A Data Model to Ease Analysis and Mining of Educational Data[1]

André Krüger[1,2], Agathe Merceron[2] and Benjamin Wolf[2]
{akrueger, merceron, bwolf}@beuth-hochschule.de
[1]Aroline AG, Berlin, Germany
[2]Beuth University of Applied Sciences, Berlin, Germany

Abstract. Learning software is not designed for data analysis and mining. Because usage data is not stored in a systematic way, its thorough analysis requires long and tedious preprocessing. In this contribution we first present a data model to structure data stored by Learning Management Systems (LMS). Then we give an overview of the system architecture that performs the structure/ export functionality and of its implementation for the Moodle LMS. Finally, we show first results using this data model for analysing usage data.

1 Introduction

It is a well known fact that data understanding and preprocessing constitutes the main work of the data analysis and mining process. Learning software is not designed for data analysis and mining. Even if many learning software do store usage data, they are designed to support learning and teaching, not to analyse the data they store. However, the field of educational data mining is emerging precisely because valuable pedagogical information is gained from analysing and mining data stored by educational software [17, 1]. As a consequence, the process of educational data mining requires long and tedious preprocessing as mentioned in various works, see [2, 13, 20] for a few examples.

In this contribution we present a data model to structure and export usage data stored by learning software. Note that separating the data used for the business, in our case the data stored by the learning system, from the data used for analysis and mining is well in the line of the usual approach in the data mining field, see for example [4]. The aim of this data model is to automate, at least partly, the usual long and tedious preprocessing and to facilitate the data exploration step that should precede data mining. Our data model is primarily oriented towards the particular educational software called Learning Management Systems (LMS). We have designed a modular and extensible architecture to realise the structure/export function and we present an implementation for the particular LMS Moodle [14]. Our system handles data that have been rendered anonymous. We show first results using this data model for analysis of usage data.

1.1 Background and Related Works

In our own university and in most universities at least in Germany, LMS are used both in distance education and face to face teaching. They are becoming a must-have in distance education [18]. They are more and more used in face to face teaching as they make the administration of a course much easier for teachers, and provide a handy way to cater for special needs. Experienced teachers can handle better a diverse classroom as

[1] This work is partially supported by the European Social Fund for the Berlin state.

supplementary learning materials can easily be made available through them for example. LMS provide a virtual place where students can find learning materials, study guides as well as an overview of their results, and where they can communicate with teachers and with other students. The functionality of an LMS can be divided in three main parts: Management of learning resources, management of users, and communication between users [18]. However statistics and reports are usually basic.

To illustrate this last point, we list below a few questions that reporting facilities of most of the currently used LMS cannot handle:

1.How many students have never viewed Learning Resource A?

2.If students do well on activity B, do they also do well on activity C?

3.If students solve exercise D, do they also solve exercise E?

4.What is the average mark on quiz G got by students who have viewed resource F?

5.Which courses use a lot of Audio Learning Resources?

To manage properly learners, especially distance learners, it is quite important to gain a good overview of their learning behaviours. To manage properly degrees, it is important to harmonize the different modules offered in a degree. The aim of our work is to complement LMS in all aspects dealing with data analysis and mining.

Many works to analyse and mine data stored by LMS have been undertaken, see [17, 1] for some overview. To the best of our knowledge, all these works do some ad hoc preprocessing of the data. They do not aim at proposing some data model for analysis and mining that could be shared by all LMS independently of their internal structure. The work undertaken in [16] bears similarity with our work in the sense that it considers all data stored by an institution in higher education and unifies it into a model to explore behaviours of students. Another work that bears similarities to our work is PSLC Datashop [9], an open repository of educational data. The analysis tools offered in Datashop are more suited for educational software such as intelligent tutoring systems. Our work is not concerned with discovering or sharing learning objects as other works such as [19, 7] do. It is concerned with describing and structuring the interactions of users with learning objects as stored in LMS. Our vocabulary to describe these interactions is partly borrowed from the vocabulary adopted by the LMS Moodle [14], as Moodle is used by a large community worldwide. The part related to the interactions of users with quizzes contains elements that the IMS specification [8] also contains, though it is much simpler that [8].

