Modelling and visualising traces for reflexivity in synchronous collaborative systems

Damien Clauzel, Karim Sehaba, Yannick Prié

Modelling and visualising traces for reflexivity in synchronous collaborative systems

Damien Clauzel
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
F-69622, France
Damien.Clauzel@liris.CNRS.fr

Karim Sehaba
Université de Lyon, CNRS
Université Lyon 2, LIRIS, UMR5205
F-69676, France
Karim.Sehaba@liris.CNRS.fr

Yannick Prié
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
F-69622, France
Yannick.Prie@liris.CNRS.fr

***Abstract*—This article addresses issues related to traces modelling and visualisation in synchronous collaborative learning. The objective is to propose models and tools for representing, transforming, sharing and visualizing 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 online 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, which we specialized for whiteboard sharing and text chatting. We modelled an IRC client and developed a first implementation.**

I. Introduction

Synchronous collaborative learning software are increasingly used in various teaching situations: discussion structuring [1], collaborative design [2], [3], construction of knowledge [4], etc. Other environments aim to be generic, such as the Platine [5], [6] or the Omeg+ [7] 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 [8], 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 the representation, treatment and sharing 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 [9]) 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, [10] stipulates that by such means, 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 to 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, aim 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 (eg synchronous films co-annotation). In terms of application, the project focuses on distance learning of French language.

The article is organized as follows: Section II presents and discusses the theoretical foundation of our work. Its consists

[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 eLycée company (http://www.eLycee.com/)

in showing the contribution of traces reflection for learning. Section III presents the general architecture of our system. Section IV details the model we have proposed to represent, to share and to visualise interaction traces in synchronous systems. Section V presents a preliminary application we built. Section VI presents the conclusion and perspectives.

II. 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 [11]). According to [12], 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 [13]) are used to build group awareness, which in turn reinforce synchronous collaboration [14] 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" [15] 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 [16]. Sherlock 2 [15] is an example of a system using this kind of reflexive incentives. Plaisant [17] used a system that graphically represents the actions performed by the learner using boxes and arrows. [18] has developed a system based on traces allowing the tutor to perceive the status of learners' work. [19] has 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 a/ that studies on the reflexive usage of traces of learners' activity in learning environments do not cover the full extend of metacognitive activities that such traces allow; b/ that traces have not been so far used as such for reflexivity in synchronous environments; and c/ that 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 so-called Trace-Based Systems, which we will apply in the context of synchronous collaborative learning tools.

III. General architecture for synchronous collaborative traces

A. Traces in synchronous collaboration

Our team has been working on traces for several years, building applications and studying various usages [20], [21]. As illustrated in Figure 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 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.

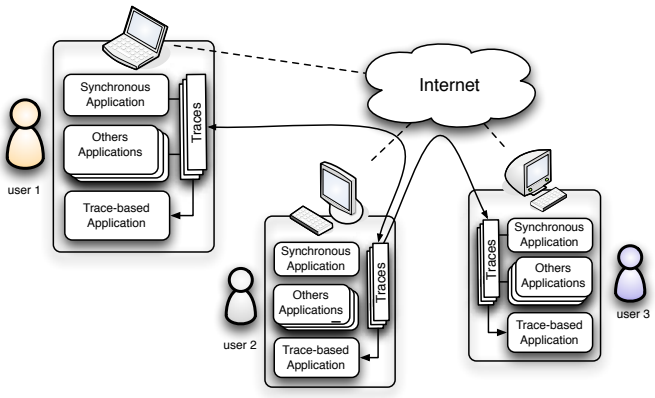

Fig. 1. Traces for individual and collective use.

In this article, we are mostly interested in synchronous collaborative applications, considering that such applications can be enhanced and extended by tracing users and providing them with two kinds of reflexivity: 1/ *online individual reflexivity* is related to personal activity visualisation; 2/ *online group reflexivity* is related to personal activity sharing and to group activity awareness

Of course, 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 (eg. muting the sound can be part of the shared trace); 2/ with activity related to other tools, be they asynchronous (eg. sharing the use of a word processor during the session) or synchronous (eg. extending one's whiteboard trace with part of one's visioconference activity).

