International Journal of Computer Science and Informatics

Volume 1 | Issue 1 Article 10

July 2011

Supervising Online Examination Systems Using Scientific Perception on Data

K. Bharathi
M.tech Department of CSE KL University Vaddeswaram,Guntur(dt),A.P, India, barathi.karnam@gmail.com

Dr.S. Balaji
Professor,Department of ECM KL University Vaddeswaram,Guntur(dt),A.P, India, ecmhod_klce@yahoo.com

P. Sindhu
M.tech Department of CSE KL University Vaddeswaram,Guntur(dt),A.P, India, podilasindhu@gmail.com

Recommended Citation

Bharathi, K.; Balaji, Dr.S.; and Sindhu, P. (2011) "Supervising Online Examination Systems Using Scientific Perception on Data," *International Journal of Computer Science and Informatics*: Vol. 1 : Iss. 1 , Article 10.
DOI: 10.47893/IJCSI.2011.1009
Available at: https://www.interscience.in/ijcsi/vol1/iss1/10

Supervising Online Examination Systems Using Scientific Perception on Data

K.Bharathi
M.tech Department of CSE
KL University
Vaddeswaram,Guntur(dt),A.P, India
barathi.karnam@gmail.com

Dr.S.Balaji
Professor,Department of ECM
KL University
Vaddeswaram,Guntur(dt),A.P, India
ecmhod_klce@yahoo.com

P.Sindhu
M.tech Department of CSE
KL University
Vaddeswaram,Guntur(dt),A.P, India
podilasindhu@gmail.com

Abstract: We present an approach and a system to let tutors monitor several important aspects related to e-tests, such as learner behavior and test quality. The approach includes the logging of important data related to learner interaction with the system during the execution of e-tests and exploits scientific visualization to highlight information useful to let tutors review and improve the whole assessment process. We have focused on the discovery of behavioral patterns of learners and conceptual relationships among test items. Furthermore, we have led several experiments in our faculty in order to assess the whole approach. In particular, by analyzing the scientific visualization charts, we have detected several previously unknown test strategies used by the learners. Last, we have detected several correlations among questions, which gave us useful feedbacks on the test quality.

Keywords: Distance learning, scientific perception, interactive data exploration and knowledge discovery.

I. INTRODUCTION

E-TESTING systems are being widely adopted in academic environments, as well as in combination with other assessment means, providing tutors with powerful tools to submit different types of tests in order to assess learners' knowledge. Among these, multiple-choice tests are extremely popular, since they can be automatically corrected. However, many learners do not welcome this type of test, because often, it does not let them properly express their capacity, due to the characteristics of multiple-choice questions of being "closed-ended." Even many examiners doubt about the real effectiveness of structured tests in assessing learners' knowledge, and they wonder whether learners are more conditioned by the question type than by its actual difficulty.

In order to teach learners how to improve their performances on structured tests, in the past, several experiments have been carried out to track learners' behavior during tests by using the think-out-loud method: learners were informed of the experiment and had to speak during the test to explain what they were thinking, while an operator was storing their words using a tape recorder. This technique might be quite invasive, since it requires learners to modify their behavior in order to record the information to analyze [1], [2], [3], [4], which might vanish the experiment goals, since it adds considerable noise in the tracked data. Nevertheless, having the possibility of collecting data about learners' behavior during tests would be an extremely valuable achievement, since it would let tutors exploit many currently available data exploration and knowledge discovery (KDD) strategies to elicit important insights on the testing activities that can be used to teach learners how to improve their performances. However, it would be desirable to devise noninvasive data collection strategies that do not influence learners' behavior during tests, so as to convey more faithful feedbacks on the testing activities.

In this paper, we present a solution enabling the recording of learners' habits during e- tests without informing them of the underlying experiment and, consequently, without asking them to modify their behavior, which potentially yields more realistic results.

Regarding the exploration of the collected data, several KDD techniques could be used. Classical data mining algorithms aim to automatically recognize patterns in the data in order to convey knowledge. However, classical data mining algorithms become inappropriate in several situations such as in multidimensional data and data not uniformly distributed. One way to overcome these problems is to use proper visual representations of data in order to stimulate user involvement in the mining process.