This paper is organised as follows. The following section introduces our data model. The third section gives an overview of the export tool that exports the data stored by an LMS into this data model. Section 4 presents a case study in analysing data stored in a specific course using our tool. Last section concludes this paper.

2 The Data Model

The data model we present is quite close to a fact constellation schema [4]. It contains three kinds of tables: tables to describe objects found in LMS, these tables can be seen as dimension tables; tables to describe interactions with learning objects, these tables can be

seen as fact tables; and third, association tables to describe associations between objects. The choice of having a table per object comes from the observation that most LMS have a limited set of objects that teachers are used to handle. Adopting this same set should make it easier for teachers to analyse usage of resources by students.

2.1 The schema

Our data model makes several general assumptions that we expose now. First we assume that an LMS contains users and courses. Users can enroll or sign in courses and sign off courses. Users can have roles like "lecturer", "administrator", "tutor", "student" and so on. A user may have different roles in different courses. For example a user can be a tutor in the course "Introduction to Programming" and a student in the course "Early American History". An LMS may contain groups that are associated to courses. Students enroll in those groups. An LMS contains forums, wikis, resources and quizzes. We call a quiz any kind of assignment, exercise or test a lecturer may wish to give to students. Forums, wikis, resources and quizzes are associated to courses. Thus a resource for example, can be used in several courses. A quiz may contain one or more questions that are also contained in the LMS. Questions are associated to quizzes and a given question can be associated to several quizzes. These general assumptions cover the particular case of LMS where resources, forums, wikis or quizzes exist only inside a given course. In this particular case an association table contains only one tuple. We assume that an LMS logs or stores interactions of users. For any given interaction, the LMS stores the identification of the user, of the course, of the resource, forum, wiki, quiz, as well as the timestamp, the nature of the interaction ("view", "modification", "creation", "attempt", "submit" and so on), the marks and the contribution when relevant.

We present now in more details the tables that are the most useful for our case study, see Figure 1. For the current full set of tables, we refer to [11, 12]. Every table contains an element *id*, which is the key or identifier of the tuple.

The five tables below describe objects usually found in LMS.

Table user: This table describes users registered in the LMS. The elements *firstaccess* and *lastaccess* are the times and dates when a user first and last accessed any kind of learning object, such as a resource, a quiz etc. in the LMS. The elements *lastlogin* and *currentlogin* are the times and dates a user logged in the LMS for the last time, respectively currently. Note that a user can log in without accessing any learning object.

Table course: This table describes courses existing in the LMS. We assume that a course exists for a given period of time. The element *timecreated* is the date and time the course has been created, usually by the administrator. The element *startdate* is the time and date the course is supposed to start, this time is usually fixed by the lecturer in charge. The element *enrolstart* is the date users are allowed to enroll in this course, and the element *enrolend* is the date users can not enroll anymore in the course. The element *timemodified* is the time and date this course has been last modified. The elements *title* and *shortname* are self explanatory.

Table quiz: This table describes quizzes existing in the LMS. As already mentioned, we call a quiz any kind of assignment or test a lecturer gives to students. The element *qtype* is the type of the quiz. It can take values such as "assignment", "SCORM" and so on,

according to the different kinds of quizzes an LMS makes available. The elements *qid* combined with *type* make up the identification of a quiz. A quiz may contain one or more questions, see table **question**. The element *title* is a title the lecturer in charge gives to this quiz. The elements *timeopen* and *timeclose* refer to the dates and times students are allowed to answer the quiz, while the element *timecreated* is the date and time the quiz has been created and the element *timemodified* is the date and time the quiz has been last modified.