For this, we consider as illustrated on Figure 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.

B. 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 [22] we defined the notion of *Trace-Based Management Systems* (TBMS) as systems devoted to the management of *modelled traces*.

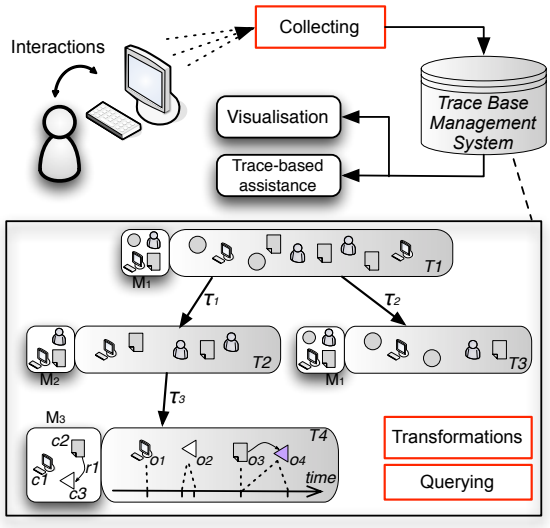

Fig. 2. A Trace-Based Management System framework for using modelled traces. The example trace $T3$ is somewhat detailed: the trace model M_3 contains three obsels types (c_1, c_2, c_3) and one relation type (r_1). T_3 contains four obsels: o_1 and o_3 are related to instants, while o_1 and o_4 are related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

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 is related to the temporal extension of the trace (e.g. it can be related to an instant or a temporal interval). As shown on Figure 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 retrieving 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 (eg. one want 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. Figure 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 T3 according to τ_3. A transformation *by fusion* (see Figure 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 [23]. We are currently developing an open-source TBMS that implements such metamodel.

C. 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 Figure 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 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 $user2$. The subject of T_6 and T_7 is the triple $(user1, user2, user3)$.

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.

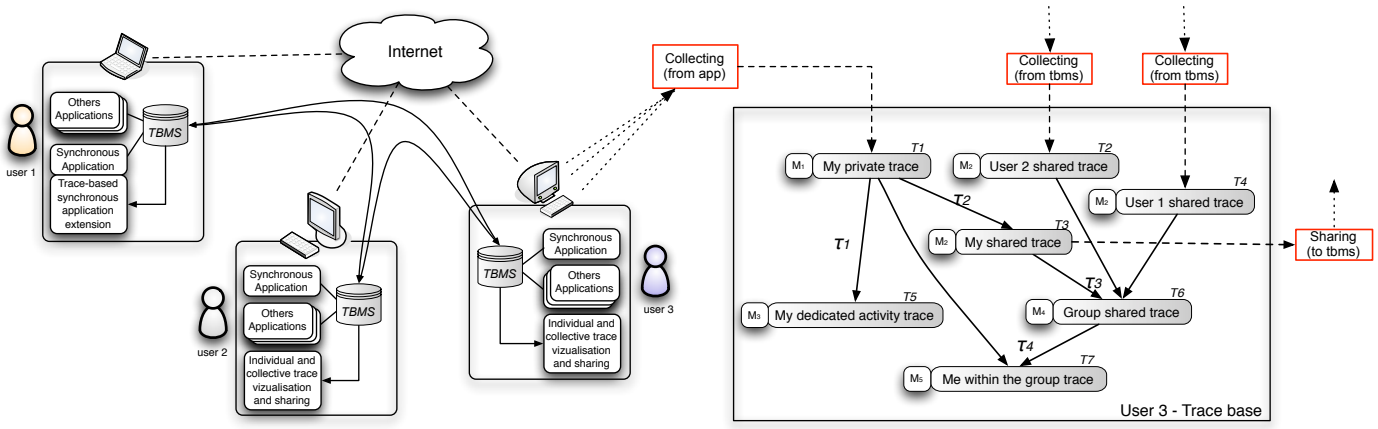

Fig. 3. Left: general architecture for individual and collective trace visualisation and sharing. Note that user 2 and user 3 have a separate trace-based application, while user 1 has a more integrated trace-based extension of the synchronous application. Right: the trace base of user 3. This base contains the primary trace T_1 of $user3$, 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.