In particular, information visualization can potentially enhance the human capability to detect structures, patterns, and relationships between data elements while exploring data. Information visualization is defined as the use of interactive visual representation of abstract data to amplify cognition. In the past, information visualization has been successfully used in an e-learning application to measure the participation of the learners to online activities.

In this paper, we propose a data exploration approach exploiting information visualization in order to involve tutors in a visual data mining process aiming to detect structures, patterns, and relations between data, which can potentially reveal previously unknown knowledge inherent in tests, such as the test strategies used by the learners, correlations among different questions, and many other aspects, including their impact on the final score.

Finally, we also present a system implementing the proposed approach. The system logs all the interactions of learners with the e-testing system interface. In particular, it captures the occurrence of question browsing and answering events by the learners and uses these data to visualize charts containing a chronological review of tests. Other than identifying the most frequently employed strategies, the tutor can determine their effectiveness by correlating their use with the final test scores. Moreover, the proposed system helps tutors to detect correlations among questions: if a question contains a suggestion to solve other questions, this is made easily visible in the charts. Last, we give a first glance at the way our system can be used for detecting unethical behaviors from the learners, such as cheating by watching on others' screens and gaming-the-system attempts. The system is Web based and relies on the AJAX technology in order to capture all of the learners' interactions with the e-testing system interface (running in the Web browser). The system is composed of a logging framework that can be instantiated in any e-testing systems and a stand-alone application that analyzes the logs in order to extract information from them and to graphically represent it.

In order to demonstrate the effectiveness of the approach and of the proposed system, we have used them in the context of a university course offered in our faculty: the framework has been instantiated in an existing e-testing system, e-Workbook, which has been used to administer e- tests to learners. The grade obtained on the tests has concurred to determine the final grade of the course exam.

II. THE SYSTEM

In this section, we describe the system implementing the proposed approach. The system is composed of a Logging Framework and a Log Analyzer application. The former, based on the AJAX technology, captures and logs all of the learners' interactions with the e-testing system interface (running in the Web browser). It can be instantiated in any e-testing system and is further composed of a client-side and a server-side module. The latter is a stand-alone application that analyzes the logs in order to extract information from them and to graphically represent it.

A. The Logging Framework

The purpose of the Logging Framework is to gather all of the learner actions while browsing web pages of the test and to store raw information in a set of log files in XML format.

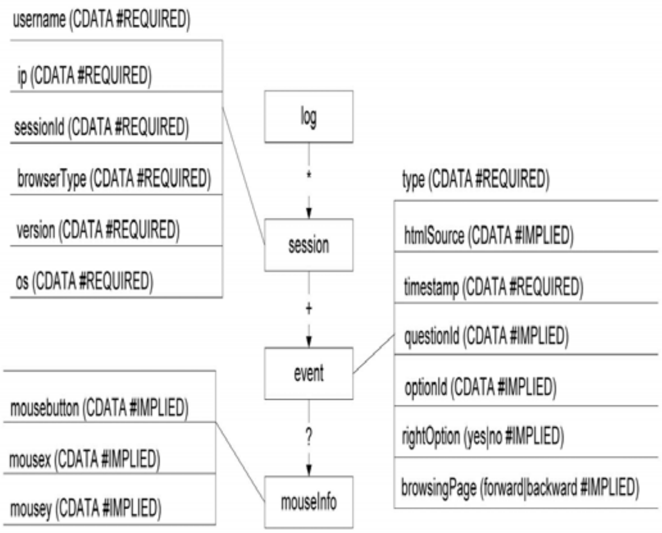

username (CDATA #REQUIRED)
ip (CDATA #REQUIRED)
sessionId (CDATA #REQUIRED)
browserType (CDATA #REQUIRED)
version (CDATA #REQUIRED)
os (CDATA #REQUIRED)
log
*
session
+
event
?
mouseInfo
mousebutton (CDATA #IMPLIED)
mousex (CDATA #IMPLIED)
mousey (CDATA #IMPLIED)
type (CDATA #REQUIRED)
htmlSource (CDATA #IMPLIED)
timestamp (CDATA #REQUIRED)
questionId (CDATA #IMPLIED)
optionId (CDATA #IMPLIED)
rightOption (yes|no #IMPLIED)
browsingPage (forward|backward #IMPLIED)

Fig. 1. The information Model for log data

The framework is composed of a client-side and a server side module. The former module is responsible for "being aware" of the behavior of the learner while s/he is browsing the test's pages and for sending information related to the captured events to the server-side module.

