Leichtere Datenanalyse zur Optimierung der Lehre am Beispiel Moodle[1]

André Krüger, Agathe Merceron, Benjamin Wolf

Fachbereich Medieninformatik
Beuth Hochschule für Technik Berlin
Luxemburger Straße 10
13353 Berlin
{akrueger, merceron, bwolf}@beuth-hochschule.de

Abstract: Lernraumsysteme (LMS) werden an Schulen, Hochschulen sowie in Firmen eingesetzt. Berichte und Statistiken gehören nicht zur ihren Kernfunktionalitäten und sind folglich unzureichend vorhanden. Für Lehrende, Bildungsinstitutionen oder Bildungsanbieter sind Informationen über die Nutzungsart und -weise der angebotenen Lehrveranstaltungen aber von entscheidender Bedeutung. Ziel unserer Arbeit ist es, herkömmliche Lernraumsysteme im Bereich Nutzungsdaten und Nutzerprofile zu ergänzen. In diesem Beitrag stellen wir ein Datenmodell vor, um die Nutzungsdaten, die vom LMS gespeichert werden, leichter zu analysieren. Folgend schlagen wir eine Systemarchitektur vor, um die von einem Lernraumsystem gespeicherten Daten in das Datenmodell zu exportieren. Die Implementierung für das LMS Moodle wird vorgestellt. Zum Abschluss erläutern wir die Datenanalyse eines Moodle-Kurses, die wir mit Hilfe unserer Anwendung durchgeführt haben.

1 Einleitung

Formelles wie informelles Lernen in der Grundausbildung sowie das ganze Leben lang nimmt in unserer modernen Wissensgesellschaft eine immer wichtigere Rolle an. Dabei gewinnt e-Learning an Bedeutung. Eine Kern-Anwendung des e-Learnings bilden die Lernraumsysteme, die an Schulen, Hochschulen sowie in Firmen eingesetzt werden. Die Hauptfunktionalitäten der gängigen Lernraumsysteme können in drei Bereiche gegliedert werden: (i) Erstellung, Bereitstellung und Verwaltung von Studiengängen, Kursen, und Lernmaterial; (ii) Verwaltung von Nutzern; (iii) Bereitstellung von Kommunikationstools wie E-Mail, Chat, Foren oder Wikis.

[1] Diese Arbeit wurde zum Teil vom Europäischen Strukturfond Berlin unterstützt.

Nutzerdaten werden in Lernraumsystemen mit dem Hauptziel gespeichert, einen Überblick über die erbrachte Leistung der Lernenden zu bieten. Berichte und Statistiken über Nutzerverhalten und Aussagen zur Beteiligung von Nutzern in Online Kursen und den Ergebnissen gehören nicht zu den Kernfunktionalitäten von Lernraumsystemen. Diese sind folglich auch nur unzureichend vorhanden. Als kleines und dennoch illustratives Beispiel wie unzureichend Statistiken sind, wird an dieser Stelle erwähnt, dass es in Lernraumsystemen nur schwer herauszufinden ist, wie viele Lernende nie auf ein bestimmtes Lernmaterial zugegriffen haben.

Die vorhandenen Statistiken sind selten mit guten Managementmöglichkeiten verknüpft. Für Organisationen im Bildungsbereich sind Informationen zur Nutzungsart und -weise der angebotenen Lehrveranstaltungen und ein für Lehrende leicht handhabbares Management von entscheidender Bedeutung. Letztendlich kann dadurch das eigene Lehrangebot und der erzielte Lernerfolg evaluiert und optimiert werden.

Ziel unserer Arbeit ist es, herkömmliche Lernraumsysteme im Bereich Nutzungsdaten und Nutzerprofile zu ergänzen. Als erstes stellen wir fest, dass Lernraumsysteme Nutzerdaten in einer Form speichern, die für die Analyse ungeeignet ist, und mühsames und langwieriges Preprocessing erfordert [MY08, Vi09]. Wie in Analytical Processing or Data Mining üblich [HK06], ist es sinnvoll ein Datenmodell zu definieren, welches von der Datenspeicherung des Lernraumsystems getrennt ist, und sich für die Analyse besser eignet. Damit kann gleichzeitig erreicht werden, dass als erster Schritt eine Anonymisierung der Nutzerdaten stattfinden kann. Ziel der Analyse ist es, Informationen in den gespeicherten Daten zu finden, die für Lehrende, Bildungsinstitutionen oder Bildungsanbieter relevant sind.

In diesem Beitrag stellen wir ein solches Datenmodell vor. Unser Datenmodell ist ein relationales Schema, das Daten und Nutzungsdaten, die vom LMS gespeichert werden, vereinigt. Im Folgenden stellen wir eine Systemarchitektur vor, um die von einem LMS gespeicherten Daten in das Datenmodell zu exportieren. Dies wurde für das LMS Moodle implementiert. Im Anschluss zeigen wir eine Datenanalyse eines Moodle-Kurses, die wir mit Hilfe unserer Anwendung durchgeführt haben.