IV. Trace models of synchronous collaborative activities

A. 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 in the same time, while depending on each others. Such 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[2]
- collaboratively editing a text: sharing a writing area. Ex.: Gobby[2]
- videoconferencing. Ex.: Skype[2]
- text chatting. Ex.: Skype or ICQ[2]
- co-browsing: several user can engage into a common browsing session, sharing URI, pushing pages, etc.; Ex.: eMédiathèque[2]
- screen sharing: remotely viewing and controlling a distant computer; Ex.: VNC[2]

Of course, an interaction mode can be used in combination with other ones (eg. 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.

[2]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

B. A generic model for synchronous collaborative activity traces

We introduce (Figure 4) 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: th 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 Figure 4) deals with the various categories of interaction modes. The second one (top right in Figure 4) 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 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 as neither the copy or 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

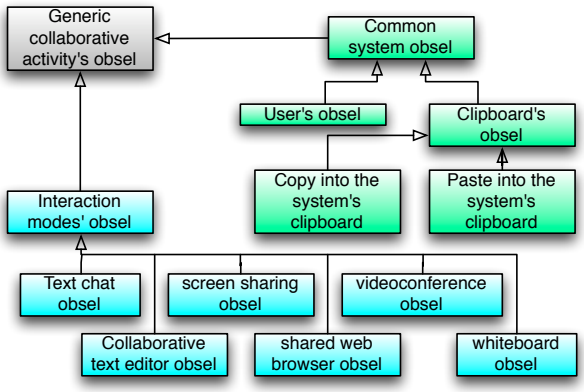

Fig. 4. Structure of the generic trace models for synchronous collaborative activities; specific obsels are not all detailed here

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.

1) Whiteboard sharing model: Figure 5 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, textual content, etc.

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.

2) Text chatting model: Figure 6 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 extension rely 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.

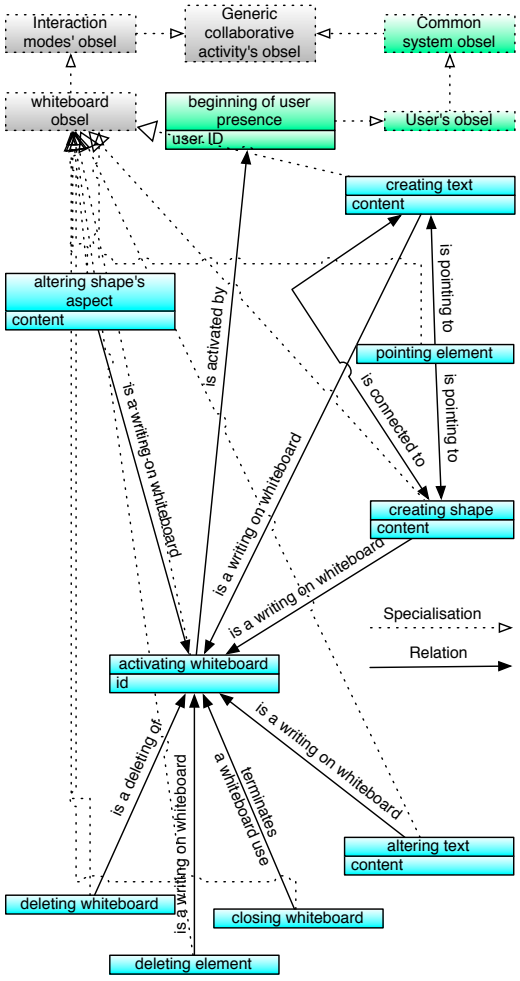

Fig. 5. Generic model of whiteboard user interactions; attributes are not all detailed here