The latter receives the data from the client and creates and stores log files to the disk. Despite the required interactivity level, due to the availability of AJAX, it has been possible to implement the client-side module of our framework without developing plug-ins or external modules for Web browsers.

JavaScript has been used on the client to capture learner interactions, and the communication between the client and the server has been implemented through AJAX method calls. The client-side scripts are added to the e-testing system pages with little effort by the programmer.

The event data is gathered on the Web browser and sent to the server at regular intervals. It is worth noting that the presence of the JavaScript modules for capturing events does not prevent other scripts loaded in the page to run properly.

The server-side module has been implemented as a Java Servlet, which receives the data from the client and keeps them in an XML document that is written to the disk when the learner submits the test. The Logging Framework can be instantiated in the e-testing system and then enabled through the configuration.

The information Model used for the log data is quite simple and is shown in Fig. 1. The information is organized per learner test session. At this level, the username (if available), IP address of the learner, session identifier, and agent information (browser type, version, and operating system) are logged. A session element contains a list of event elements. The data about user interactions are the following:

- event type,
- HTML source object involved in the event (if present),
- mouse information (pressed button and coordinates),
- timing information (time stamp of the event), and
- more information specific of the event, i.e., for a “response” event (a response given to a question), the question and option identifiers with the indication of whether the response was right or wrong are recorded.

An important concern in logging is the log size. If an experiment involves a large set of learners and the test is composed of many questions, log files can reach big sizes.

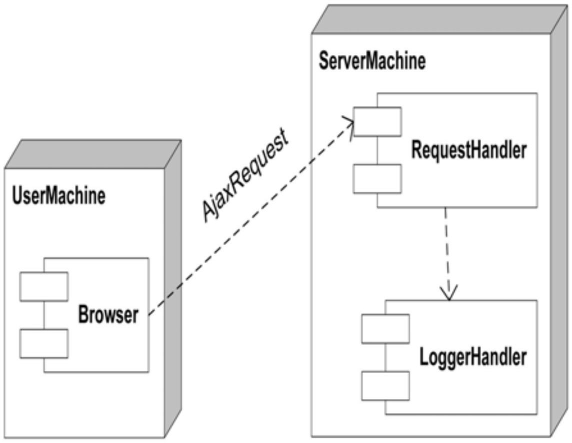


Fig. 2. The logging framework architecture.

A configuration system, including the following configuration settings, has been conceived in order to reduce log sizes:

- list of events to capture,
- subset of attributes for each event,
- sections of the web pages (divs or table cells) to be monitored as event sources,
- time interval between two data transmissions from the client to the server, and
- sensitivity for mouse movements (short movements are not captured).

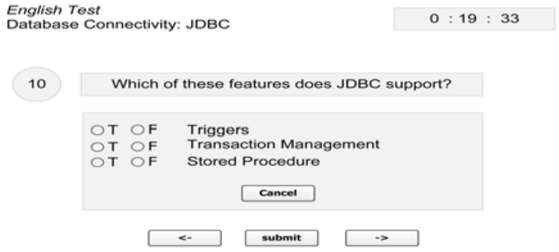
On the client side, everything can be done in the Web browser. The JavaScript modules for event capturing are dynamically generated on the server according to the configuration settings, are downloaded, and run in the browser interpreter. Data are sent to the server through an AJAX request. On the server side, a module called RequestHandler receives the data and sends them to a module called LoggerHandler, which organizes the XML document in memory and flushes it to the disk every time a learner test session finishes. The architecture of the framework is graphically represented in Fig. 2.

B. Instantiation of Logging Framework in the eWorhbook System

The above described framework has been instantiated in an existing Web-based e-testing system, named eWorkbook, which is used for evaluating the learner’s knowledge by creating (the tutor) and taking (the learner) e-tests based on multiple-choice question types. eWorkbook adopts the classical three-tier architecture of the most common J2EE Web applications. The Jakarta Struts framework has been used to support the Model 2 design paradigm, a variation of the classic Model View Controller (MVC) approach. Struts provides its own Controller component and interacts with other technologies to provide the Model and the View. In our design choice, Struts works with JSP for the View, while it interacts with Hibernate, a powerful framework for object/ relational persistence and query service in Java, for the Model.

