Automatic Analysis Assistant for Studies of Computer-Supported Human Interactions *

Christophe Courtin, Stéphane Talbot

▶ **To cite this version:**

Christophe Courtin, Stéphane Talbot. Automatic Analysis Assistant for Studies of Computer-Supported Human Interactions *. Transactions on Computational Science, 2009, 5794, pp.572 - 583. hal-02884231

HAL Id: hal-02884231
https://hal.science/hal-02884231

Submitted on 29 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Automatic Analysis Assistant for Studies of Computer-Supported Human Interactions*

Christophe Courtin and Stéphane Talbot

Équipe Systèmes Communicants, Université de Savoie, Campus de Savoie Technolac,
73376 Le Bourget du Lac cedex
{Christophe.Courtin,Stephane.Talbot}@univ-savoie.fr

Abstract. This paper presents a system architecture to bridge the gap between the users computing activity in collaborative platforms and the analysis of this activity which is carried out by researchers in human and social sciences. This research work tends to highlight the capacity of a computer-supported observation station, based on a theoretical model called TBS (Trace-Based System), to assist researchers automatically in their activity of analysis using a high abstraction level. We present the modules of a prototype of an observation station called CARTE (Collection, activity Analysis and Regulation based on Traces Enriched) which enable the interoperability between the collaborative platforms, where the users produce raw traces and the analysis environments, where the researchers study traces of a very high abstraction level.

Keywords: trace, automatic analysis, observation, learning activity.

1 Introduction

The introduction of the technology in the collaborative learning activities has aroused the interest of the researchers in human and social sciences, who study the various forms of concerned cognitive and social activities [1]. There is a complexity in these studies such that it is necessary to assist them, as is possible to do with technology. Actually, the analysis of human interaction with a computer system remains a human activity with a high abstraction level, where the technology can assist the researcher by helping him/her to achieve his/her tasks.

In parallel to these research works in human and social sciences, there are others in computer science relative to the design of computer-supported collaborative learning systems, which we label observation based on activity traces. The relationship of the concerned actors (i.e. learners and teachers) to the software system is where the interest of the study of the traces for learning lies. Indeed, we have shown that we could exploit the activity traces, by means of a regulation model [2], in order to modify the system processing so that it tends towards the user's own model of this system, that we call the use model [3].

* This work is part of the cluster ISLE project, supported by the "Region Rhone-Alpes" in France.

U. Cress, V. Dimitrova, and M. Specht (Eds.): EC-TEL 2009, LNCS 5794, pp. 572–583, 2009.
© Springer-Verlag Berlin Heidelberg 2009

2 Study Context

The study we present in this article aims to define the significance of the results of research work in computer science in the field of the observation of the collaborative learning activities, for the automatic assistance of human and social science researchers' activities, on the understanding of the analysis processes of the socio-cognitive interactions. It is necessary now that we define precisely the context of the study presented above.

The socio-cognitive study of computer-supported human interactions, is based on the software traces (object of study) (e.g. row data from a log file, data from an instrumentation of the software tools [4]) and on the computerized traces (e.g. video/audio transcription). The research practices in human and social sciences are, among others, the creation and the sharing by the research communities of corpora of traces useful for the understanding of the activities performed. Those of computer science are collection from various sources (e.g. log file, video), representation in different abstraction levels and help in the analysis of multi-modal and multi-user interaction traces. The traces processing is carried out in an independent system, called "observation system", which is designed from a theoretical model of observation which we shall not re-describe in this article due to the lack of space [5], [6]. In general, we define a trace as being a sequence of temporally-situated "observeds". The concept of "observed" represents a datum relative to the observation activity. For the interpretation of the trace in a given context, we associate to it a semantics by means of a use model and we use the term MTrace (trace + model). In the context of the socio-cognitive study of the computer-assisted human interactions, the researchers use three types of trace:

1. Interaction traces stemming from software tools which are used for the collaborative activities;
2. Calculated (or created) traces which correspond to a computer translation of traces stemming from non computer sources (e.g. transcriptions of video);
3. Enriched traces, i.e. stemming from transformations of interaction traces or calculated or already enriched by means of use models of the software tools from which they arise.

