Monitoring Student Activities in Computer-integrated Classrooms to Provide Teacher Feedback

Jens Hardings, Nelson Baloian, H. Ulrich Hoppe José A. Pino

September 25, 2008

1 Introduction

Using computer technology or not to support a learning process, the achievement of the learning goals depends on many factors [1]. According to many studies, the systematic monitoring of the students work is a key success factor. Only
when teachers are able to determine the learning process and the learning stand
of the students they are also able to teach them properly [2]. This is also
coherent with Cotton, who says “The body of educational research literature
which has come to be known as the effective schooling research identifies the
practice of monitoring student learning as an essential component of high-quality
education” [3].

Taking in consideration these facts, many authors have developed systems
that monitor students work mainly for the case of distance learning supported
by a Learning Management System (LMS). LMS are well suitable for tracking
students activity since most of them provide at least a low level
logging which registers all the actions of the students. These log files can after-
wards be automatically analyzed to extract higher level information regarding
students progress [4].

Researchers that have studied assessment and evaluation techniques say that
standardized testing has received considerably more research attention than
have classroom testing and other classroom-level assessment methods [5]. How-
ever, seat work has been mentioned to be an important activity of the lecture.

In fact, it has been reported that comparing the behaviour of effective teachers (i.e., those whose students achieve highly or higher than would be expected given background variable) with that of less effective teachers revealed the importance of monitoring the class during seatwork periods [3].

However, monitoring students during seatwork may involve teachers moving around the classroom, being aware of how well or poorly students are progressing with their assignments, and working with students one-to-one as needed. These activities might be quite time consuming and sometimes difficult to perform with bigger number of students. Moreover, Cotton says that the most effective teachers have systematic procedures for supervising and encouraging students while they work and initiate more interactions with students during seatwork periods, rather than waiting for students to ask for help. Even in class with not many students, these actions take a lot of time and effort to do and only teachers having with a vast experience or applying a smart methodology would be able to do this.
In the past, many authors have developed computer-based learning material in the form of “electronic worksheets” supporting the seatwork of students. They have been often called “Active Documents” and have been developed in order to provide the students with a richer environment to interact with, as well as allowing the collaborative work by making use of available networks [6, 7, 8, 9]. In most cases, an XML Document Object Model has been used as a way to manipulate this kind of documents and to store them in permanent storage devices. Having students working on these electronic documents, where they have to complete their task by modifying the document content opens up the possibility of an automatic and therefore systematic analysis of their work. For example, it can be used to find out how are the students advancing in the completion of the tasks proposed in the active document, if they are filling it with the right answers, etc. In this paper, we present system which allows a teacher to flexibly monitor the work of students working individually or collaboratively in electronic documents in their computers.
2 Literature

As already discussed in the previous chapter, monitoring students’ work is an important issue in pedagogy. Accordingly, many educational researchers have studied this practice and its effect in the learning process. First and foremost, it has been said that assessment is a valuable practice to promote learning [10]. The relationships between classroom work evaluation and student outcomes involving learning strategies, motivation, and achievement have been studied in [11] concluding that classroom evaluation has a powerful direct and indirect impact. This impact can be either positive or negative. Therefore, assessment has to be carefully planned.

In [12] the authors classify classroom assessments into three categories, each serving a different purpose. Summative assessments summarize what students have learned at the conclusion of an instructional segment. Diagnostic assessments typically precede instruction. Teachers use them to check students’ prior knowledge and skill levels, identify student misconceptions, profile learners’ in-
terests, and reveal learning-style preferences. It also provides information to assist teacher planning and guide differentiated instruction.

Formative assessments occur concurrently with instruction. These ongoing assessments provide specific feedback to teachers and students for the purpose of guiding teaching to improve learning. Formative assessments include both formal and informal methods, such as non-graded quizzes, oral questioning, teacher observations, draft work, think-alouds, student-constructed concept maps, learning logs, and portfolio reviews. This third type of assessment is the one supported by the work presented in this paper.

