was observed during the collect. For instance, the subject of T_1, T_3 and T_5 (My private trace, My shared trace, My dedicated activity trace) is user3, while the subject of T_2 (User 2 shared trace) is *user*2. The subject of T_6 and T_7 is the triple (*user*1, *user*2, *user*3).

As it is not the main subject of this article, we will not go deeper into trace-based systems theory. We will neither address privacy issues related to trace exploitation, which is a complex question overcoming widely the scope of the work we present here. Let us just state that we are aware of the question and that we ensure in all our developments that the user be provided with full property and control of the diffusion of her trace. The remainder of the paper is devoted to presenting our trace models for synchronous collaboration and our first developments.

4. Characterisation of modelled traces

One of the finalities of collecting modelled traces from the user's activities is to loop back to him for providing support, guidance, automation, for short anything that can improve his computer-based work. Such exploitation of the modelled traces requires not only just the traces' models but also a trace exploitation framework that will allow the user to express his uses of his own traces. In this sense, among the exploitations of modelled traces that we consider in this work is the interactive visualisation of traces. In this context, we are particularly interested in determining the properties of the modelled traces that are important in the visualisation process. It consists in determining the way in which a trace can be visualised according to its characteristics. In this section, we present first the characteristics of the modelled traces we have defined. Then we describe the aspects related to the purpose of exploiting the modelled traces.

Fig. 4 describes the general architecture of a learning system using modelled traces, with emphasis on the interactive visualisation aspect. The learner interacts with his enhanced learning software. The software collects the users' interactions and sends those data to the trace-based management system. Inside the TBMS, the modelled traces are dispatched to the software components that uses the modelled traces. In our case, the modelled traces describing the user's interaction are sent to the trace-based assistant in charge or presenting those traces to the learner. The purpose being to support the learner in his analysis of his own activity. For doing this, the learner interacts with his assistant in order to construct an adequate representation of his activity's trace.

4.1. Elements of characterisation

We identify the following characteristics of a modelled trace. These characteristics are used for discriminating modelled traces and reasoning on their exploitation:

Number of obsels: How many obsels are present in the modelled trace. In terms of cognitive load (for visualisation) or manipulation (for scripting the computer environment), providing the user with just a few obsels is very different from giving him a lot of them.

Number of obsel types: How many types of obsels are present in the modelled trace. The more the type of obsels in a modelled trace, the more complex it is to process, to visualise, and to manipulate. Some trace exploitations require a very strict range of obsel types, because of the user profile (age, capacities, ...) or his activity.

Nature of obsels temporality: How the obsels are inscribed in time. There are two possible kinds of temporality for an obsel: instant and extended. An instant (or punctual) obsel is located at a precise moment in time, while an extended obsel covers a time span.

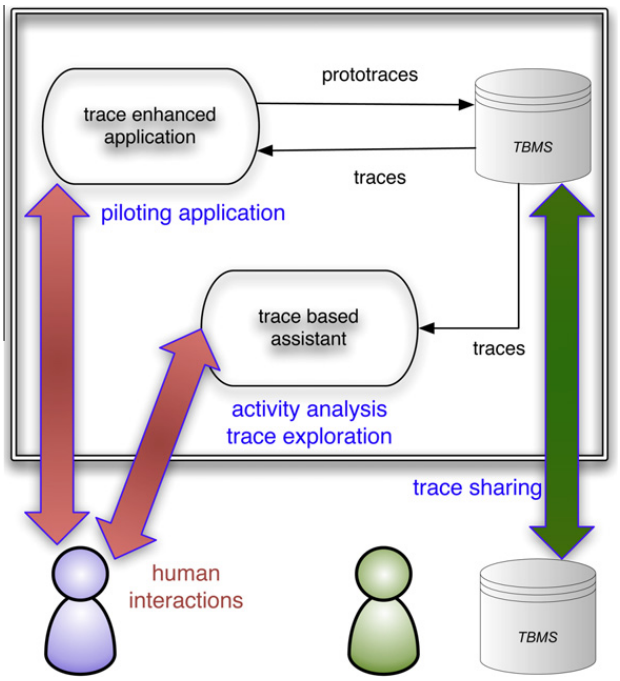

Fig. 4. Subpart of the global architecture, showing the interactive visualisation components.

Number of relations: How many relations are present in the modelled trace. This has an impact on the trace complexity, because for understanding and processing an obsel we have to analyse as well the other obsels that it is connected to.
Type of relation: Two possibilities here: a relation can be between two obsels of the same type, or between obsels of different types.
Number of relation types: How many types of relations are present in the modelled trace. The more the type of relations, the more needy is the process of a modelled trace.
Direction of relations: This characteristic is about the relations between obsels. Depending on the traced activity, we can have in a trace two kinds of directions for relations. The directions of the relations have a deep impact on the exploitation of a modelled trace. For example, in a trace in which all relations go toward the past, we know at any given time that no relations are left pending (that is, with only an origin and no destination because the closing obsel has yet to come). In a trace with mixed direction for relations or relations that go only toward the future, we have to handle the dangling relations and ensure that when the trace is closed all the relations are in a complete state.

Unidirectional: The relations go only toward the future or toward the past.
Mixed: The relations go in a combination of both directions.