The final research question we are going to treat in this article is: how can an independent observation station provide an automatic assistance to researchers in human and social sciences, who carry out analyses of multi-modal and multi-user traces of interaction using a specific computing environment? In this article, we will focus on the computing aspects of the question, such as the interoperability of the systems where the activities take place (e.g. production, communication) and the analysis system that will be described in the following paragraphs. The other computing questions which derive from our study concern, among others, the capacity of an observation station to produce automatic analyses which make sense for the human observers.

The observation of the multi-modal and multi-user interactions in a context of computer-mediated collaborative learning is an intrinsically multidisciplinary activity. We humbly measure the limits of computer science when complex activities are carried out, compared to the greater capacity of analysis of the human being. The results

that we expect from our research works are the design of computer models which aim to stretch these limits.

In this article, we first set out the context of the trace analysis that we split into three areas (or families of software tools): the collaborative platforms, the observation stations and the analysis environments. The following chapter deals with trace structuration which is used in each family to make specific analyses. Then, we will present our trace-centered approach to allow interoperability between the various analysis systems. From this, we present the bases of the design of an automatic analysis assistant for researchers in human and social sciences. This study leads us to a conclusion and a perspective view of this research work in a multi-disciplinary project, which starts in the coming months.

3 Trace Analysis

We have underlined the difficulty in carrying out automatic analysis of traces of multi-modal and multi-user interactions by use of a computing environment, which makes sense for researchers in human and social sciences. Conversely, the latter are confronted with a complexity of the analysis such that it is necessary for them to have assistance, in particular in the tasks involving very many parameters or time consuming tasks. We identify three families of software tools to carry out these analyses (see Fig. 1). The first family corresponds to the collaborative platforms which contain the tools of production, communication, cooperation and regulation according to the four dimensions of the clover model of [7] augmented by the regulation by [8]. These software tools intrinsically produce traces which, according to the situation, can be directly exploited for observations generally in recorded mode. We will study the case of the DREW © (Dialogical Reasoning Educational Web) system [1], which we will present hereafter, and which allows the exploitation internally of its own traces. The second family corresponds to the analysis environments which include software tools dedicated to assisting the observers in the management, the synchronization, the visualization and the analysis of activity traces. In these environements, most of the aforesaid actions are carried out manually on the traces. In our study context, namely the analysis of the human interactions with a computer system, we will study the TATIANA © (Trace Analysis Tool for Interaction ANAlysts) system [9], also presented hereafter, which fits the aforesaid description. The third family, which is at the heart of our study, concerns the observation stations, i.e. environments which fit the specification of the theoretical observation model which we call a TBS (Trace-Based System) [5], [6], on which we have contributed to its definition with other actors in the Technology Enhanced Learning and Teaching (TELT) field. Today, we have a prototype called CARTE (Collection, activity Analysis and Regulation based on Traces Enriched) [2] which fits this specification and of which we will present the modules which answer the problem presented at the beginning of this article.

3.1 Collaborative Platforms

Collaborative platforms represent the digital workspace environments which place at users' disposal software tools to carry out collectively the tasks of the field of application. We will limit ourselves here to the tools dedicated to the collaborative learning.