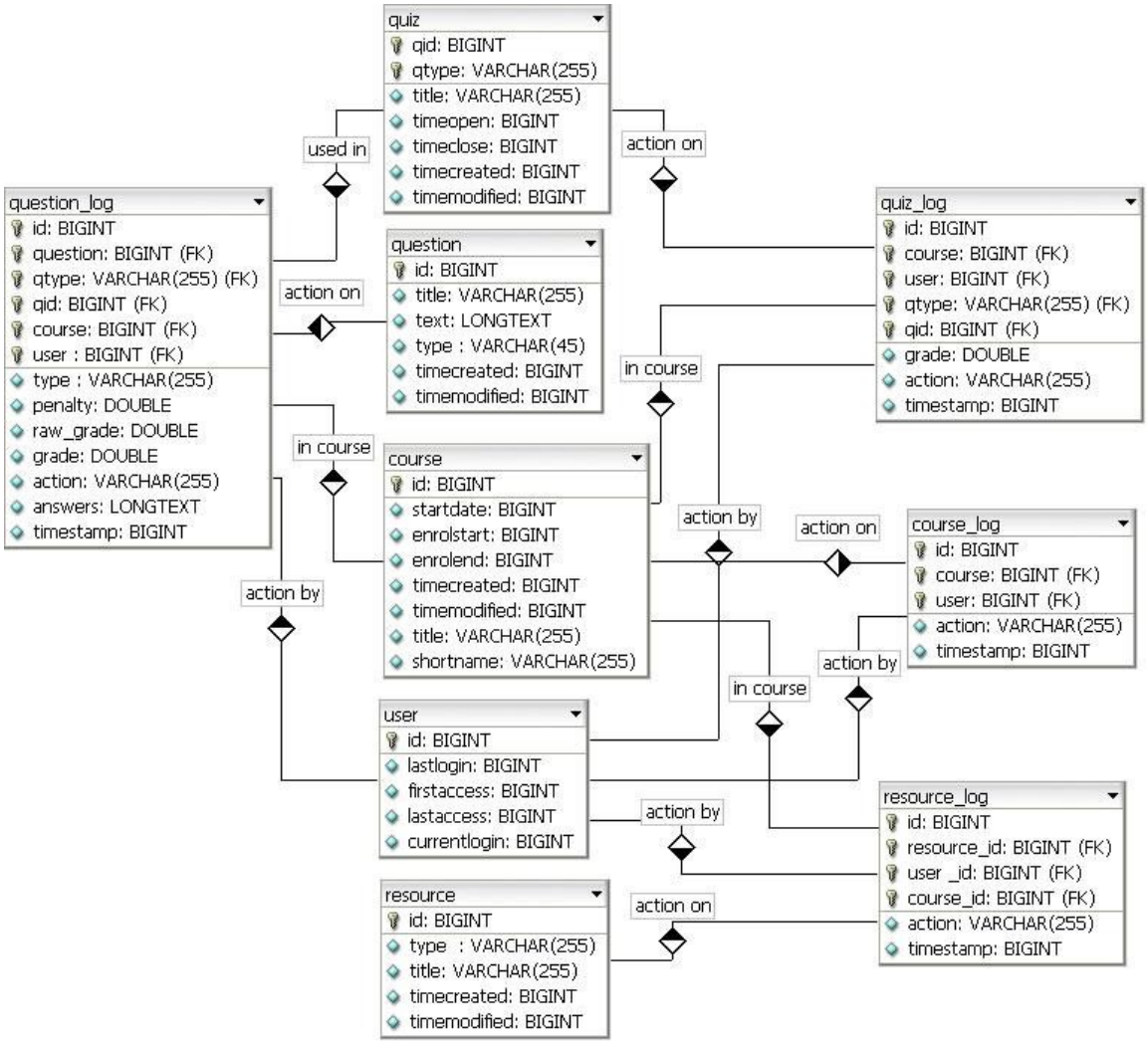

Figure 1. Snapshot of the relational schema

Table question: This table describes questions that make up quizzes. The element *title* is the title of the question, while the element *text* is the actual text of the problem to solve. The element *type* is a category like "multiple-choice", "true-false" etc. The elements *timecreated* and *timemodified* are as described in the table quiz.

Table resource: This table describes resources available in the LMS that lecturers may use in courses. The element *type* describes the type of the resource like "file", "uri", "directory", "audio", "picture" and so on. The elements *timecreated* and *timemodified* are as for quiz. The element *title* is the title of this resource like "transparencies01".