Nature of the trace's temporality: The trace's temporality can of two kinds. One is contiguous, that is the obsels can be seen as belonging to a single time span. The other one is fragmented, and the trace contains episodes of obsels, separated by intervals. For example, a contiguous trace can be a trace of a work session for a learner, and a fragmented trace would be a trace describing a semester of a learner's work sessions. In the second case, the obsels will be grouped within the trace's temporality, according to the trace's time reference.
Nature of the trace's reference of temporality: For a given specification of a trace's temporality, we identify two aspects:

Time nature: The time nature of a trace can be natural or conceptual. Natural time is the time expressed by the usual human definition, where the conceptual time is defined by other mean within the activity (milestones in a process, event-based clocks, etc.)
Temporal reference; The temporal reference of a trace can be relative or absolute. Absolute compared to a time-located event (date, for example); relative to another obsel: before/after another obsel. In the second situation, we may know the relative position of the obsels within the trace, but not their corresponding location in time.

Length of trace: The time length of the modelled trace expressed within the time reference of the trace.
Nature of trace's subject: The nature of the trace can be of three kinds: unique (a single subject), complex (a group of individual) or mixed (a combination of individuals and group via subject's reification).
Nature of the number of subjects in the trace: At any given time within the modelled trace, the number of subjects can be different. The nature of this number has various consequences on further uses of the trace, because it allows us to eventually make some inference about the actors of a traced computer-based activity. The number of trace's subjects can be known or unknown, fixed or open.
Nature of obsel's attribute: The attributes of an obsel can be of any type, and not just text: image, sound, video, complex object (including another modelled trace). Therefore the nature of each attribute has a deep impact on the processing of the obsel, as some type of data can be opaque for a computer and only understandable by a human.

Nature of trace: When in the TBMS, a trace can have different possible origins and be classified as such:

Primary trace: We call "primary trace" as all the traces directly issued from collect. By definition, a primary trace need not be transformed.

Computed trace: We call "computed trace" as all the traces associated to an automatic transformation and to an ensemble source traces, automatically computed by the Trace-Based System and updated along the evolution of the source traces. These traces can be compared to the views of a traditional relational database system.

Stored trace: We call "stored trace" as all the traces stored into a Trace-Based System (by opposition to computed traces). A stored trace can be a primary trace, a manually transformed trace, or a computed trace that has been frozen in a state at a given time.

Conjoint trace: We call "conjoint trace" as all the traces having a multiple subject. A conjoint trace is generally constructed by temporal fusion of two traces having different subjects.

Enriched conjoint trace: We call "enriched conjoint trace" as a conjoint trace in which one of the subjects had been added to a conjoint trace. Therefore, it is a definition based on the enrichment process of a trace. A conjoint trace can only be recognized as enriched if we take into account the transformations that have been applied on it. The most classic case of enrichment is when one adds to a conjoint trace, constructed by fusion of two traces having for subjects x and y, some obsels about subject (x,y), or some relations between obsels having different subjects.

State of trace: A trace can be "open" or "closed". We call "closed trace" as a trace that is no more a target of obsels collection; an "open trace" is a trace that is still augmented by new obsels and relation.

4.2. Purpose of the exploitation

As a knowledge source, traces can be exploited by both human users and computer systems. Indeed, traces can be used as tools to control the interactive applications, to generate adaptive learning scenarios or to assist the learner in his/her learning activities. Our research is focused on the use of traces by human users. In this context, the interest of the exploitation is mainly on the production of resources/documents that help the user to understand the activity of the different actors of synchronous collaborative learning. This interest is both for designers, tutors and learners.

The trace allows the designer to detect the emergence of new uses made by the tutor and the learner, and to assess the suitability of a resource in a learning situation, e.g. the number of aid applications and assistance. The trace allows monitoring of the learner, e.g. the number and duration of consultations of the course. It also allows the assessment of the level of collaboration among learners, e.g. the number of messages posted on a forum.

The information contained in the interaction traces is part of the use of the learner. By having its own traces, the learner should be better able to take its environment and therefore adapt their work. Indeed, the visualised traces will allow learners to ponder the experiences they lead, the results they produce, and the knowledge they conclude. By this means, the learner can ensure the relevance of his/her approach or to readjust his/her actions.

In order to provide an adapted visualisation to each actors of the synchronous collaborative learning system, the system will represent the profile of each actor. In addition to the role of each actor in the learning system (tutor, designer or guardian), the profile should represent the preferences and knowledge of users. It is according to each profile as the visualisation or actions on the visualisation are adapted (some actions are valid only for designers, others only for tutors, etc.).

As a conclusion, the description of the modelled trace we presented in this section allows us to characterise each trace, so we can reason on what kind of treatment is doable on it, and how it will have to be done. Inasmuch as the model of a modelled trace allows us to reason on the trace's meaning, it does not inform us on the trace's content when it comes to the presence of obsels and relations. Both the model and the internal knowledge of the trace's content are necessary for further processing a modelled trace, specially when it comes to synchronous collaborative activities.

5. Trace models of synchronous collaborative activities

5.1. Interaction modes and tools

In its most generic aspect, we consider that a synchronous collaborative tool is a computer environment allowing a group of persons to realize an activity together and at the same time, while depending on each other. Such an environment can be composed of several software components that support group regulation, communication and production. Synchronous collaboration is supported by interactions happening in shared workspace, written discussion, video conferencing, etc.

To take into account the variety of synchronous tools and activities, we propose to define an "interaction mode" as a means for a user to interact with another user, as an established practice of interacting through a computerized channel.