C. A synchronous trace example

As a concrete example, let's imagine the following situation: Alice, Bob and Charlie are three learners engaged into a synchronous collaborative activity. Their work is to search the web for precise informations, and to collect useful resources for a 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, beside her usual applications, Alice has a software component that allow 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 has 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

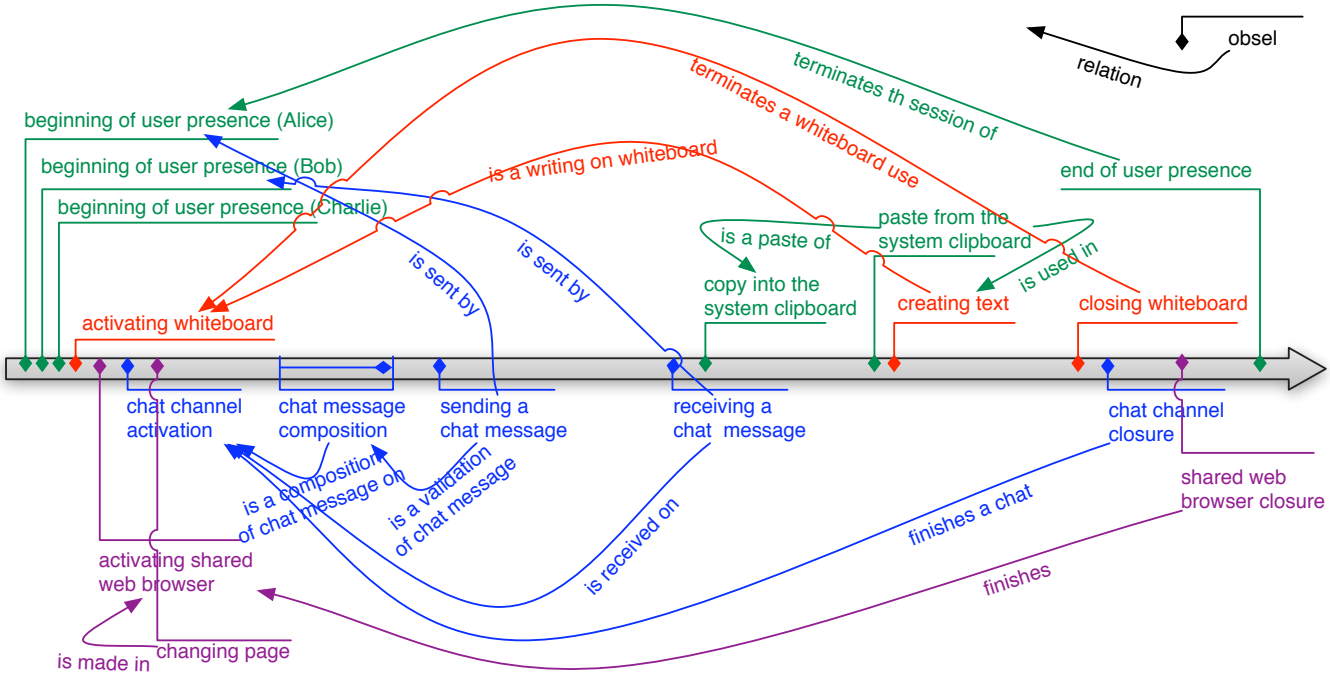

Fig. 7. Alice's activity trace

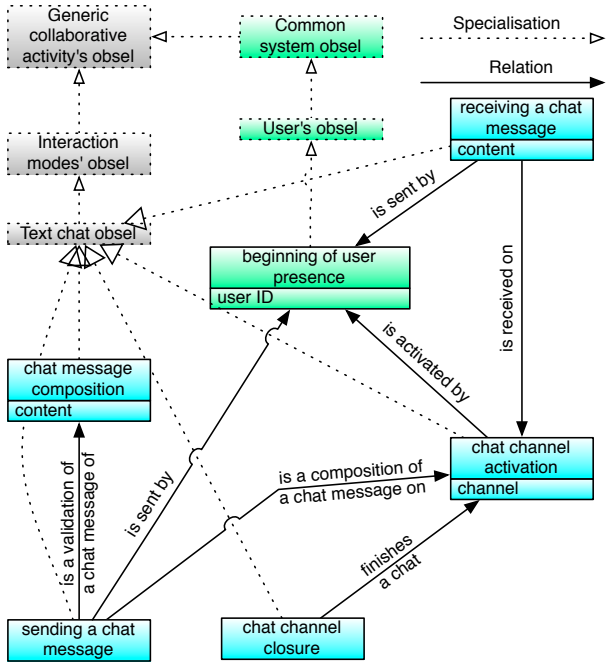

Fig. 6. Generic model of text chat user interactions, attributes are not all detailed here

open her web browser. She goes on the chat and read 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. Those possibilities are discussed in the next section. For the current example, we focus on Alice's trace with a timeline visualisation (see Figure 7). 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.

D. 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 III-C, we are currently engaged into 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.

For the moment, we have defined general simple principles that we will gradually improve with the results coming from our prototypes. Amongst 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 amongst 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 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 followings:

- 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 partake 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.

V. A FIRST APPLICATION

Following those requirements, we are currently developing several software components and a test prototype for experimenting with our trace-based approach for supporting synchronous collaborative activities.

The first one is an IRC client called WeeChat[3], that we did extend for trace collecting[4]. When the user interacts with the IRC client, or when events happen (connection of a user, receiving of a message, request for a file transfers, etc), WeeChat sends the corresponding obsels to the user's TBMS. Figure 8 presents our specialization of the generic model of text chatting interaction mode (see Section IV-B2) for describing the