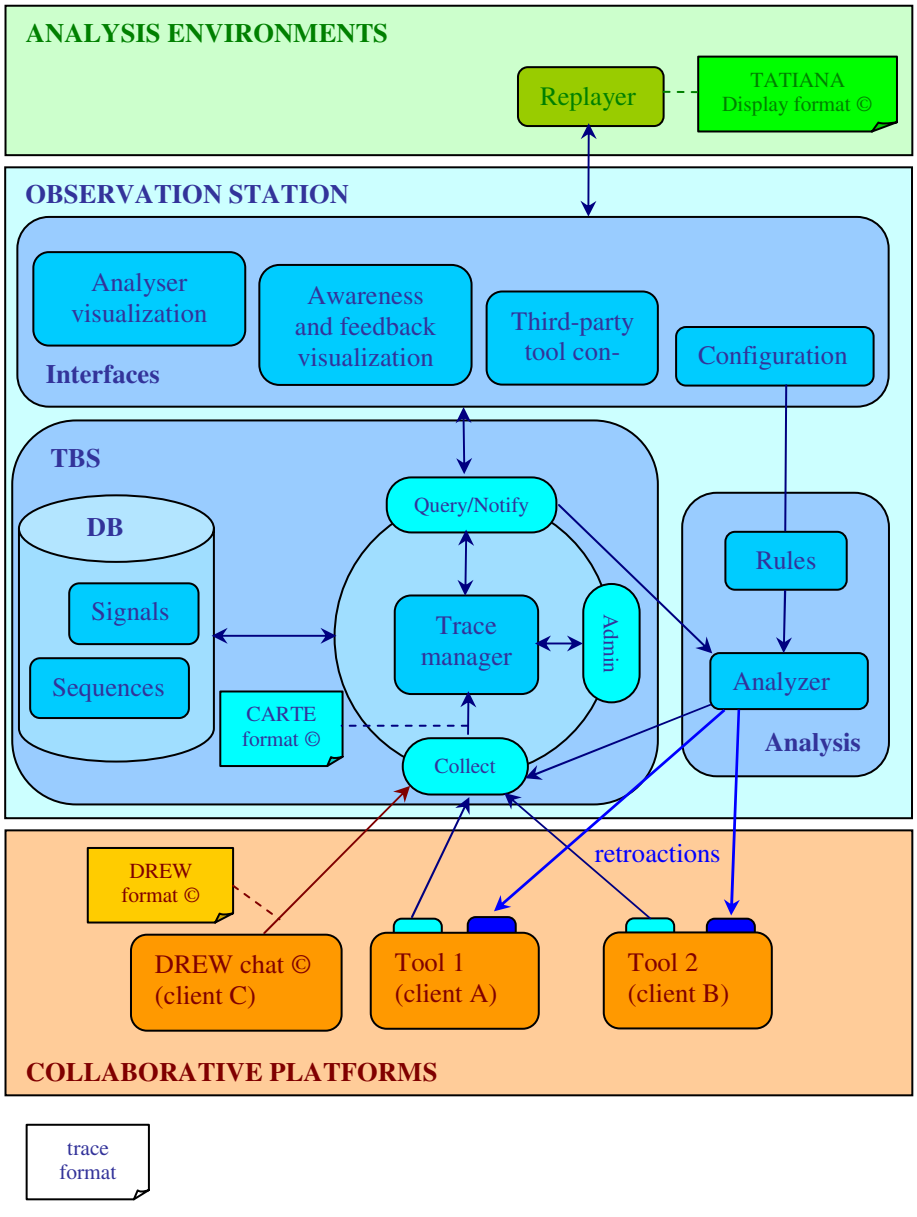

Fig. 1. The observation station

In the context of our research, we carried out experiments during a practical class in an English course (foreign language) at the university [4]. The students were to define remotely in pairs English vocabulary stemming from a text. The system was composed of a production tool called “Jibiki” (a specific collaborative text editor to create multilingual dictionaries), created for a research project in linguistics and

whose working depends on trace exploitation [10], of a communication tool called "CoffeeRoom" (a chat room in which communication spaces are represented by tables), of a group structuration tool and of an activity monitoring tool (awareness). The activity traces were gathered by a collection API (Application Programming Interface) of an observation station, by means of an instrumentation of the aforesaid software tools (see Tools 1 and 2 in the Fig. 1). The semantic level of the traces which were analysed automatically with the observation station thanks to predefined tool use models, fit the analysis objectives of the concerned observers, i.e. the users of the system (learners and teachers). The chosen architecture, namely the externalisation of the trace processing towards an observation station, allows us to plan a posteriori other more detailed exploitations of the trace base thus constituted. We will present below the Query / Notify API, of the CARTE observation station, for the exploitation of this base with other analysis objectives.