We identify the following interaction modes in synchronous collaborative activity:

- Sharing a whiteboard: participants can draw, write, insert resources, etc. Example of a tool implementing such interaction mode: Dabbleboard.[3]

[3] http://www.dabbleboard.com, http://gobby.0x539.de, http://www.skype.com, http://icq.com/, http://www.eLycee.com/what_is_elycee/eMediatheque/, http://en.wikipedia.org/wiki/Virtual_Network_Computing.

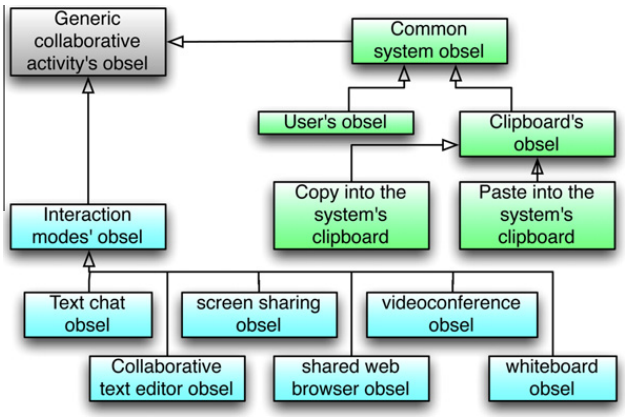

Fig. 5. Structure of the generic trace models for synchronous collaborative activities; specific obsels are not all detailed here.

- Collaboratively editing a text: sharing a writing area. Ex.: Gobby.[3]
- Videoconferencing. Ex.: Skype.[3]
- Text chatting. Ex.: Skype or ICQ.[3]
- Co-browsing: several users can engage in a common browsing session, sharing URI, pushing pages, etc.; Ex.: eMèdiathéque.[3]
- Screen sharing: remotely viewing and controlling a distant computer; Ex.: VNC.[3]

Of course, an interaction mode can be used in combination with other ones (e.g. videoconferencing and sharing a whiteboard). Also, there is not always a strong connection between a software and the interaction modes: an application can implement several interaction modes. For example, Skype (voice and video) instantiate a videoconferencing interaction mode, but also a text chatting interaction mode.

5.2. A generic model for synchronous collaborative activity traces

In Fig. 5, we introduce a generic model of traces in a synchronous collaborative activity, built upon a description of a generic synchronous collaborative environment. The purpose of this model is to propose a way to formally describe any synchronous collaborative activity. Our approach is based on a modular decomposition of the activity description: the model is composed of several sub-models related to interaction modes and one sub-model related to the whole activity.

The obsels are organized within a specialization hierarchy. At the top level is the generic obsel from a synchronous collaborative activity, describing that the user has made a temporally situated interaction within the traced computer-based environment.

There are two main parts in this generic model of the synchronous collaborative activity. The first one (bottom in Fig. 5) deals with the various categories of interaction modes. The second one (top right in Fig. 5) focuses on global interactions. It contains obsels for describing the participants of the synchronous collaboration, particularly the user and her actions that are not specific to a precise interaction mode, but global to her computer environment like copy and paste, etc. Such an approach gives us the possibility to express transmodal relations between obsels. For example, one can think of doing a copy from a text chat for pasting it onto a whiteboard. We designed our model such that neither the copy nor the paste interactions belong to a specific interaction mode, but belong to the common computer environment.

Each of the interaction modes is described, in a generic manner, by a specific interaction mode model. This model can be further specified for matching the precise feature of applications, and extending for supporting new interaction modes. We detail here two interaction modes: the *whiteboard sharing* model and the *text chatting* model.

5.2.1. Whiteboard sharing model

Fig. 6 shows our modelling of a generic whiteboard software. It allows us to describe the user's interactions with any kind of whiteboarding software. We identify two generic kinds of objects, "text" (such as typed by user) and "shape" (everything else); the generic actions being to create, to alter and to delete them.

The "content" obsels' attributes have complex types, specific to each whiteboard application. They contain data about objects such as position, shape, colour, and textual content.

The model is expandable. One can think for instance of extending it for integrating a semantic aspect, if the software allows it, with a new obsel *text correction* describing the action to fix a spelling mistake in a text, without altering its meaning, and a new relation *is linked to* linking together two connected elements, them being text or shape.