A revision of the literature has shown that most computer-based systems have focused on students work monitoring of e-learning. In [13], two objectives were discussed for e-learning assessment – the recognition of problems among students in e-learning (e.g., they are not reading the materials, they are spending too much time in discussions), as well as the evaluation of e-learning to improve the quality of courses (recognising those course materials that are not being used). WebCT was used in this research as the CMS. To facilitate the analysis
of results, a new, extra tool was proposed, which enables the visualisation of analysis results.

Another study [14] performed evaluation of e-learning to assess and control the study process. Log file analysis (time and number of accesses) was used. Log files used in the study were not Web server log files, although the authors spoke to the possibility of using Web server log files in research about student activity.

Web log analysis was used in [15] with the purpose of assessing the effectiveness of course usage, defined as course usage intensiveness, manner and usefulness. The author of the study asserts that three sources can be combined to obtain an overall insight into students’ online learning process – Web logs, student demographics, and survey results. Statistical methods were used for the data analysis.

Many researchers have been proposing the use of data mining for the evaluation of e-learning. Log files are also used as data sources in [16, 17]. A data warehouse is usually used for management information analysis of various kinds,
as well as for Web data analysis.

Computer technology has already been used systematically inside the classroom for some time [18, 19, 20] its potential to assess in-classroom students’ work has not been explored as much as the case of e-learning has been. Our present work is a contribution in that direction.

3 System Description and Architecture

The framework being used as a basis for this research is CiCv2, which stands for Computer-integrated Classroom. In this framework, both teacher and students use a computer in the face to face sessions, allowing them to interact on different levels. The teacher has the ability to present teaching material, distribute assignments, exchange individual or group messages and share documents through an archive. A flexible modeling tool, Freestyler [21], displayed in the top left of figure 1, implements a series of visual languages for modeling in the domain of a particular subject. This students have the possibility to use the modeling tool
in collaboration by way of shared workspaces that are provided in the CiCv2 framework.

Teacher and students have access to a central repository which also has the role of an authentication service. Once authenticated, they can access the documents on the repository, share them with the class, and send documents to particular users or groups. In the face to face session, the teacher establishes a session and the students will sign in. From that point on, the application on the teacher’s computer acts as a server for that session, having some control on the activities of the students by allowing or disallowing interactions among students and by assigning activities. Within a session, documents can also be exchanged and shared locally, without access to the central repository. By sharing a document we mean working simultaneously on the same document, propagating the changes among the participants as they occur.

The framework consists, as we have seen, of several components. The components and their role are specified as follows.
Figure 1: CiCv2 Framework
3.1 Central Repository

The central repository gives users the ability to authenticate and access files as well as interact with the other users’ applications. Additionally, it serves as a directory for currently active sessions in order for the students to connect to the session managed by their teacher.

The central repository keeps a log of its interactions with student and teacher applications.

3.2 Modeling tool

The Freestyler modeling tool has been integrated into the application available to both teacher and students. It can be used as a whiteboard application to present material and solutions to proposed problems, as well as modeling in several learning domains. The learning domains are enabled by the usage of plug-ins that define specific visual modeling languages [9] which support one or more domains.

The content in the modeling tool is divided into pages that can be added,
erased and copied. Additionally, each page can be shared among several or even all members inside a session, in which case a change of the content on behalf of one member gets immediately updated on all of the participants sharing that page.

The modeling tool saves its documents in the context of the application it is embedded in, and from there the document can be distributed to other applications or to the central repository. The teacher may distribute documents for this application as a work guide for the session, and can automatically evaluate student’s advance and correctness of solutions, among other information.