The analysis of human interactions with a computer system consists frequently in replaying the activities which have been carried out with software tools or described from other sources (e.g. video transcription). The DREW © system has been designed with this idea in mind. DREW is a supported-collaborative learning platform [1] which allows the researcher to manage computer-mediated interaction traces stemming from the collaborative tools' use (e.g. DREW's chat or DREW's shared text editor). DREW can read its own traces (sequences of events with XML format) and reproduce the activity, by means of an internal replayer, within the software tools (e.g. DREW's shared text editor) which were initially used [11]. Furthermore, the DREW replayer can be synchronized and controlled by an external replayer, such as for example that of the TATIANA © analysis tool. Thus, the researchers can add indicators in the DREW replayer, for example a color for each participant to differentiate the contributions. This coupling between an analysis environment and the DREW collaborative platform allows manual creation of epistemic indicators (high level traces) from behavioral observeds (lower level traces). The DREW system meets in a satisfactory way the expectations of researchers in human and social sciences, for analyses based on the principle of replaying in specific tools.

3.2 Analysis Environments

Analysis environments supply software tools to assist researchers in the management, synchronization, visualization and analysis of the traces in order to create new ones which make sense for them. Below we present briefly the TATIANA environment, which assists researchers in the analysis of traces stemming from contexts that are complex because multimedia, multi-modal (synchronization of several trace sources) and multi-user [9]. As explained above, the researchers can use TATIANA to create "replayables" ("display format") and analyses (use of language for defining filters/rules) [12]. A "replayable" can be displayed in a tabular form (an array with several columns such as Time / User / Message / Tool) or a graphical one (using style sheets). The filters, based on XQuery scripts [13], are used to convert the data into an XML format (with the description of their structure), then into the format that can be understood by TATIANA ("display format" or pivot format or sequence of events). The TATIANA replayer reads a file of traces and interprets each event as an additional client would have done during the observation. The traces are replayed in the

analysis sense of the term, but also in the action sense of the term, i.e. in the tools of the collaborative environment (e.g. DREW's chat tool). TATIANA has therefore been designed to interact with replayers of collaborative platforms. A script to import is used to select one or several sources in an input file (e.g. a discussion in a chat, a video transcription). It is worth noting that tasks relative to analysis (e.g. categorization, annotation, argumentation), realized manually, may be complicated and repetitive. The automation of some of these tasks devolved to the researchers, would require interoperability with an observation station to carry out transformations of traces thanks to predefined use models of the tools. We will present below the API which allows this coupling with the CARTE observation station.

3.3 Observation Stations

The third family concerns the observation stations, i.e. environments dedicated to the collection of traces stemming from various sources, their structuring according to a generic format, their transformation according to the predefined analysis objectives and their re-injection towards analysis tools or towards the tools from which they arise (retroaction principle of CARTE [2]) to make possible the regulation of the activity. In our study context, an observation station is a system, based on a theoretical observation model, which provides observers (e.g. researchers) with observation services in order to facilitate their tasks of analysis. According to the CARTE system's architecture, the analysis and the tools to exploit the traces are different from the tools used to carry out the activities of the application domain, i.e. all the functionalities relative to the collection, structuration and analysis activities are integrated into the observation station. In our study context, the researchers (observers) can use the CARTE observation station to keep track of interactions whose treatment must aim to facilitate their tasks in their analysis system. CARTE allows the real time collection of traces in order to make analyses either in parallel with different use models, or that are successive (thus cumulative) in order to increase the abstraction levels of the resulting traces. Thus, certain repetitive elementary analysis tasks which are carried out with TATIANA can be automated with CARTE, as well as certain more complex tasks stemming from the researchers' experience. Furthermore, we have observed that the TATIANA replayer allowed the manual integration of awareness data to increase the understanding of the traces. By means of the retroaction mechanism, the CARTE system can automatically supply external awareness tools, which could be those of TATIANA. Therefore, the analyst him/herself has to first specify, in the observation station, with a rule-based use model of the tools, the awareness information s/he wishes to receive during the analysis with TATIANA.