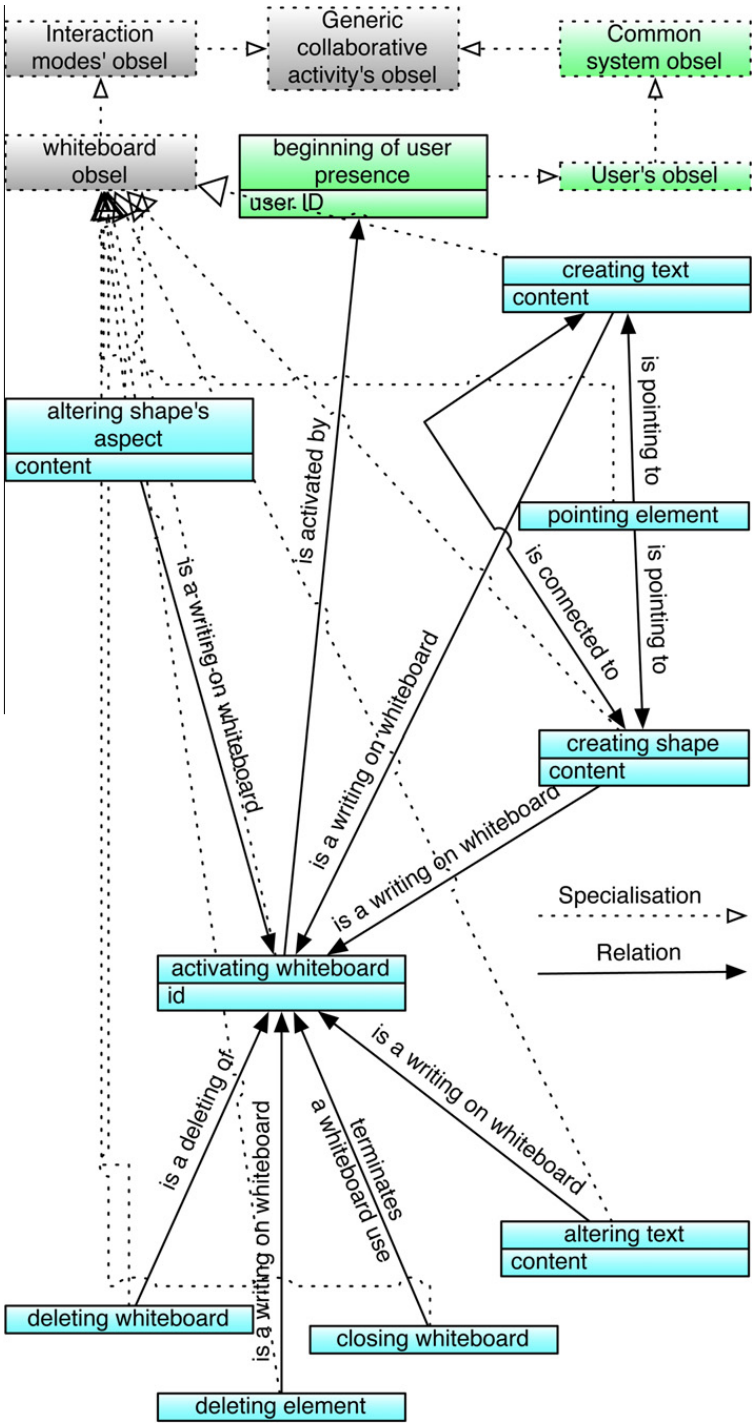

Fig. 6. Generic model of whiteboard user interactions; attributes are not all detailed here.

5.2.2. Text chatting model

Fig. 7 shows our modelling of a generic text chat software. It allows us to describe the user's interactions made in any kind of text chat software. The central obsel is *chat channel activation*; it represents a conversation channel for the user, and therefore contains in its attributes all the global informations about this precise conversation. The obsels in relation to this channel, such as sending and reception of messages, relay on it for contextualisation.

This model is also expandable. One can think to immediately expand it in order to add the concept of "conversation", with the relation *is an answer to* linking two messages, the second being a direct answer to the first one. Such an extension relies on being capable of automatically analysing the structure and content of a chat channel for inferring such relation.

We could also have added a relation describing the link between a user and a chat channel; but this relation is non-trivial because software or communication protocols do not always announce the user's presence, except when he is talking (before that, they are *invisible* for a newcomer). That is why we do not include this relation in our generic model.

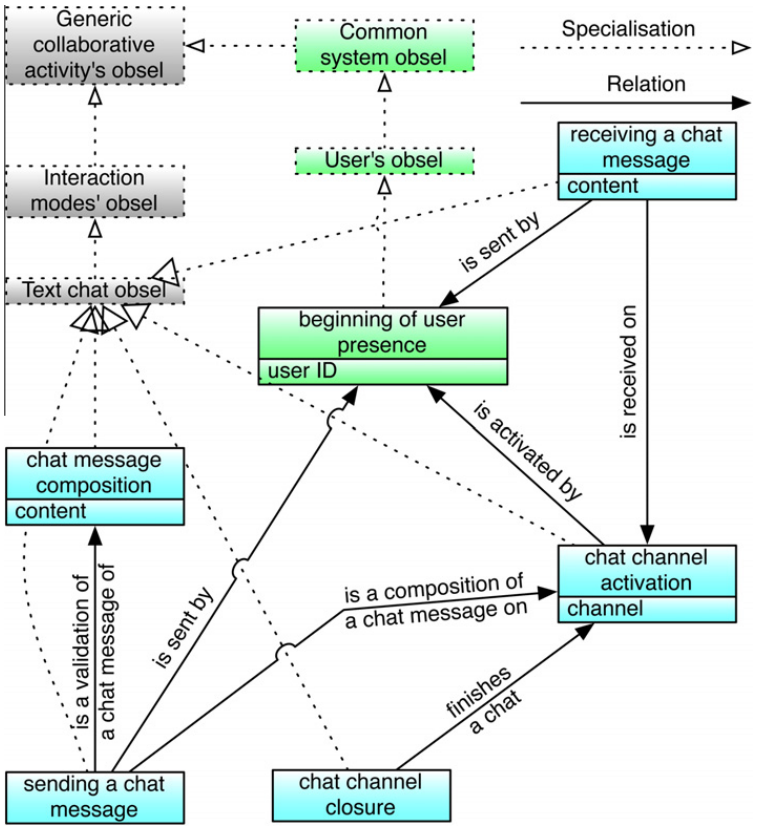

Fig. 7. Generic model of text chat user interactions, attributes are not all detailed here.