This modeling tool can be used as a basis for developing problem based learning scenarios. It can be used to model in a wide array of domains, from mind mapping to specific simulations, and new capabilities can be added in the form of alternative visual languages by creating a new plug-in that defines them. As the document has a rich structure that represents visual languages and graph-based models, it is suitable to be used to add semantics and meaning to the data, allowing the teacher to monitor the students’ progress and difficulties.
3.3 Student Application

The application available to students allows them to interact with the repository by sending and receiving files. Inside a session, the student application can also interact with the teacher and other students by exchanging messages in a chat, exchanging documents to be opened independently or sharing documents, in which case they can modify a common document in real-time. While it is possible to manage any type of document by having an associated application, in this work we are considering only documents used by the embedded modeling tool, Freestyler.

The documents can be saved locally, on the central repository, or shared among participants in the classroom session. The application also maintains a local log detailing the actions of the user.

3.4 Teacher application

The teacher application has the same functionalities as the student application. Additionally, the teacher can control some of the activities the students can
perform, and send assignments. It is also in this application where we added a querying system so the teacher can monitor the session as needed.

3.5 Shared pages manager

The shared pages, consisting of individual pages in the modeling tool, need a server inside the session to share the events generated by changes on behalf of the users. This server is embedded into the teacher application, and is started as soon as the teacher creates the session.

When a page is shared on behalf of any user inside a session, that shared page becomes available for any other user inside the session to join. The shared pages manager maintains a copy of the whole page, as well as a log of individual changes as they occur.

4 Querying system

As already mentioned, the teacher application in CiCv2 was enhanced with the capability of performing queries, intended to allow the teacher to monitor the
classroom situation. The querying capability is implemented as an additional plug-in inside the modeling tool, which avoids to clutter the interface with external modules, and uses a simple visual language to manage and use the queries inside the classroom session.

Using the querying capability the teacher can access the meaningful information sources available in the CiCv2 scenario. It can extract information from files and log information from the central repository, the logs of the shared pages manager, the locally stored documents, and from the documents and logs of all student applications participating in the session.

The queries are generally prepared in advance, and can be specific to the activities in that particular session, or general purpose queries that can be useful in any session. Either way, the queries are readily available as a way to minimize the teacher’s involvement in technical details during the classroom session. However, it is also possible for the teacher to adjust specific parameters on queries to achieve the desired results.

When the teacher has selected a query, executing it involves pressing a but-
ton, and the result will either appear beside the query, or generate some changes
to the currently active document, such as adding new pages with results. It is
also possible to program queries to be executed periodically or at a specific time,
having access to updated results without any more interaction.

Session-specific queries require teacher involvement in the preparation. As
with the session material, where the teacher prepares specific problems to be
solved by the students, preparing the respective queries is an additional step
that has to be taken into account, making the activity inside the session more
productive.

4.1 Query implementation

The queries developed for this work are available to the teacher through a vi-

sual representation for each element, grouped together as an additional visual
language. The visual language presents basic queries that are used as building
blocks. These basic queries can be combined to create new queries.

The implementation of the query functionality is separated from the visual-
ization of such query. Following the Model-View-Controller pattern, the query functionality implementation is used as the model part of the pattern. For each model, a specific graphical representation is created, which allows to manipulate any potential variable stored by the query that should be user-defined. The output of one query is connected as input to another query by visually connecting edges in a graph.

As an example, it is possible for example to obtain the difference between two documents by using one FileQuery for obtaining each file and connecting them to a DiffQuery. In order to hide the complexity, these three queries can be encapsulated inside a ComplexQuery, so the end user sees only one simple query that performs what is expected. As a result, the teacher can see one complete query that hides all complexity and just delivers a result when needed.

The queries have access to a context, that allows access to external sources of information. When a query needs to execute some part remotely, it sends that part to the remote location. There, the context is set accordingly, so that the remotely executing part has access to the local resources and sends the results
4.2 Internal data structure and Document Object Model

The modeling tool maintains an internal data structure with the model, and has the ability to translate that model, creating an XML Document Object Model (DOM) that can be serialized as a file. This translation works both ways, allowing to restore the internal data structure from the file.

In a normal setting, a query can make use of documents that have been stored, e.g. accessing the central repository, and documents that are currently being edited. A document that is being edited may have a corresponding version stored either locally or on the repository. However, that version might be out of date and will not contain any changes made since it was saved. As a result, a query will need to access either both models, or translate one to the other.

