Trace Analysis in Instrumented Collaborative Learning Environments†

Christophe Courtin
Laboratory Systèmes Communicants – University of Savoie
73376 Le Bourget-du-Lac Cedex, France
{Christophe.Courtin,Stephane.Talbot}@univ-savoie.fr

Stéphane Talbot

Abstract

Observation activity, on instrumented collective learning situations, enables participants to appropriate themselves corresponding systems in their own practice. In this paper, we focus on the analysis stage of the observation process, but we underline the correlations between the different stages. We present an analysis model and a draft of trace description language. Finally, we explain in an example how an observer (e.g. teacher) can exploit collected information to understand particular episodes.

Keywords: observation, trace, analysis model, collaborative learning.

1. Introduction

Nowadays, computer science plays a preponderant role in learning activities because of its technological possibilities. In practice, we nevertheless notice that learning activities’ specificities are difficult to implement. In particular, when it is a matter of collaborative learning in which the exchanges are essentially of a human nature, and where automatic processing of a learning process is difficult to achieve. It is essential to provide the human user with means adapted to her/his practice, enabling her/him to appropriate the software device when s/he carries out activities.

We attempt to solve the aforementioned problems by favouring the observation of instrumented collective learning situations (ICLS), by means, among others, of instrumentation of the software tools used [4]. Anne Lejeune [9] underlines the necessity of making automatic observation and regulation of human activities in ICLS.

2. Objectives

In this paper, we focus on one of the numerous observation activity objectives, namely the analysis of the collected traces. Nevertheless, it is worth noting that the different observation stages are not totally independent [7][6]. In particular, activity trace interpretation may be facilitated when the elements collected are structured. We thus propose a model enabling activity trace analysis which is based on a representation of the traces, and which therefore promotes the exploitation of the information collected.

3. Trace collection and structuring

As we have shown in [4], it seems to be necessary to provide instrumented tools in order to identify more precisely the operations carried out within them. The information collected structuration (cf. Figure 1) must facilitate activity interpretation at a higher abstraction level by means of a specific use model (which will be described in this paper). An abstraction level corresponds to an observer’s specific point of view (e.g. pedagogical, communicational, etc.).

Figure 1. UML model for traces

† This work is part of the ACTEURS project (2005-2007), supported by the French Ministry of National Education, Higher Education and Research.
We share the idea that it is necessary to know the observation objectives in order to define in advance the observed elements adapted to the expected interpretation. In Carron’s research work [1], the observed elements are defined in an observation pre-experiment stage. Smith and Korel [11] underline the huge number of elements which would be collected in the case of a bottom-up approach, i.e. collecting all the elements and sorting afterwards.

4. Trace analysis

Capturing events in order to reuse them later is not a new idea, as is shown in Jaczynski’s research work on Web navigation [8]. However, most of the systems supporting instrumented collaborative learning situations come up against interpretation difficulties for the end-users of collected traces. Some works have striven to solve this problem, and have led to the establishment of use models [2]. Desmarais’ research work [5] about statistics production on collected traces, through software systems dedicated to human learning, establishes that such information is insufficient to describe precisely the learning process. Other research work on data mining technics, associated with web access logs, have shown that it is possible to assess in fine the learning process [13].

During our different experiments in the laboratory [4][6][7][1], we have observed that it is essential to highlight several interpretation levels, in order to identify particular periods in learning situations. What we have previously noticed may be explained by the natural diversity of human observers, and of their language level. We express these different activity interpretations by applying use models on the software tools used.

4.1. Use model

As far as our study is concerned, a use model is based on previously structured information associated with specific semantics for the optional parameters, that we call “templates”. We distinguish two types of templates: signals and sequences (cf. Figure 1). Signals may represent low abstraction level information, corresponding to elementary actions on tools (e.g. save a page). They may also represent high abstraction level information, deduced from a recognized learning activity (e.g. to succeed in writing a definition). Sequences express composite actions, which make sense for an observer when they are put together (e.g. edit_page + save_page = edition). If we extrapolate this notion, we deduce that sequences can be composed of signals and/or other previously created sequences.

This recursive dichotomy of templates allows one to adapt the information collected to the observation objectives. For instance, a teacher may wish to measure the degree of collaboration between a pair of students. This objective has a high abstraction level, and therefore may be expressed by a sequence with explicit teacher-oriented meta-description. In addition, in case the teacher wishes to know about collaboration conditions in more detail, s/he can exploit the information embedded in the sequence. It is worth noting that this change in granularity level may lead to a change in abstraction level, because the analysis system explores as far as the recognition of elementary operations (e.g. sending a message, page saving, etc.).