We have noted that the trace analysis could be carried out at various abstraction levels: from the primary traces (e.g. raw data from a log file, data from instrumentation [4]) in the collaborative platforms, from temporarily situated behavioral "observeds" in the observation stations and epistemic indicators (elaborated by the researchers themselves) in the analysis environments. To determine the levels of analysis according to [14], sociologists consider three criteria: the temporal context engaged in the observation, the volume and the nature of the units of observation. Then there is the question of the selection of the analysis material and the sources from which it comes. Certain types of analysis, which consist in climbing the scale of

abstraction, imply interaction traces stemming from collaborative platforms, traces that are calculated (or created) by means of use models in an observation station and traces that are enriched by the techniques of categorization, of argumentation and of annotation in the analysis environments. In the following part, we shall present the traces formats of the various levels which will be exploited in the transformations in order to lead to relevant analyses.

4 Traces Structuration

With the observation theoretical model, traces are placed at the heart of the trace-based system. In this approach, we add an abstraction level between actions to be carried out and the various applications. Thus, if a communication tool (e.g. a structured chat room) is replaced by another one which fits the same specification, the collected raw traces (that we call "signals") will then be impacted, but not the enriched traces (that we call "sequences") because of their high abstraction level. Therefore, the trace format used in the CARTE system has to be as generic as possible. We consider that trace information is divided into two parts: the first one which is common to all the traces (e.g. source, date, etc.) and the second one which is specific to the various tools used (e.g. a sentence in a collaborative text editor, a table name in a structured chat room). The former is represented by the metadata and the latter by a list of parameters (see Fig. 2). Both of them are described below.

With the trace manager of CARTE, we are able to manage two kinds of activity traces:

- signals which correspond to time pinpoints and elementary elements (e.g. a user action, a state modification of the system, and so on);
- sequences which split into a chronological succession of signals or sub-sequences. Obviously, a sequence also has a duration and normally should make sense to understand what has happened with the tools.
- Signals contain :
- a source (the person or the tool which has generated the signal);
- a tool (the tool or the instance tool in which the event has taken place);
- a date (the timestamp which says when the event happened);
- an event id (the list of possible events will of course depend on the tools we use: connection, disconnection, message emission, etc.);
- a textual description and
- a list of parameters (which contains the variable parts of signals). In this list we should find everything that is needed for us to understand what has happened. For example who and what are involved in the event. Obviously, this part will also change with events and tools.

The sequences are more complex than signals. They are composed of signals or sub-sequences and, as signals, have parameters. In our implementation, each sequence is stored with:

- a start date (timestamp – the beginning of the sequence);
- an end date (timestamp – the end of the sequence);

- a type id, which characterizes the kind of sequence;
- a source (the person or the tool which has recognized the sequence – the source is generally the analyzer);
- a textual description and
- a list of parameters (which gives all significant details of the memorized episode).