We have chosen to use the DOM as the basis to execute the queries. This decision is based on the efficiency of translating from the internal model to DOM, existence of standards like XQuery [22] and the maturity of the seri-
alization/deserialization process for XML DOM, since the information has to be serialized at the remote locations to be transferred to the teacher application.

4.3 Basic Queries

The basic queries are the basic building blocks that allow arbitrarily complex queries to be implemented by combination. Some queries do not have any input, and only generate output, such as a constant query that always returns a fixed result, and a “current document query” that always returns the current document as defined by the context where it is being executed. Other queries are terminal queries and do not provide an output, such as a “Save Query”, that simply saves its input into a file whose name it also fetches from its input, or the “Object Creation Query”, that creates a visual object that gets added as a new element in the modeling tool. The other queries have inputs that are processed to create a single output, such as the XQuery, that executes a particular XQuery on its input, generating a single output.
4.4 Execution of queries

Several queries can be combined by connecting the output of one to be the
input of the next, using the pipe metaphor common in process streams in oper-
ating systems. The execution starts with the “last query”, i.e. the query that
whose output is not fed into another, and this query activates all of its input
queries. This can be seen in figure /reffig:ControlDataFlow, where the control
flow advances from left to right. At any point, the control flow can continue
its execution remotely, sending the query and the remaining queries to a re-
move location. At the remote location, the query will be rebuilt, given access to
the local context and continue the control flow. This continues with the input
queries, activating all of their respective input queries, until the leaf queries are
reached, which do not have any external input.

Then, the information flows in the opposite direction (right to left in fig-
ure /reffig:ControlDataFlow, delivering the results that are processed in every
step, reaching the “last query” again. If the query was executed remotely, the
result is sent over the network in opposite direction, finishing the remote execution and continuing with the local processing. The “last query” receives the final result and it generally is one of the so called “terminal queries”, which present or save the result in a way that is useful for the teacher.

5 Experiments

In order to test the monitoring activities stated above, we have developed a series of experiments which are described below. Each experiment is situated in a specific context or scenario, where students are asked to perform some activities and the teacher uses the querying system to monitor the session. The
group of students for all experiments is the same, consisting of 16 undergraduate and 2 graduate students, between 22 and 26 years old. All of the students were taking the course on distributed computing, and none had previous knowledge of either the CiCv2 environment nor the particular problems presented. As we intended to use the tool in as real an environment as possible, we used it in learning sessions with the same constraints a teacher would encounter normally. This included very short introductions of both the CiCv2 environment and each modelling plug-in, no more than 10 minutes in each experiment, which was enough to get the experiments started.

5.1 Tracking student pace and identification of solutions

This scenario was set up to identify the status of students while they are working on their tasks. The status is available as a set of indicators by which the teacher may identify partial progress in specific sub-tasks as well as total progress, considering each student as well as the entire group. For this purpose, information needs to be gathered from diverse sources within the system, and then aggre-
gated into a specific output. Figure 3 presents a specific query that is available
to the teacher (a), and the output that is generated by said query (b). In this
scenario, students were asked to work on a series of exercises related to dis-
tributed programming, each presented to them on a different page. The figure
shows the result for each exercise and each student, creating a matrix showing
the level of advance. The advance starts by “none”, changing as the student
completes necessary programming steps.

On exercise on page 1, the student is expected to create a collaborative
session on a server by using some programming methods previously discussed
in class. Optionally, the student can verify whether the session was successfully
created, and the teacher can see which students have completed the optional
part (“create + Ver” instead of “createOnly”). On page 2, the exercise asks
the students to connect to the server and fetch all existing sessions, to print
them out. On page 3, the students need to connect to a particular session,
but many times the students forget to first check whether that session exists
(see students “pedro” and “juan” who only join, without getting the list first).
Figure 3: Determining student pace and specific solutions

Similar partial results exist for exercise on page 4, where students are expected to create modifications in a session and on page 5, where they are asked to fetch data from a session and print it out.