The setting-up of a use model consists in defining activity patterns with a learning activity description language. The choice of the trace description language depends on its adequation with the observation objectives of the learning activity under consideration, i.e. its description capacity. Among potential description languages, Carron [1] proposes to use regular expressions in order to transform generally low abstraction level observed elements (signals generated by tools) into higher abstraction level ones (sequences and signals generated by the analysis system) capable of being interpreted by a human observer. For the purposes of our research work, we apply rules on templates composed of logical operators, which will be described in the “Implantation” part. The rules have a double advantage: trace filtering and enrichment of information. Indeed, the recognition of patterns reinforces the idea of Smith and Korel [11] about filtering trace noise, according to a given observation objective. In addition, activity description in a higher abstraction level may convey and create signals not produced directly by the system, because of their semantic level being too high (e.g. to succeed in writing a definition).

Furthermore, it is worth noting the double independence of the analysis with regard to the observation objectives on the one hand, and to the software tools used on the other. Indeed, the analysis relies on a use model which expresses a particular point of view, the observer’s, with a specific language level. The substitution of a use model allows analysis adaptation to different observation fields (e.g. sociological expertise, pedagogical evaluation, HCI design). As far as analysis independence relative to the software tools used is concerned, the optional parameters use tool semantics (cf. Figure 1). Thus, the substitution of one communication tool for another one consists in redefining this low level semantics, without compromising the high level of observation.

In practice, the process of carrying out the analysis may be divided into three stages: configuration of
templates (to define the signals’ optional parameters),
rule processing (to recognize signals and sequences),
and presentation of the resulting traces for
interpretation. The analysis consists in defining a use
model of the different tools used, according to a
particular observation objective. If several observations
are made, there will be several use models providing
results which are either independent of each other (e.g.
HCI observation implies GUI event oriented analysis
when pedagogical evaluation implies task-oriented
analysis), or connected as for the example presented in
the “Implantation” part (cf. Figure 2): the use model
for pedagogical evaluation is based on results provided
by the use model for interaction observation.

5. Implantation

To date, we have implemented a first version of our
system to try and validate our collection model of
intra-tool and inter-tool use traces [4]. The objective of
our current research work is to apply our own use
model. A prototype is currently in progress. The use
model is supposed to allow collected trace analysis,
based on predicted uses of concerned software tools.
As is mentioned above, we use rules based on logical
operators (AND, OR, NOT), on temporal relations
(THEN), and on priority relations (parentheses).

We are going to describe the previously presented
element of collaborative activity to illustrate our use
model. More precisely, we attempt to recognize a
collaborative work episode with our model. The
episode we are going to describe concerns students’
co-writing of a topic provided by a teacher, and
integrates communication between the participants.

According to our collection model [4], simplified
with Figure 1, we present signals and sequences related
to the writing activity. In the template of “edition”
sequence below, the user “u1” has written on “page1”
in the period of time [?t0, ?t1].

[analyser, ?u0, ?t1, ?description1,

Figure 2. Sequence recognition and signal generation
The corresponding rule to recognize such an activity is described as follows:

- **Rule condition part:**
  - \((a[\ast, \text{tool}, \ast, \ast, \text{?event}, \text{?u}, \text{?p}]) \text{ THEN } b[\ast, \text{?tool}, \ast, \ast, \text{?event}, \text{?u}, \text{?p}])\)
  - with \(a.\text{?tool}=b.\text{?tool} \land a.\text{?t}<b.\text{?t} \land a.\text{?event}=\text{edit-page} \land b.\text{?event}=\text{save-page} \land a.\text{?u}=b.\text{?u} \land a.\text{?p}=b.\text{?p}\)

- **Rule execution part:** “editation” sequence and “definition done” signal are generated.

In the template of “co-edition” sequence below, the users “u1” and “u2” has written on the same “page,” in the period of time \([?t_0, ?t_3]\).

- \([\text{analyser}, ?t_0, ?t_3, \text{?description}, \text{?display}, \ast, \text{?event}, \text{?u}, \text{?p}]\)
- \([\text{analyser}, ?t_0, ?t_3, \text{?description}, \text{?display}, \ast, \text{?event}, \text{?u}, \text{?p}]\)
- \([\text{wiki, wiki}, ?t_0, \text{?description}, \text{edit-page}, \text{?u}, \text{?page}]\)
- \([\text{wiki, wiki}, ?t_0, \text{?description}, \text{save-page}, \text{?u}, \text{?page}]\)
- \([?\text{su}_1, ?\text{sp}_1, ?\text{type}_1]\)
- \([\text{analyser}, ?t_2, ?t_3, \text{?description}, \text{?display}, \ast, \text{?event}, \text{?u}, \text{?p}]\)
- \([\text{wiki, wiki}, ?t_2, \text{?description}, \text{edit-page}, \text{?u}, \text{?page}]\)
- \([\text{wiki, wiki}, ?t_2, \text{?description}, \text{save-page}, \text{?u}, \text{?page}]\)
- \([?\text{su}_2, ?\text{sp}_2, ?\text{type}_2]\)
- \([?\text{cu}, ?\text{cp}, ?\text{type}]\)