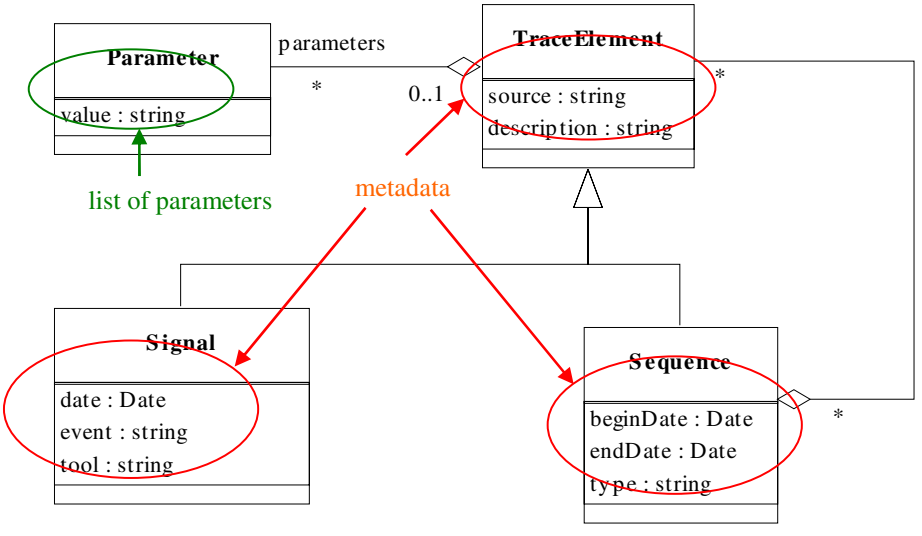

Fig. 2. UML model for traces

We consider that whatever the trace format we obtain, it could be possible to transform it into the CARTE trace format (which is based on the XML standard) thanks to its flexibility (i.e. parameters). From a technical point of view, there are specific scripts to import and export traces (e.g. from the XML format of DREW) according to the objectives of the interoperability (i.e. collection, exploitation, interrogation, notification) between the observation station and the external tools. We present below the various corresponding API (Application Programming Interfaces) of the CARTE observation station.

5 Observation Station API

5.1 Collection API

The collection API (Application Programming Interface) is aimed at simplifying the process of collection of the traces. This API provides application developers with tools which intend to help them during the task of instrumentation of the tools in the implementation phase (or adapt them afterwards). Furthermore, we postulate that if there are tools which provide the possibility of transforming and adapting application traces from other formats (log files, xml files, and so on), we expect that more and more applications will offer the possibility of generating traces. As a consequence, we

should quite easily be able to create tools to exploit these traces and to provide the users and groups with added values.

In our prototype, CARTE, a service (the collector) is used to give access to the collection API. Two methods **sendSignal** and **sendSequence** have been defined and allow tools to send signals or sequences to the TBS (Trace-Based System). Because we are using a J2EE server for our system, the API are accessible either as a WEB service, or as java RMI or Corba objets (depending on how the J2EE server is configured).

In order to simplify the use of our system, we are also providing different versions of these methods. In the simplest ones, just one parameter should be given: the signal or the sequence to record. In a second version of the methods, the signals or the sequences can be described using their inner parts: time_stamp, source, tool, event, description, parameters, and so on. Last but not least, a special version (that actually the analyzer utilizes) includes a supplementary parameter which is used to associate a session ID (identifier) with the traces gathered and generated during the same analysis. This ID can afterwards be reused, for example, to visualize the results of a given analysis or to replay it.

5.2 Trace Exploitation API

In the first version of our system, the analyzer was tightly coupled with the trace manager. This had some drawbacks – for example, it was impossible to run different analyses simultaneously or to use different analyzers.

Using the same approach that we had employed for the collection, we were led to define the trace exploitation API in order to normalize the use and the interrogation of traces. Actually, we have identified two kinds of usages:

- interrogation needs – we should be able to request the trace base for interesting traces;
- notification needs – some tools (the on-line analyzers, for example) should be notified when new traces (signals or sequences) are produced and recorded in the trace base.

So we have defined API for each of these two usages: interrogation API and notification API.

5.3 Interrogation API

This version of the API is used to make requests on the trace base. It's a “pull-oriented” API: the tools will use it to ask the TBS to search for particular trace elements (signals or sequences) and send them back.

As for the collection API, the interrogation one is still very simple: actually, the service provided defines only two methods **searchSignal** and **searchSequence**. The names are quite self-explanatory: the first one is used to search signals and the second one sequences.

Each method exists in different variants. In the first one, a “pattern”, i.e. a partially instantiated signal (or sequence), is used for the search: the type and the parameters of the signals (the sequences) can be defined or left free. The TBS then returns the list of

the signals (or sequences) which match the pattern. For example we could search all the signals generated by a chat room in which the first parameter is equal to "Arthur".