interactions made by a user on a generic IRC client (middle). We expanded this extension for covering the specific features of the WeeChat application, as well as its implementation of IRC (bottom). Figure 8 shows how the different levels of abstraction are specialized, and the obsels that can be directly used with their specification from the generic model. We consider this first specialisation/implementation of our generic model a first validation of this model and of our general approach of modelling synchronous activities with interaction modes.

The second software is a tool for visualising traces, but also interacting with them by applying transformations (trace fusion, filtering, etc.) and sharing them with other people. The current implementation is kind of rough, but it already allows to show the user's own trace together with a trace resulting from the fusion of other users' traces. Future versions of the tools will propose several visualisation modes, transformation and sharing possibilities, etc. the goal being to create a complete generic software for interactive visualisation of traces.

[3]http://www.weechat.org/

[4]http://gforge.liris.cnrs.fr/projects/weechat-traces/

VI. 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 group. It consists in proposing tools allowing the user to visualise and analyse their own experiences, the results it produces and the knowledge deduced.

The general principle of our method is to observe, by various means, the user's actions and represent them in structures called *observed elements*. Thus, we have presented a general framework for using modelled traces (based on observed elements) and trace-base management. 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. As a first implementation, we have extended the WeeChat IRC client for trace collecting and implemented a first tool for trace visualisation.

Our first informal tests with such approach give quite interesting results on the modelling and the architectural side. Our first text-based visualisation tool is operational and will be extended with more user-related functionalities, so as to be able to test its usefulness in real synchronous communication situations. Our current objective in the ITHACA project 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.