where \(?t_0<?t_1<?t_2<?t_3\); \(?\text{cu}=\text{S}\) with \(\text{S}=U(?\text{su}_1, ?\text{su}_2)\)
and \(\text{card}(S)>1\); \(?\text{cp}=?\text{sp}_1 \land ?\text{sp}=?\text{sp}_2\);
\(?\text{type}_1=?\text{type}_2=\text{edition} \land \text{type}=?\text{co-edition}\)

In other words, co-edition =

- \([\text{analyser}, ?t_0, ?t_3, \text{?description}, \text{edition}, ?, \text{?cu}, ?\text{cp}, \text{?type}]\)

The corresponding rule to recognize such an activity is described as follows:

- **Rule condition part:**
  - \(c[\ast, ?t_0, ?t_1, \ast, \ast, \ast, ?, \text{?su}, \text{?sp}, \text{?type}]\)
  - AND
  - \(d[\ast, ?t_0, ?t_1, \ast, \ast, \ast, ?, \text{?su}, \text{?sp}, \text{?type}]\)
  - with \(c.\text{?t}<d.\text{?t}_1 \land c.\text{?su}=d.\text{?su} \land c.\text{?sp}=d.\text{?sp} \land c.\text{?type}=d.\text{?type}=\text{edition}\)

- **Rule execution part:** “co-edition” sequence then, “co-edition-begin” and “co-edition-end” signals are generated.

We observe from this episode that the trace description language allows one to recognize different granularity levels, without necessarily changing abstraction levels. Generally speaking, the change in granularity may be used to improve the understanding of a particular situation. For instance, the teacher recognizes the following episode with a high abstraction level (in normal, signals generated by tools, and in bold, signals generated by the analysis system):

- **communication** (\((?u_1, ?u_2), \text{table}_1\));
- **definition_done** (\((?u_1, ?u_2), \text{table}_1\));
- **teacher_checking** (\((?u_1, ?u_2), \text{page}_1\));
- **definition_failed** (\((?u_1, ?u_2), \text{page}_1\));
- **co-definition_done** (\((?u_1, ?u_2), \text{page}_1\));
- **teacher_checking** (\((?u_1, ?u_2), \text{page}_1\));
- **definition_succeed** (\((?u_1, ?u_2), \text{page}_1\));

Studying this episode allows to identify on one hand communication between students “u1” and “u2”, and on the other hand “u1”’s writing activity, followed by the pair’s co-writing activity. Finally, the teacher intervenes twice to check the work done, the validation result being visible each time.

This first analysis may satisfy an observation objective such as “has there been collaboration within the pair?”. Nevertheless, this analysis level informs neither about the reasons for the failure of the first writing activity, nor about the means to recover this problem which has allowed, in fine, the production of a satisfying result. This is because of inadequate granularity which, consequently, masks some explanatory information. A more in-depth analysis may be achieved by using a finer granularity. For instance, the teacher can explore the communication activity, and therefore the nature of the changes (parameter ?msg_content) within the pair, in order to understand how the students have solved the problem. This change in granularity, associated with a filter on the event “send_msg”, may lead to the following result (in bold, signals generated by the analysis system, and with an arrow, signals of lower granularity level):

- **communication-begin** (\((?u_1, ?u_2), \text{table}_1\));
- **send_msg** (\((?u_1, ?msg_content), \text{table}_1\))

The analysis of the exchanges between the two students, before and after the failure, is likely to inform the teacher about the pair’s strategy. It is worth noting that an analysis between different abstraction levels (as for our example) implies an adaptation of the visualization interface, in order to present information in an understandable form for the teacher.
6. Conclusion and perspectives

The work presented in this paper is part of an overall project about the development of an observation station. The analysis stage completes work on the collection stage [4], and highlights the necessity of structuring the information collected. We have presented a use model, and a draft of trace description language with several abstraction levels. Finally, we have explained how an observer may look for more precise information by changing granularity levels.

Our future work consists in reinforcing user considerations in the use model by means of regulation of the latter. Indeed, regulation allows to modify dynamically the use model according to user activity [9][12]. In other words, the dynamic use model enables the software analyser to add new uses to the predicted ones [2].

7. References