5.3. A synchronous trace example

As a concrete example, let us imagine the following situation: Alice, Bob and Charlie are three learners engaged in a synchronous collaborative activity. Their work is to search the web for precise informations, and to collect useful resources for future work. For doing this, they use a shared web browser. They have a chat for communicating, and the result of their searches is organized onto a shared whiteboard. Alice, Bob and Charlie all have a TBMS on their computer, and are tracing their software. They share altogether their activity's traces, allowing everybody to know what the others are doing.

In this example, we are following Alice and her trace. On her computer, besides her usual applications, Alice has a software component that allows her to visualise, to manipulate and to share her activity's traces. This software allows her to see what she has and the members of her group have done. The precise features and behaviours of this tool are defined by Alice's teacher.

The example scenario is the following: first, Alice logs into her activity environment and discovers that Bob and Charlie are already here. She displays the activity's whiteboard and opens her web browser. She goes on the chat and reads the messages from Bob and Charlie. Bob finds in his web browser an interesting resource and pushes the page to Alice and Charlie. Then Alice writes on the chat that she will collect this resource; Charlie answers OK. So, Alice copies the web resource's URI into her clipboard; after that she pastes the URI onto the shared whiteboard as a new text. She then closes the whiteboard, the chat and the web browser and stops working.

There are several possibilities for presenting a trace to a user: literal text, graph, timeline, etc. These possibilities are discussed in the next section. For the current example, we focus on Alice's trace with a timeline visualisation (see Fig. 8. Alice's trace is represented here with a timeline approach, on which are placed all the obsels of her personal trace. Each interaction mode (common, text chat and whiteboard) is displayed in its own colour for clarity. The relations, such as specified in the trace's model, are displayed as oriented arcs connecting the obsels.

5.4. Requirements for trace visualisation

Presenting a trace to a user is not a trivial task, first because of the temporal nature of a trace (a trace can cover periods ranging from minutes to months), and second because of the complex information it contains. As stated in Section 3.3, we are currently engaged in a process of identifying the various characteristics of modelled traces, in order to define how to sustain individual and group reflexivity with trace sharing and visualisation in realtime.

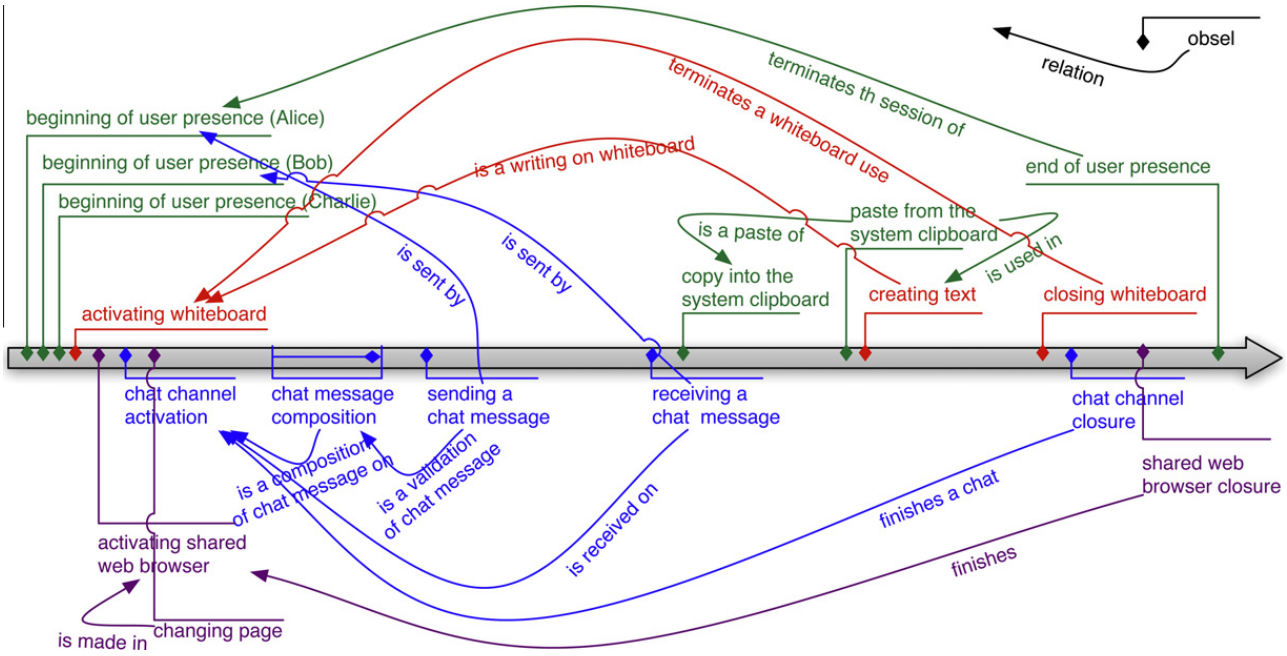

Fig. 8. Alice's activity trace.

For the moment, we have defined general simple principles that we will gradually improve with the results coming from our prototypes. Among them, we state that a software for interactive visualisation of traces must support the following properties:

- selecting the trace(s) to visualise;
- browsing of traces according to various characteristics: time, obsels' types, etc.;
- choosing among several visual renderings for interacting with traces;
- applying transformation on traces: fusion, filtering, etc.;
- selecting obsels for further work (refactoring, export, etc.).