In the second variant, the patterns are not applied directly to the traces but are defined and interpreted using some "use model" (which, of course, should have been previously defined). The interpretation of the traces is then used to find out which ones should be returned. For example, we could search all the sequences where one parameter can be interpreted (using a particular "use model") as a "user name" and has "Arthur" as its value. In this last form, the pattern format is very similar to the one used within the analyzer.

5.4 Notification API

This version of the API is "push-oriented": the tools (analyzers, visualization tools, …) act as trace consumers and are registered in the TBS, which acts as a trace producer and notifies the tools when new signals or sequences arrive.

The notification service of the observation station defines two methods, **addSignalListener** and **addSequenceListener**, which are used respectively to register signals and sequences consumers. On the analysis environment side, the tools have to implement either the **SignalListener** or the **SequenceListener** interfaces which define respectively the methods **sendSignal** and **sendSequence** that the TBS uses to send the signals and sequences it receives.

As for the interrogation API, the listeners (the traces consumers) have the ability to specify, using a "pattern", which kind of traces they want to be notified of. As we have seen before, this pattern should be used to match the traces directly or by means of a particular "use model".

Our main goal with these API is to obtain more modularity in our system, in particular to untie the "trace manager" from the analyzer. The advantages of the approach are numerous. For example, if we need to we could replace the actual analyzer by another one (and use a statistical analyzer or a case-based one if they are more suitable) or even run different analyzers side-by-side.

These API also greatly simplify the connection of external third-party tools like activity trackers or visualizers and improve the extendability of the system: more sophisticated filters can easily be implemented using the simple ones described above and be used to gradually improve the complexity of the filters and requests the tools could use to interact with the TBS.

6 Utilization of the API

As a proof concept we have used the API with the tools we are employing with our system, CARTE, for the experiments we have carried out at the university [4]. So all the tools used during these experiments, the group structuration tool, the dictionary elaboration tool (the "JIBIKI" [10]) and the chat tool (the "CoffeeRoom"), have been instrumented in order to send traces to the TBS (using the collection API).

For example, in the chat facility, each time a user performs an action, the chat server generates a specific signal: for each kind of action a participant can do, we have defined a specific event (and a family of signals).

- On connection/disconnection the chat tool sends a signal, which has this particular event id (connection or disconnection) and with one parameter (the user name of the participant who has connected/disconnected).
- When a participant sends a new message, the server sends a "talk" signal with three parameters (the user name of the sender, the communication space where the message has been sent and the content of the message).
- And when a chat communication space is created or deleted (in our chat tool, the "CoffeeRoom", users can create or delete new communication spaces, represented by tables), two specific signals are also created and sent to the TBS. These particular signals have two parameters: a user name and a name identifying the corresponding communication space created or deleted.

The other two tools used for our experiments have been similarly instrumented. So, during an experiment, we are able to record all the significant actions performed by any of the participants (i.e. the students and the teacher).

We have defined the Query/Notify API more recently, so actually we are only using them in order to separate the analyzer from the TBS. The first steps in this direction were made a few months ago and the separation is now effective. The first consequence is that the same analyzer can now be used to perform on-line or off-line analyses (to perform the off-line analyses we just have to re-play the selected traces). We are now changing the analyzer in order to be able to run different analyses in parallel. Another outcome is that normally, any analyzer which implements our API should be able to be plugged in the TBS.

Next we will similarly separate the internal visualization tools from the TBS. And to go one step further, we also plan, in a new project, to use the API to connect our system with DREW [1] and TATIANA [9]. After that, DREW will be seen as a new collaborative platform able to use CARTE to store its traces. In the same way we hope to be able to use TATIANA on these traces as a new analysis tool; not as an automated one like CARTE's current analyzer, but as a learning expert-oriented one.

New experiments will then be carried out, which we hope will demonstrate firstly that our approach is realistic and secondly that if we could combine automatic analyses with human ones, we would obtain better results.