The three following tables describe interactions with learning objects. They are the facts that are stored while users use objects of an LMS.

Table quiz_log: This table describes the information that a LMS stores when users interact with quizzes. The element *user* is the *id* (the key) of the user who interacted. The element *course* is the *id* (the key) of the course in which the interaction took place. The elements *qid* and *qtype* refer to the quiz that was tackled. The element *grade* is the mark obtained in the quiz. The element *timestamp* gives the date and time of the interaction. The element *action* gives the kind of action that took place. An action can be "view", in that case the user simply looked at the quiz, "attempt", in that case the user attempted the quiz, "submit", in that case the user attempted and finished the quiz, "modify" if the quiz has been modified etc. .

Table question_log: This table describes the information that a LMS stores when users interact with a question of a quiz, and contains all elements already included in the table **quiz_log**. The element *penalty* gives the penalty marks given in that interaction. If a quiz is run in adaptive mode then a student is allowed to try again the question after a wrong answer. In this case one may want to impose a penalty for each wrong answer to be subtracted from the final mark for the question. The amount of penalty is chosen individually for each question when setting up or editing the question. The element *raw_grade* gives the raw mark obtained in that interaction. The element *grade* gives the marks for that question in that interaction when *penalty* has been taken into account. It includes also an element *question*, the *id* of the question that was tackled, the elements *type*, which can take values like "multiple choice", "true/false" and the element *answers*, the actual answer or answers, when several answers are allowed, given by the user in the interaction.

Table resource_log: This table describes the information that a LMS stores when users interact with resources. This table contains elements that are similar to the ones of table **quiz_log**.

Finally our data model contains a number of association tables to associate objects with each other, see [11,12].

3 System Architecture

Figure 2 presents an overview of the system architecture. The central part of the system is the abstract class "*ExtractAndMap*" that describes and partly implements functionalities concerning data extraction from an LMS and data generation for the data model. To create the present data model with an LMS, what is needed is to implement the abstract extract methods according to the features of the LMS. The concrete save function can be inherited as is. The system contains an implementation for Moodle. It is implemented in Java, uses the Database Mysql [15] and the persistence framework Hibernate [6].