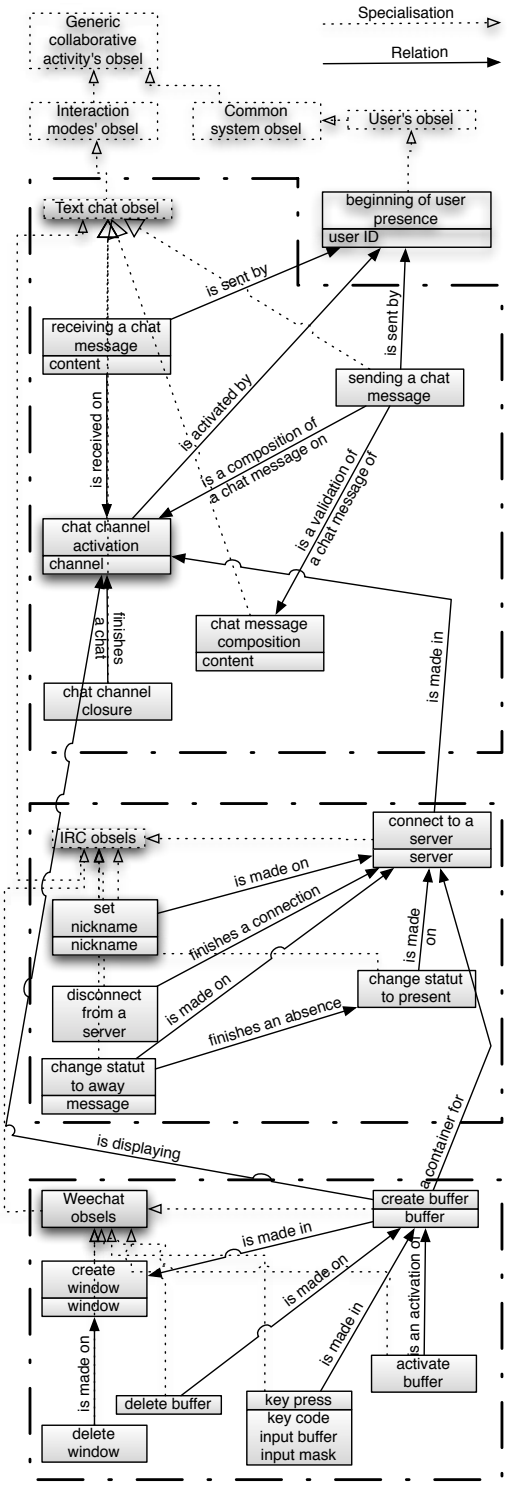

Fig. 8. Exerpt from the WeeChat's specific trace model

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 enhancement.

References

[1] M. Baker, M. Quignard, K. Lund, and A. Séjourné, "Computer-supported collaborative learning in the space of debate," in *Proc. of the Int. Conf. on Computer-Supported Collaborative Learning*. Dordrecht, Netherlands Kluwer Academic Publishers, 2003, pp. 11–20. 1

[2] A. Soller, F. Linton, B. Goodman, and A. Lesgold, "Toward intelligent analysis and support of collaborative learning interaction," in *Proceedings of the Int. Conf. on Artificial Intelligence in Education*. Amsterdam, Netherlands: IOS Press, 1999, pp. 75–82. 1

[3] M. Constantino-González and D. Suthers, "Coaching collaboration in a computer-mediated learning environment," in *Proceedings of the International Conference on Computer-Supported Collaborative Learning*, L. Erlbaum, Ed., 2002, pp. 583–586. [Online]. Available: http://lilt.ics.hawaii.edu/lilt/papers/2002/Constantino-Suthers-CSCL-2002.pdf 1

[4] D. Suthers and D. Jones, "An architecture for intelligent collaborative educational systems," in *Proc. of Int. Conf. On artificial intelligence in education*. Amsterdam, Netherlands: IOS Pres, 1997, pp. 55–62. 1

[5] D. Raymond, Y. Yano, C. E. Mauad, V. Baudin-Thomas, T. Gayraud, M. Diaz, K. Kanenishi, and K. Matsuura, "Bringing mobility to synchronous collaborative activities: Recent enhancements of the "platine" platform," in *WMTE*, 2005, pp. 59–61. 1

[6] D. Raymond, K. Kanenishi, K. Matsuura, Y. Yano, V. Baudin, T. Gayraud, and M. Diaz, "Synchronous cscl with platine environment," in *Proc. of the International Conference on Computer-Supported Collaborative Learning 2005 (CSCL2005)*, May 2005, pp. 45–47. 1