Using activity levels the teacher can identify which students are currently working on the system and which ones are likely to need assistance, making her aware of the pace of both, individual students and the whole group. Students progressing either slower of faster than expected will be noticed quickly by the teacher, making it possible to take a closer look, either by directly approaching the student, or by using a different query to have a closer look at the student’s
work. When a student is advancing faster than his peers, the teacher might want to provide additional challenges, ideas or assignments in order to keep stimulating the advanced students, or show the work to other members of the class in order to discuss the solution and alternatives. On the other hand, when a student is having problems to solve the proposed exercise, the teacher can take adequate measures to overcome the specific problems that are being identified.

In this scenario, a particular solution can be characterized sufficiently for the teacher to automatically identify students who have successfully solved a problem. This can also be applied for identifying students making common errors or omissions. In figure 3 part (b), the teacher can see that in the different stages, students can have fulfilled none, a part or the whole exercise. The columns show the status of each student on the exercise on a specific page, allowing the teacher to compare easily each student’s activity. For example, the student identified by username “jorge” has not finished exercises on pages 3 to 5, but was the only student to achieve more than the rest in exercise on page 1. In this case, jorge actually did more than was asked in the exercise, while
Pedro finished all exercises, but did not complete exercise on page 3 completely.

During the session, the system correctly identified the progress made by the students, allowing the teacher to be aware of the current state of the work performed by the students. The hints given by the teacher to the students were relevant, and they helped to guide the class in a natural way. In particular, it was possible to identify early on that at least five students were not making any progress at all, despite the fact that they seemed to be very busy working on the exercises. Of these students, only two approached the teacher or asked for help. Upon a closer look it was clear that a technical problem was keeping the CiC plug-in from working properly, so the students had no way to verify their exercise did work. Changing the laptop for a few of these students and having others working in groups with computers that were working correctly solved the problem, not only for the two students who proactively asked for help, but for all five of them.
5.2 Assessment of correctness of solutions and interaction

In the next scenario, we intend to show how the querying system is capable of identifying when a student uses one of multiple possibly right ways of resolving a problem. We presented a problem in which students had to model and simulate a stochastic process, which could either be done in a natural but cumbersome way by following the description literally, or by noting that the problem was identical to a problem that was much easier to model. The teacher had access to a query that identified which path each student seemed to have chosen. With this information the teacher could start discussions in which students evaluate their peer’s solutions and learn more than one way to solve the problems.

In this scenario we also added information about the collaboration activities on behalf of the students, for the teacher to determine whether the group is doing real group work or each student is advancing on their own. For this purpose, the teacher has access to statistics for each group of students, identifying more active students and passive ones. In figure 4, we can see the query that provides
Figure 4: Student activity within a group

the statistics (a), and the results within one specific group (b).

The statistics gathered as the session progresses did not show major changes
in the distribution of actions among the participants. Generally it was possible
to identify one or two students in a group that had a participation that
was slightly over the level of the other participants. As the teacher can see
these numbers, it is possible to take a closer look at the groups that show a
pronounced difference of participation among its members, to verify that they
are for example collaborating using direct face to face communication, or take
action when a problem is encountered.
It was also possible to use queries that identified common errors like using the wrong element for modeling a particular stochastic scenario. As expected, several users made a quite common mistake, resulting in wrong results. When the teacher detects this kind of errors, she determines the best way to handle the situation, for example asking for differing results, starting a discussion and either letting students find out why the results are different and which result is correct, or taking a more direct approach by telling the correct solution.

5.3 Assessment of complex models

In the case of complex models, it is increasingly difficult to characterize a “right” or a “wrong” solution, and the teacher has to use more generic information to determine where to look for problems or right answers. The queries used in this scenario are generally applicable to any situation.