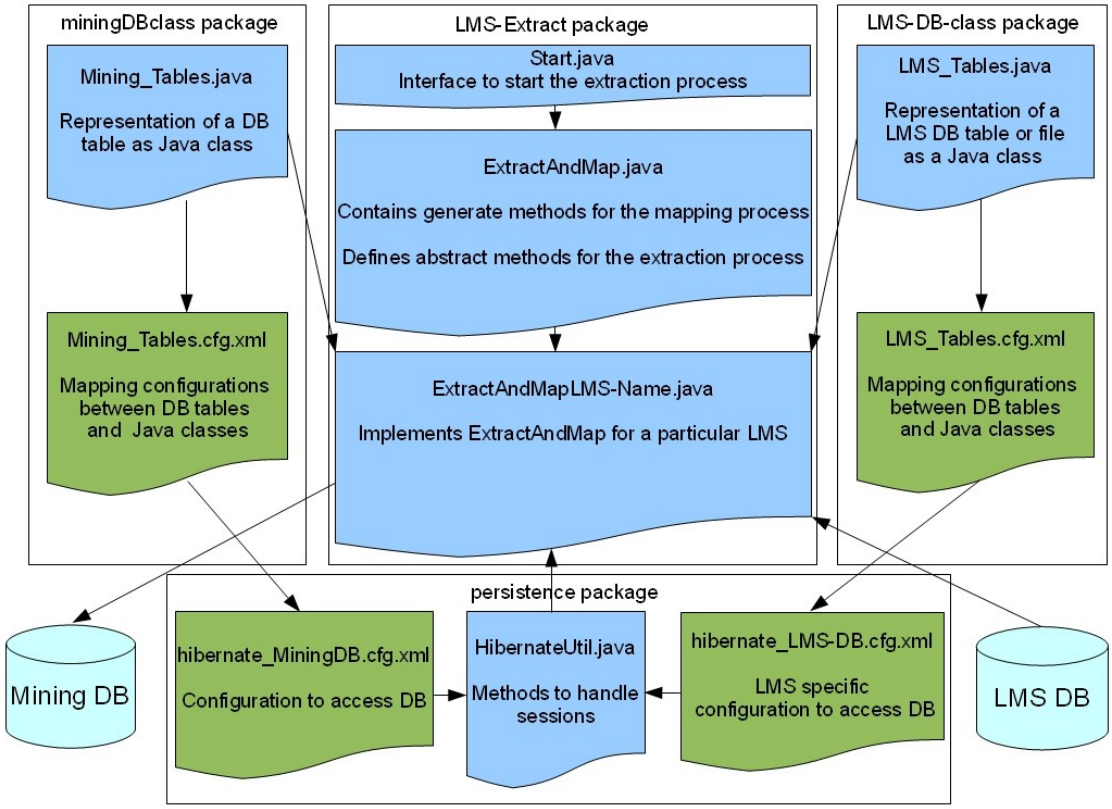

Figure 2. System architecture

4 First Results

We have used our system to analyze the course "Introductory Programming with Java" taught in face to face teaching to first semester students enrolled in the degree "Computer Science and Media" at the Beuth University of Applied Sciences, Berlin, in winter semester 2009/2010. In that semester 65 students were enrolled in this course.

The teaching of this course is supported by the use of the Learning Management System Moodle in which mandatory as well as additional resources are uploaded for students. A list of 8 exercises belongs to the mandatory resources. Students have to solve these 8 exercises to get a mark for the practical part of the course. The lecturer has additionally offered gradually in the semester 7 self-evaluation exercises on key concepts of Java programming like methods, arrays, statements etc.. These exercises are not compulsory. Solving them is left to the sole discretion of the students. The lecturer is interested to know whether students have used these self-evaluation exercises and, most importantly, whether solving or attempting them could have a positive impact on the marks obtained in the final exam. Figure 3, obtained by simple queries, gives an overview of how students have used these exercises. For each exercise the column on the left means *view*, the column in the middle means *attempt* and the column in the right means *close attempt*. Note that *view* means that students have clicked on the resource, *attempt* means that they have submitted a solution and *close attempt* means that they have finished the exercise.

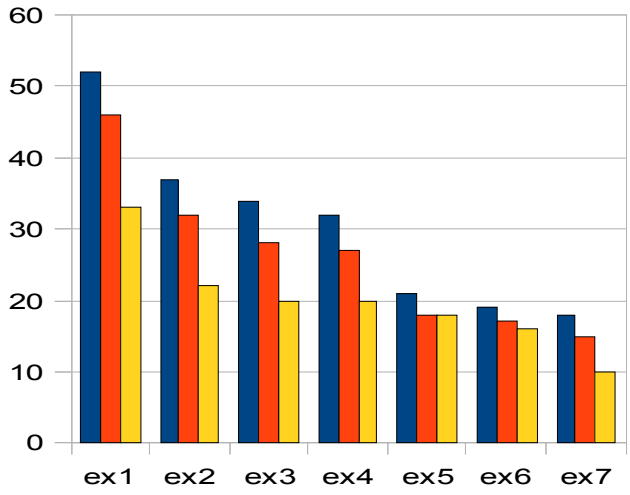

Figure 3. Access to self-evaluation exercises

One notices a pattern that we have already observed in other courses regarding optional self-evaluation exercises [13, 11]: As the semester progresses always less students make use of them. We are interested in investigating whether a dedicated group of students emerges that keep doing the exercises during the semester. Therefore we want to know whether associations such as "if students complete exercise 2, they complete exercise 1" or , "if students complete exercise 3, they complete exercise 2", and so on, hold. For that we have used the method exposed in [10]. Indeed, the answer is positive as Table 1 shows. The association rule *2→1* means "if students complete exercise 2, they complete exercise 1". All rules have a rather high confidence and are rated as interesting both by lift and cosine. We recall that confidence is a number between 0 and 1 (highest is 1), that lift rates a rule as interesting if its value is above 1, and that cosine rates a rule as interesting if its value is above 0.66. Support gives the proportion of the data involved in the rule.