7 Conclusion and Perspectives

The work presented in this paper is a preliminary stage for a multi-disciplinary project. We have set out the observation area as being the association of three families of tools (1. The collaborative platforms; 2. The analysis environments; 3. The observation stations), providing the researchers (observers) with traces which allow various abstraction levels. We have described a specific module of the CARTE observation station which is able to interoperate with raw traces from the DREW collaborative platform and enriched traces from the analysis environment TATIANA, in order to assist researchers in human and social sciences automatically in their analysis tasks. In the near future, we plan to place at researchers' disposal modules, included in the CARTE observation station, implementing functionalities to facilitate the automation of certain tasks of the analysis process in any analysis environment. The final objective of such research work is the creation and the sharing of corpora of traces by the communities of researchers, who try to understand the activities performed.

References

1. Corbel, A., Girardot, J.-J., Lund, K.: A Method for Capitalizing upon and Synthesizing Analyses of Human Interactions. In: van Diggelen, W., Scarano, V. (eds.) Workshop proceedings Exploring the potentials of networked-computing support for face-to-face collaborative learning, 1st European Conference on Technology Enhanced Learning (EC-TEL 2006), Crete, Greece, pp. 38–47 (2006)
2. Courtin, C.: CARTE: an Observation Station to Regulate Activity in a Learning Context. In: IADIS International Conference, Cognition and Exploratory Learning in Digital Age (CELDA 2008), Freiburg, Germany, pp. 191–197 (2008)
3. Courtin, C., Talbot, S.: Trace Analysis in Instrumented Collaborative Learning Environments. In: 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006), Kerkrade, The Netherlands, pp. 1036–1038 (2006)
4. Talbot, S., Courtin, C.: Trace Analysis in Instrumented Learning Groupware: an experiment in a practical class at the university. In: Seventh IASTED International Conference on Web-based Education (WBE 2008), Innsbruck, Austria, pp. 418–422 (2008)
5. Settouti, L.: Systèmes à base de trace pour l'apprentissage humain. In: Actes du colloque de l'Information et de la Communication dans l'Enseignement Supérieur et l'Entreprise (TICE 2006), Toulouse, France (2006)
6. Settouti, L., Prié, Y., Marty, J.-C., Mille, A.: A Trace-Based System for Technology-Enhanced Learning Systems Personalisation. In: 9th IEEE International Conference on Advanced Learning Technologies (ICALT 2009), Riga, Latvia (2009)
7. Ellis, C., Wainer, J.: A Conceptual Model of Groupware. In: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW 1994), Chapel Hill, North Carolina, United States, pp. 79–88 (1994)
8. Ferraris, C., Martel, C.: Regulation in groupware: the Example of a Collaborative Drawing Tool for Young Children. In: 6th International Workshop on Groupware (CRIWG 2000), Madeira, Portugal, pp. 119–127 (2000)
9. Dyke, G., Lund, K., Girardot, J.-J.: TATIANA: un logiciel pour l'analyse des interactions humaines médiatisées par ordinateur, de la spécification à l'implémentation. Research report, G2I-EMSE 2008-400-005 (2008)
10. Mangeot, M., Chalvin, A.: Dictionary Building with the Jibiki Platform: the GDEF case. In: 5th Language Resources and Evaluation Conference (LREC 2006), Genoa, Italy, pp. 1666–1669 (2006)
11. Corbel, A., Girardot, J.-J., Jaillon, P.: Drew: A Dialogical Reasoning Tool. In: 1st International Conference on Information and Comunication Technologies in Education (ICTE 2002), Badajoz, Spain (2002)
12. Dyke, G., Girardot, J.-J., Lund, K., Corbel, A.: Analysing Face to Face Computer-mediated Interactions. In: 12th Biennial International Conference (EARLI 2007), Budapest, Hungary (2007)
13. W3C. XQuery 1.0, An XML, `http://www.w3.org/TR/xquery/`
14. Drulhe, M.: Orientations épistémiques et niveaux d'analyse en sociologie. SociologieS, Théories et recherches (2008), `http://sociologies.revues.org/index2123.html`