There can be several visual renderings of a trace. One can choose to explore a trace using a natural text rendering approach, while another would prefer a graph with a fisheye for details, or a timeline, etc. The strong decoupling of the trace's visualisation from its content implies to propose to the user some tools for managing the representation methods (at least obsels selection).

As we are working on visualisation of shared traces in synchronous collaborative environments, our main objective is to be able to share and to visualise traces on individual and collective bases. We also need to be able to visualise collections of past traces (such as the ones concerning finished activities) in order to analyse and to share *past* activities.

Concerning the synchronous collaborative aspects on traces visualisation, our needs are therefore the following:

- Sharing and accepting traces: for providing group awareness in a trace-supported synchronous collaborative activity, being able to share traces is critical. It must be doable on an individual or collective base, after selecting or constructing the very traces that are going to be shared.
- Sharing and accepting traces presentation styles: as traces visualisation relays on rendering definitions, those can be shared as well among the activity participants in order for them to have a common representation of activity's traces.
- Sharing and accepting traces transformations: in the same way as for the traces and presentation style sharing, users must be able to share and to accept traces transformations.
- Partaking of group trace: for achieving group awareness, a user must be able to collect information from the actions of the other members of his group. This is done by trace sharing, where each member of the group partakes trace(s) of his interactions with the rest of the group. Every user then has the possibility to collect and process those traces, via transformations, for producing a personal meaningful trace describing the global group activity.

6. Informatics framework for interactive visualisation of modelled traces

In this section we describe the technical elements used for construction an interactive visualisation of a modelled trace. We present here our approach to propose to the users the modelled traces they produce during their synchronous collaborative learning activities. We use the work of [26] for identifying the needed visualisation techniques; our objective being to do this later through a systemic approach.

The goal is to associate a "visualisation style" to a modelled trace, or a set of modelled traces, in order to provide an adapted visualisation to the user. A visualisation style for traces is composed of at least one entity (the visual structure), and its associated computer widget. Each entity of trace is associated to a widget rendering, and all the widgets associated to a given trace provide a complete visualisation style of trace. Naturally, the choice of a particular visualisation style is determined according to the characteristic defined in Section 4.

So a complete visualisation style for modelled traces is composed of: a visual structure, some visual modalities, some visual accents, and some possible interactions. The visualisation styles are stored into files, and can be shared between users; for example, the members of a group engaged in a synchronous collaborative activity can decide to share a visualisation style in order to have a common representation of their activity. We present here the general description of our framework for constructing an interactive visualisation of a modelled trace, and we conclude this work by defining the first criteria for a simultaneous interactive visualisation of modelled traces.

6.0.1. Visual structures

The visual structure is the framework on which is built the rest of a visual and interactive representation of a modelled trace. It defines the structure that provides support for the positioning of trace's elements (the relations and the obsels, with their attributes). Depending on the visual structure, some interactions with the trace's elements will be made easier and the traces will be more understandable. Currently, based on the work of [27], we selected the following visualisation structure for working with the traces produced by our prototypes:

Temporal band: This representation emphasises the temporal aspect of a modelled trace, by placing on a structure the trace's obsels according to their time references and time length.
Wheel: Here, obsels and relations are put on a wheel structure, looping indefinitely as time progress. This representation puts emphasis on parallelisms between obsels while respecting a space constrained allocation.
3D space: This method is mostly used for traces exploration. The traces to visualise are rendered into a three-dimensional space. This allows us to create a graphical representation of traces based on three dimensions that we want to compare (for example: the traces' subjects, the obsels' types, and the time).
Time slices: The approach of this representation is to explode the trace's temporality into time slices for constructing a discrete collection of the trace's obsels. A slice being an atomic time span where obsels are spatially grouped according to their properties. Navigation among the time slices provides the chronology.
Synthetically rendering: The trace's content is processed to produce an abstract representation. The form can be a collection of statistical data, a synthetic summary, a preview of the first and last obsels, etc.

6.0.2. Visual modalities

The modality of visualisation is the approach used for presenting to the user a trace's element. We are using the following two modalities:

Shape: The various elements to represent in different manners are associated to a different shape. The shapes can be simple geometrical forms (circle, square, triangle, ...), or chosen within a collection of icons associated to a meaning (an eye for reading, a pen for writing, an arrow for a movement, ...).
Colour: We assign a different colour to each element's type to represent with this modality.

6.0.3. Visual accents

We call "visual accents" the graduations applied to a modality of visualisation, in order to differentiate the various elements. Currently, we use two types of visual accents:

Alteration accent: For a given modality (colour, form, etc.) used to visualise a type of trace's element, we can apply a slight graduation in order to provide differentiation. This can be made on colours (lighter or darker, shift in colours' palette, ...), on size (shorter, bigger, with italic, ...) and on every possible variation of characteristics.
Texture accent: The textural accent is about applying a change on the rendering of elements for a trace visualisation. While the "shape" and "alteration" change the nature and the size of an element, the "texture" accent provides a way to graduate the intensity of the element aspect. The textural graduation can be achieved by overlay badges, patterns and everything that can help to discriminate a general rendering of a visual element.