Table 1. Association rules "if students attempt exercise x, they also attempt exercise x-1"

Associations	*2→1*	*3→2*	*4→3*	*5→4*	*6→5*	*7→6*
support	0.34	0.28	0.25	0.2	0.22	0.14
confidence	1	0.9	0.8	0.72	0.88	0.9
lift	1.97	2.66	2.6	2.35	3.16	3.66
cosine	0.82	0.86	0.8	0.69	0.82	0.71

To investigate whether solving these self-evaluation exercises has a positive impact on the marks in the final exam cannot be achieved by correlation or regression analysis as not so many students have solved them. That means for many students we would have missing data, since 45 students have written the final exam. We have simply queried our

data and looked at the average mark on each group of students. The result is given in Table 2.

Table 2. Completing self-evaluation exercises and marks in the exam.

Exercise	min.	max.	mean	s.deviation	meanNoEx
General	1	13	8.63	4.34	7
Ex1	1	13	9.48	4.06	7.33
Ex2	1	13	9.56	3.97	7.95
Ex3	1	13	9.2	4.4	8.26
Ex4	1	13	8.56	4.77	8.68
Ex5	1	13	9.21	4.54	8.29
Ex6	1	13	10.91	3.4	7.70
Ex7	9	13	11.67	1.41	7.69

The line *General* is the minimum, maximum, mean and standard deviation obtained taking all students who have taken part in the final exam. The last column gives the mean for students who have not solved any exercise. The line *Ex1* gives similar results restricting the population to students who have completed the first self-evaluation exercise. The last column gives the mean for students who have not solved the first self-evaluation exercise. And so on till *Ex7*. One notices that the highest average and smallest standard deviation in the final exam is obtained in the group of students who have completed exercise 7 (11.67 and 1.41 respectively), while smallest average is obtained in the group that has not solved any exercise (7). Given the results of the association rules, students who have completed exercise 7 have most probably completed all optional self-evaluation exercises. These results may speak for a positive impact on the final mark of the self-evaluation exercises. However our small population prevents of making any strong conclusion since statistical tests to check the significance of the difference in the average, like t-test, usually require a sample of size 30 or more. The line for exercise 4 looks different and requires more investigation.

5 Conclusion and Future Work

In this paper we have presented a data model to structure and export the data that most LMS usually store in scattered places into an homogeneous schema. The first aim of our data model is to automate and alleviate the preprocessing that is needed to explore, analyse and mine these data. We have designed an architecture of the tool that does the actual structure/export functionality, and implemented it for the LMS Moodle. Finally we have used our tool to analyse the data stored in the course “Programming 1” in our university. We have focused our analysis on the optional self-evaluation exercises. The analysis shows that, as the semester progresses, less students solve them. It shows also that a group emerges that keeps solving them and that reaches slightly better marks in the final exam.

Another aim of this data model is to couple loosely an LMS and the analysis of the data it stores. If the persistence functionality of an LMS is changed, only the structure/export tool needs to be changed, not the analysis tools linked to the data model. Note that the implementation of the module that performs the export functionality for a concrete LMS has to be programmed with particular consideration regarding performance as the data stored inside an institution can be huge. However this programming happens only once.

As noticed in [5] on-line learning is likely to grow, and so is the use of LMS. As pointed out in [1] the number of works tackling data stored by LMS is increasing. We hope that this work will help boost results and best practices in that area.

We have used this data model mainly to explore thoroughly how students use learning resources in a course. Results of such an exploration are enlightening for teachers and are necessary to conduct a better informed data mining afterwards. Will this model be robust enough to answer any pedagogical question using data mining techniques? It depends on who is asking. Our data model contains all interactions that users perform with any object of the LMS along with the timestamp. Therefore a whole range of pedagogical questions related to navigation, performance prediction, activity of students for instance should be treatable.