The application is fully accessible through a Web browser: no plug-in installations are needed, since it only delivers fully accessible HTML code and the included client-side scripts conform to the standard ECMAScript language. It is worth noting that the system has been tested on recent versions of most common browsers (i.e., Internet Explorer, Netscape Navigator, FireFox, and Opera).

The integration of the Logging Framework in the eWorkbook system has been rather simple. For the integration of the server-side component, the Java ARchive (JAR) file containing the framework classes has been imported as a library in the system. Furthermore, a modification to the system's deployment descriptor has been necessary. For the integration of the client-side component, it has been necessary to consider the structure of the eWorkbook system interface. The test is launched in a child browser window of the system web page. This window displays a timer to inform the learner of the time left to complete the test and contains the controls to flow (forward and backward buttons) between the questions and the button to submit the test. The stem and the form containing the options are loaded in an iframe window placed in the center of the page. A screenshot of the web page showing the test interface is shown in Fig. 3. The JavaScript modules for capturing the events have been included in both the external window and the internal iframe, while the modules for communicating with the server have only been included in the main page. In order to identify the source element of the events, the HTML code of the eWorkbook interface has been slightly modified by adding HTML element identifiers to several elements (form interactive elements such as labels and radio buttons).



English Test
Database Connectivity: JDBC

0 : 19 : 33

10 Which of these features does JDBC support?

○T ○F Triggers
○T ○F Transaction Management
○T ○F Stored Procedure

Fig. 3. Screenshot of the test execution.

C. The Log Analyzer Application

The data analysis and test visualization operations are performed by a Web-based stand-alone application, optionally hosted on the same server of the Logging Framework, which takes as input the XML log files. The Log Analyzer Application is composed of two modules: the Query Engine and the Chart Generator. The Query Engine module performs queries on the log files in order to obtain the desired data as an instance of the data model shown in Fig. 1. Since the log files are in XML format, the queries necessary to perform the above operation have been expressed in the XQuery language and have been performed on the log files by using an implementation of the JSR 22—XQuery API for Java. Once the data in the format shown in Fig. 1 are obtained, these are given as input to the Chart Generator module. This module has been implemented through a Java Servlet, which takes the identifier of the learner whose test is going to be analyzed as a parameter. The module dynamically generates the chart and returns it in the PNG format. To produce the charts, the Standard Widget Toolkit (SWT) API has been used.

The choice of dynamically constructing charts reduces the space necessary for the storage of the images: for caching purposes, only images obtained within the browser session are saved on the server and are deleted at the end of the session. With this approach, the computation necessary to construct each visualized chart is executed once per test analysis session, even when the same chart is visualized more times during the same session.

III. CONCLUSION

We have presented an approach and a system to let tutors monitor learners' strategies during e- tests. The approach exploits scientific visualization to draw the data characterizing the learner's test strategy, in order to trigger the tutor's attention and to let him/her discover previously unknown behavioral patterns of the learners and conceptual relationships among test items. In this way, the tutor is provided with a powerful tool that lets him/her review the whole assessment process and evaluate possible improvements.

We have extensively used the implemented system experimentally to evaluate e-test strategies in the courses of our faculty, in order to assess the whole approach.

This lets us discover several relevant patterns regarding the test quality, the characteristics of used strategies, and the impact on the final score.

IV. REFERENCES

[1] J. Bath, "Answer-Changing Behaviour on Objective Examinations," J. Educational Research, no. 61, pp. 105-107, 1967.

[2] J.B. Best, "Item Difficulty and Answer Changing," Teaching of Psychology, vol. 6, no. 4, pp. 228-240, 1979.

[3] J. Johnston, "Exam Taking Speed and Grades," Teaching of Psychology, no. 4, pp. 148-149, 1977.

[4] C.A. Paul and J.S. Rosenkoetter, "The Relationship between the Time Taken to Complete an Examination and the Test Score Received," Teaching of Psychology, no. 7, pp. 108-109, 1980.

[5] "Gennaro costagliola,Massimilialo and Giorddano and Giuseppe polese","Monitoring etests through visualization",2009.