It was possible to identify cases in which students were working on models that appeared extremely complex when compared to the model solution available to the teacher. In all cases where the difference was above twice as many
nodes and edges as expected, a close look revealed that they were creating a
new model on the same page as the first one. Other students also decided to
start a new model because the first one was not satisfactory, but by creating the
new model on another page, or by deleting the previous model before starting
over. The three approaches seen in this session do provide different results. A
teacher needs to be able to interpret these results, verify that the real cause of
the results effectively fits with the possible interpretations (e.g. by approach-
ing a student or using a query to view the students model) and possibly take
remedial actions.
It is not always necessary for each teacher to make their own interpretation of possible causes for the results of a query. The courseware should include the reasoning and possible interpretations of the results. A teacher should expect the courseware to include the queries that are specific to this classroom situation or activity. Additionally, the courseware should include some suggestions of how to react based on the results. For example, in this case it should provide range of values and relations among values that have been found to be typical in a setting in which no problems arise. Additionally, the courseware should indicate values and relations among values that have been found in situations where certain problems occurred, such as students not knowing how to use the modeling tools, lack of familiarity with the theory of the model, or when the students have already solved the posed problems. A teacher may interpret the results accordingly, verify or alternatively disambiguate the interpretation and decide what action to take by using these hints.

In the examples shown in this chapter the teacher had no need to perform changes on the queries more than modify at most one parameter for a specific
query. Thus, it is not possible to make any evaluation on the usability of the querying system on a deeper level.

6 Conclusions

From the experiments described in the previous chapter, we can see that it is in fact possible to automatically extract valuable information from the worksheets the students are working on to assess their work and give meaningful feedback at the right moment at least in three different ways:

1. Tracking the students in order to monitor their advance on the work.

   This gives the opportunity to assist students which might be advancing at a slower pace than the rest of the class or identify advanced students to give them positive feedback and/or provide them with more problems to solve.

2. Identify students that might have reached a correct solution in order to show them to the rest of the class in case their solution is a classic or
a unexpected one. It can also serve to identify cases where students are developing too complex solutions compared with the “classical” solution the teacher might have in mind. In these cases, the teacher can assist the students guiding them in order to help them find simpler solutions.

3. Monitoring the level of activity of the students either during individual or collaborative learning sessions. A low level of activity can indicate the students are not well prepared to solve the proposed task or that the task itself could be ill formulated. It can also serve as an indicator that perhaps the switching between different learning activities (e.g. from individual solving problem to collaborative work) might be not well designed.

The last point of the list indicates that this tool can also be used to validate or discard a set of planned learning activities as “best practices”.

A very important characteristic of this approach is its generality. In fact, the developed query mechanism can be applied to any computer supported learning environment in which the documents on which the students should work
on are structured following the DOM standard and have their corresponding representation in XML. This is because the query system is based only in the comparison of XML documents and does not use the semantical meaning of them. In fact, we think that the most important contribution of this work is the automatic assessment approach based on queries, and not the system developed to validate the feasibility of our hypothesis.

By using a modeling tool as the example to test our approach we show that automatic assessment by querying the documents the students are working on is very powerful. This is indeed a complex scenario since in many cases more structured tasks can be given to the students thus making the querying much easier. For example, the working sheet might be reading a text and answering questions about its content with multiple choices or a list of mathematical exercises with unique answers, like in the case of additions, subtraction, multiplication and/or division problems.

Finally we would like to point to the fact that this approach does not only apply to the monitoring of documents being worked out by students but it can
also applied to other scenarios: for example, this approach can be used to flexibly control the evolution of the documents in a workflow system to find out if certain processes are advancing at the required path.

References


cational Multimedia, Hypermedia & Telecommunications (ED MEDIA),
June 2007.


[6] J.P. Hasebrook and H.A. Maurer, Learning Support Systems for Organiza-

intrusive active document annotation via adaptive information extraction,”
Workshop Semantic Authoring Annotation and Knowledge Management
(European Conf. Artificial Intelligence), 2002.

for the specification and development of an environment for distributed cscl