[7] J. Lonchamp, "Supporting synchronous collaborative learning: A generic, multi-dimensional model," *Int. Journal of Computer-Supported Collaborative Learning*, vol. 1, no. 2, pp. 247–276, June 2006. [Online]. Available: http://dx.doi.org/10.1007/s11412-006-8996-7 1

[8] C. Jones, L. Dirckinck-holmfeld, and B. Lindstrom, "A relational, indirect, meso-level approach to CSCL design in the next decade," *IJCSCL*, vol. 1, no. 1, pp. 35–56, 2007. 1

[9] P. Jermann, A. Soller, and 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*, vol. 15, no. 4, pp. 261–290, 2005. 1

[10] D. Masciotra, "Réflexivité, métacognition et compétence," *Vie pédagogique*, vol. 134, pp. 29–31, feb 2005. 1

[11] S. O'Mahony and F. Ferraro, "The emergence of governance in an open source community," *Academy of Management Journal*, vol. 50, pp. 1079–1106, Oct. 2007. [Online]. Available: http://www.business.ualberta.ca/tcc/documents/TII_3_OMahoney_Ferraro_final.pdf 2

[12] D. A. Schön, *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, June 1983. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0465068782 2

[13] A. J. G. Cockburn and H. Thimbleby, "A reflexive perspective of cscw," *SIGCHI Bull.*, vol. 23, no. 3, pp. 63–68, 1991. 2

[14] U. Farooq, J. M. Carroll, and C. H. Ganoe, "Supporting creativity with awareness in distributed collaboration," in *GROUP07: International Conference on Supporting Group Work*. ACM Press, 2007, pp. 31–40. 2

[15] S. Katz, A. Lesgold, G. Eggan, and M. Gordin, "Modelling the student in sherlock 2," *International Journal of Artificial Intelligence in Education*, vol. 4, no. 3, pp. 495–518, 1992. 2

[16] C. Gama, "Towards a model of metacognition instruction in interactive learning environments," Ph.D. dissertation, University of Sussex, 2003. 2

[17] C. Plaisant, A. Rose, G. Rubloff, R. Salter, and 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*, 1999. [Online]. Available: http://hcil.cs.umd.edu/trs/99-11/99-11.pdf 2

[18] C. Despres and S. George, "Supporting learners' activities in a distance learning environment," *International Journal of Continuing Engineering Education and Lifelong Learning*, vol. 11, pp. 261–272, 2001. 2

[19] D. Clauzel, C. Roda, M. Raglianti, G. Stojanov, and al, "Deliverable 1.3: AtGentive Conceptual Framework and Application Scenarios," Consortium AtGentive, Tech. Rep., Sep. 2006, atGentive European project. [Online]. Available: http://www.AtGentive.com 2

[20] D. Cram, D. Jouvin, and A. Mille, "Visualizing interaction traces to improve reflexivity in synchronous collaborative e-learning activities,"

in *6th European Conference on e-Learning*, Oct. 2007, pp. 147–158. [Online]. Available: http://liris.cnrs.fr/publis/?id=3000 2

[21] L. Sofiane Settouti, Y. Prié, J.-C. Marty, and A. Mille, "A trace-based system for technology-enhanced learning systems personalisation," in *The 9th IEEE Int. Conf. on Advanced Learning Technologies*, Jul. 2009. [Online]. Available: http://liris.cnrs.fr/publis/?id=3974 2

[22] J. Laflaquière, L. S. Settouti, Y. Prié, and 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*, Oct. 2006. [Online]. Available: http://liris.cnrs.fr/publis/?id=2473 3

[23] L. S. Settouti, Y. Prié, P.-A. Champin, J.-C. Marty, and A. Mille, "A Trace-Based Systems Framework : Models, Languages and Semantics," LIRIS , University of Lyon, France, UMR CNRS 5205, Université Lyon 1, technical report, 2009. [Online]. Available: http://hal.inria.fr/inria-00363260/en/ 3