We begin to notice recurring questions that users of LMS are interested in. A future work is to continue enhancing and structuring those questions, so that we try out our data model further. We aim also at not restricting our data model to LMS, but consider other learning software as well. Our next step in that direction is to consider learning portals. We are also working on a graphical user interface for users to query and mine the data model in an intuitive way. This interface should work as an adaptive front-end for the user, the real analysis work will be done by connecting suitable queries and mining tool already available. Our work is open for the community and will be released soon on [12].

Acknowledgment We thank warmly our colleague Prof. Dr. Ripphausen-Lippa for her cooperation in analyzing the data of her course “Introductory Programming with Java”.

References

[1] Baker, S.J.D.R, Yacef, Y. The State of Educational Data Mining in 2009: A Review and Future Visions. *JEDM - Journal of Educational Data Mining,* 2009, 1(1), p. 3-17.

[2] Dekker, G., Pechenizkiy, M., Vleeshouwers, J. Predicting Students Drop Out: A Case Study. In [3]. p. 41-50.

[3] Barnes, T., Desmarais, M., Romero, C. & Ventura, S. (Eds.), *Proceedings of the Second International Conference on Educational Data Mining,* 2009. Cordoba, Spain.

[4] Han, J., Kamber, M. *Data mining: concepts and techniques,* 2006. Morgan Kaufman publishers.

[5] Hislop G.J. The Inevitability of teaching Online. *Computer, IEEE Computer Society,* 2009, 42(12), p. 94-96.

[6] Hibernate Relational persistence framework. //www.hibernate.org/, 06.05.2010.

[7] IMS Global Learning Consortium Learning Object Discovery and Exchange Project Group. http://www.imsglobal.org/lode.htm, 06.05.2010.

[8] IMS Question and Test Interoperability Results Reporting. http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_resultv2p1pd2.html, 25.04.2010.

[9] Koedinger, K.R., Cunningham, K. A. S., Leber, B.. An open repository and analysis tools for fine-grained, longitudinal learner data. *First International Conference on Educational Data Mining, 2008.* Montreal, Canada, p. 157-166.

[10] Krueger A., Merceron, A., Wolf, B. When data exploration and data mining meet while analysing usage data of a course. *Third International Conference on Educational Data Mining,* 2010. Pittsburgh, USA.

[11] Krueger A., Merceron, A., Wolf, B. Leichtere Datenanalyse zur Optimierung der Lehre am Beispiel Moodle. *Proceedings of the 8. e-Learning Fachtagung Informatik*, 2010. Duisburg, Germany: Lecture Notes on Informatics.

[12] Krueger A., Merceron, A., Wolf, B.. User's Data and Profiles in LMS. http://learn.beuth-hochschule.de/datamining, 13.05.2010.

[13] Merceron, A., Yacef, K. Interestingness Measures for Association Rules in Educational Data. *First International Conference on Educational Data Mining,* 2008. Montreal, Canada, p. 57-66.

[14] Moodle. Learning Management System. //moodle.org, 06.05.2010.

[15] MySQL open source database. http://www.mysql.com/, 07.05.2010.

[16] Pechenizkiy, M., Trcka, N., Vasilyeva, E., van der Aalst, W., De Bra., P. Process Mining Online Assessment Data. In [3], p. 279-288.

[17] Romero, C., Ventura, S. Educational Data Mining: A Survey from 1995 to 2005. *Expert Systems with Applications 33*, p. 125-146.

[18] Schulmeister, R. (2005): *Lernplattformen für das virtuelle Lernen. Evaluation und Didaktik*. 2005. Oldenbourg.

[19] Verbert K., Wiley, D., Duval, E. A Methodology and Framework for the Semi-automatic Assembly of Learning Objects. *Proceedings of the European Conference on Technology Enhanced Learning EC-TEL*, 2009. p. 757-762 .

[20] Vialardi Sacin, C., Bravo Agapito, J., Shafti, L., Ortigosa, A. Recommendation in Higher Education Using Data Mining Techniques. In [3], p. 190-199.