6.0.4. Interactions

One of the important aspects of in a learning activity is the metacognitive process that a learner has to go through. Visualising the trace of his own activity supports the learner in this work, specially when the learner has the possibility to manip-

Table 1
List of the possible interactions on the trace elements, without taking into account the specificities imposed by the visualisation structures.

Possible actions on obsel	Possible actions on an trace	Possible actions on relation	Possible actions on trace visualisation
Examine	Delete	Examine	Navigate in the same level
Delete	Apply a transformation	Delete	Choose the obsel to visualise
Modify	Share	Modify	Add an obsel
Compare	Accept a shared	Compare	Add a relation
–	Compare to other trace	Follow	Mark a moment
–	Visualise	Back	Mark an obsel
–	Stop a visualisation	–	Mark a relation
–	Change a type of visualisation	–	Share a visualisation rule (type, current location, etc.)

ulate this reflexive representation in order to adapt it to its own needs. The interactive aspect of the visualisation allows the learner to adapt the visual representation of his activity, so it can match his mental representation. We list in Table 1 the doable interactions on the various elements of the interactive visualisation.

6.1. Simultaneous visualisation of traces

When doing activity analysis or exploration, one can want to look at several modelled traces at the same time, for doing comparison, parallelism, etc. In simultaneous visualisation, the traces have to be synchronised in order to share the same referential. The traces can have their own different models and described distinct human interactions, but they need to be matched again at least one common characteristic. Currently, we identify the following matches:

Temporality: The traces have the same nature of time, so they can be aligned on a shared time scale. This allows us to create a common time-based representation.
Subject: The traces have the same subject, so they can be aligned on a shared user description. This allows us to create a unified representation of a user activity by combining different approaches for describing human interactions.

When two traces to visualise are connected by a transformation (that is, a transformation is applied to the first trace in order to produce the second trace), there is a need to put some emphasis on two aspects:

The first aspect to highlight is, in the second trace, the ensemble of obsels and relations resulting from the transformation. By putting emphasis on those elements, we can help the user to see the connections between the traces and how the transformation process is applied.

The second aspect to take care about is the temporal synchronisation between the traces. As the second trace, resulting from a transformation, is a more abstract view of the situation described by the first trace, they both share the same temporality (reference and time span).

7. Results and evaluation

Our work is currently at the end of a conceptual phase. We did lay down the theoretical approaches and are moving toward a working implementation. For now, our goal is to finish implementing the software components as a first proof of concept. This will demonstrate that it is indeed possible to have a collaborative learning system enhanced with modelled traces, matching our formalisms. After that, we will improve the software in order to be able to run some experiments with them, and see what is the impact of our approach on the cognitive works of a group of people engaged in a collaborative activity.

As a first implementation, we have extended the WeeChat IRC client for trace collecting and implemented a first tool for trace visualisation. Thus, the developed application allows us to get the collected modelled traces, sent by the TBMS under the RDF-XML format, then to extract data from the traces (the types of obsels, the relations, ...) and to visualise them in order to allow the user to interact with an abstract representation of his activity.

Our first informal tests with such an approach, run on ourself, give quite interesting results on the modelling and the architectural side: our first graphical visualisation tool is operational and is currently being extended with more user-related functionalities. We are already able to recognize particular moments (such as a series of technical glitches causing user disconnections and reconnections, or a rapid exchange of questions and answers) of a past chat session. But our visualisation software is actually not good enough for doing a precise reading of an unknown activity.

Our short-time goal is to test the usefulness of our software in real synchronous communication situations. In the ITHACA project, our current objective is to integrate as a plug-in the generic module of trace management in two Technology Enhanced Learning (TEL) platforms (French learning/general school support), and to adjust the modelling and instrumentation for tracing the various collaborative tools, while building dedicated trace visualisation tools adapted to the specific learning tasks of these TELs. A special effort will be devoted to a precise study of the trace transformations that will be needed so as to reach adequate levels of abstraction.

8. Conclusion and future work

This article presents a model of traces dedicated to synchronous collaborative activities. Our research is based on awareness, meta-cognition, self-perception and reflexive learning in order to improve students' skills as an individual or as a group. It consists in proposing tools allowing the user to visualise and analyse their own experiences, the results he produces and the knowledge deduced.

The general principle of our method is to observe, by various means, the user's actions and to represent them in structures called *observed elements*. Thus, we have presented a general framework for using modelled traces (based on observed elements) and the trace base management system. We then have proposed a generic trace model for synchronous collaborative activity based on the notion of interaction mode (roughly related to a communication channel), and we have specialized and illustrated this model for two modes: whiteboard sharing and text chatting.

We are currently working on a software implementation of our models so we can confront them to real situations.

Future works also include the study of automated trace-base learning of users' habits or activity schemes that could be reused by users themselves, through trace-based assistants, or by experts for TEL environments enhancements.

References

[1] D. Clauzel, K. Sehaba, Y. Prié, Modelling and visualising traces for reflexivity in synchronous collaborative systems, in: International Conference on Intelligent Networking and Collaborative Systems (INCoS 2009), IEEE Computer Society, 2009, pp. 16–23. Laurèat du Best Paper Award.
[2] M. Baker, M. Quignard, K. Lund, A. Sèjournè, Computer-supported collaborative learning in the space of debate, in: Proceedings of the International Conference on Computer-Supported Collaborative Learning, Kluwer Academic Publishers, Dordrecht, Netherlands, 2003, pp. 11–20.
[3] A. Soller, F. Linton, B. Goodman, A. Lesgold, Toward intelligent analysis and support of collaborative learning interaction, in: Proceedings of the International Conference on Artificial Intelligence in Education, IOS Press, Netherlands, Amsterdam, 1999, pp. 75–82.
[4] M. Constantino-Gonzáflez, D. Suthers, Coaching collaboration in a computer-mediated learning environment, in: L. Erlbaum (Ed.), Proceedings of the International Conference on Computer-Supported Collaborative Learning, pp. 583–586.
[5] D. Suthers, D. Jones, An architecture for intelligent collaborative educational systems, in: Proceedings of the International Conference on Artificial Intelligence in Education, IOS Press, Amsterdam, Netherlands, 1997, pp. 55–62.
[6] D. Raymond, Y. Yno, C.E. Mauad, V. Baudin-Thomas, T. Gayraud, M. Diaz, K. Kanenishi, K. Matsuura, Bringing mobility to synchronous collaborative activities: recent enhancements of the ''platine" platform, in: WMTE, pp. 59–61.
[7] D. Raymond, K. Kanenishi, K. Matsuura, Y. Yano, V. Baudin, T. Gayraud, M. Diaz, Synchronous CSCL with platine environment, in: Proceedings of the International Conference on Computer-Supported Collaborative Learning 2005 (CSCL2005), pp. 45–47.
[8] J. Lonchamp, Supporting synchronous collaborative learning: a generic, multi-dimensional model, International Journal of Computer-Supported Collaborative Learning 1 (2006) 247–276.
[9] C. Jones, L. Dirckinck-holmfeld, B. Lindstrom, A relational, indirect, meso-level approach to CSCL design in the next decade, IJCSCL 1 (2007) 35–56.
[10] P. Jermann, A. Soller, M. Muehlenbrock, From mirroring to guiding: a review of the state of the art technology for supporting collaborative learning, International Journal of Artificial Intelligence in Education 15 (2005) 261–290.
[11] D. Masciotra, Rèflexivitè, mètacognition et compètence, Vie pèdagogique 134 (2005) 29–31.
[12] S. O'Mahony, F. Ferraro, The emergence of governance in an open source community, Academy of Management Journal 50 (2007) 1079–1106.
[13] D.A. Schön, The Reflective Practitioner: How Professionals Think in Action, Basic Books, 1983.
[14] A.J.G. Cockburn, H. Thimbleby, A reflexive perspective of CSCW, SIGCHI Bulletin 23 (1991) 63–68.
[15] U. Farooq, J.M. Carroll, C.H. Ganoe, Supporting creativity with awareness in distributed collaboration, in: GROUP07: International Conference on Supporting Group Work, ACM Press, 2007, pp. 31–40.
[16] S. Katz, A. Lesgold, G. Eggan, M. Gordin, Modelling the student in sherlock 2, International Journal of Artificial Intelligence in Education 4 (1992) 495–518.
[17] C. Gama, Towards a model of Metacognition Instruction in Interactive Learning Environments, Ph.D. Thesis, University of Sussex, 2003.
[18] C. Plaisant, A. Rose, G. Rubloff, R. Salter, B. Shneiderman, The design of history mechanisms and their use in collaborative educational simulations, in: Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference.
[19] C. Despres, S. George, Supporting learners' activities in a distance learning environment, International Journal of Continuing Engineering Education and Lifelong Learning 11 (2001) 261–272.
[20] D. Clauzel, C. Roda, M. Raglianti, G. Stojanov, et al., Deliverable 1.3: AtGentive Conceptual Framework and Application Scenarios, Technical Report, Consortium AtGentive, AtGentive European Project, 2006.
[21] Y. Lafifi, K. Halimi, A. Ghidbani, N. Salhi, Learners monitoring based on traces in CSCL system, INFOCOMP Journal of Computer Science 8 (2009) 61–72.
[22] D. Cram, D. Jouvin, A. Mille, Visualizing interaction traces to improve reflexivity in synchronous collaborative e-learning activities, in: 6th European Conference on e-Learning, pp. 147–158.
[23] L. Sofiane Settouti, Y. Prié, J.-C. Marty, A. Mille, A trace-based system for technology-enhanced learning systems personalisation, in: The 9th IEEE International Conference on Advanced Learning Technologies.
[24] J. Laflaquiére, L.S. Settouti, Y. Prié, A. Mille, A trace-based system framework for experience management and engineering, in: Second International Workshop on Experience Management and Engineering (EME 2006) in Conjunction with KES2006.
[25] L.S. Settouti, Y. Prié, P.-A. Champin, J.-C. Marty, A. Mille, A Trace-Based Systems Framework: Models, Languages and Semantics, Technical Report, LIRIS, University of Lyon, France, UMR CNRS 5205, Universitè Lyon 1, 2009.
[26] T. Baudel, Visualisations compactes: une approche déclarative pour la visualisation d'information, in: IHM '02: Proceedings of the 14th French-Speaking Conference on Human–Computer Interaction (Conférence Francophone sur l'Interaction Homme–Machine), ACM, New York, NY, USA, 2002, pp. 161–168.
[27] C. Daassi, L. Nigay, M.-C. Fauvet, A taxonomy of temporal data visualization techniques, Revue Information Interaction Intelligence 5 (2006) 41–63.