Viele Arbeiten beschäftigen sich mit der Analyse von Daten, die von LMS gespeichert werden, siehe [RV07, BY09] für einen Überblick. Zweck solcher Analysen ist es, pädagogisch relevante Information zu finden. Wie beispielsweise herauszufinden, was zum Erfolg/Misserfolg in einem Kurs führt, oder wie das Lernmaterial benutzt wird. Diese Arbeiten setzten meist ad hoc ein mühsames Preprocessing der Daten voraus, um die Daten analysieren zu können. Als Folge daraus bleibt die Analyse beschränkt: sie wird nicht jedes Semester neu ausgeführt, oder es werden nur wenige ausgewählte Kurse analysiert. Ein erstes Ziel ist es, das mühsame Preprocessing weitgehend zu automatisieren. Nur damit kann unser eigentliches Ziel, die Analyse mit Managementmöglichkeiten zu verknüpfen, erreicht werden.

Dieser Beitrag ist wie folgt organisiert. Der nächste Abschnitt führt unser Datenmodell ein. Abschnitt 3 stellt die Systemarchitektur vor. Im Abschnitt 4 wird eine Fallstudie vorgestellt, die mit Hilfe unserer Anwendung geführt wurde. Der letzte Abschnitt enthält eine Zusammenfassung und einen Ausblick.

2 Datenmodell

2.1 Anforderungen

Der Entwurf unseres Datenmodells wurde von drei Hauptanforderungen geleitet. Erstens sollte es unabhängig von einem bestimmten LMS verwendet werden. Zweitens sollte jede Art von Analyse möglich sein. Drittens sollten Dozenten die üblichen Objekte eines LMS leicht wieder finden. Für die erste Anforderung wurden Annahmen getroffen, welche die meisten LMS erfüllen, und die im folgenden Absatz erläutert sind. Für die zweite Anforderung beschreibt unser Datenmodell komplett jede Interaktion, die gängige LMS registrieren, in Log-Tabellen, siehe Abschnitt 2.2. Für die dritte Anforderung spiegelt unser Datenmodell die üblichen Objekte, die LMS zu Verfügung stellen, in Tabellen wider, siehe 2.2. Die Beschreibung der Elemente einer Tabelle orientiert sich sehr am Vokabular des LMS Moodle, das sehr verbreitet ist. Die Beschreibung der Quiz-Interaktionen enthält ähnliche Elemente wie [IMS10], ist aber wesentlich einfacher.

Wir gehen davon aus, dass ein LMS Benutzer, Kurse, Lernmaterial und Kommunikationstools verwaltet. Benutzer können Kurse belegen. Wenn eine Benutzerin einen Kurs belegt, gibt es zwei Daten: das Anmelde-Datum und das Abmelde-Datum. Außerdem hat ein Benutzer in einem Kurs eine besondere Rolle, wie Student, Dozent, Administrator, usw.. Es ist möglich, in einem LMS Gruppen von Nutzern zu bilden. Diese Gruppen werden innerhalb von Kursen gebildet bzw. gelten systemweit. Nutzer können sich diesen Gruppen selbständig zuordnen oder werden durch Dozenten zugeordnet. Eine besondere Kategorie von interaktivem Lernmaterial bildet das Quiz. Das Quiz ist ein Sammelbegriff, der jede Art von Übung, Aufgabe, Problem, Test, Hausarbeit usw. abdeckt. Ein Quiz kann mehrere Fragen enthalten und eine Frage kann mehreren Quiz zugeordnet werden. Sowohl für das Quiz als auch für die Fragen wird ein Zeitstempel gespeichert, der das Erstellungsdatum angibt und ein Zeitstempel, der den Zeitpunkt der letzten Änderung festhält. Des weiteren gibt es Lernmaterialien wie Folien, Dateien, URLs, usw.. Solch ein Material werden wir als Ressource betrachten. Die Ressourcen haben ein Erstellungsdatum und ein Änderungsdatum, die angeben wann diese dem System hinzugefügt wurden und wann sie zuletzt geändert wurden. Als Kommunikationstools betrachten wir zunächst Foren und Wikis. Wir nehmen an, dass Quiz, Ressourcen, Foren und Wikis, Kursen zugeordnet sein können. Wir nehmen an, dass Benutzer mit Lernmaterial interagieren können: Sie können Ressourcen, Fragen und Quiz, Wikis und Foren anschauen oder verändern wenn der Benutzer ein Dozent ist. Außerdem können sie Fragen und Quiz beantworten, an Foren oder Wikis beitragen usw.. Wir nehmen an, dass alle diese Interaktionen mit Zeitstempeln gespeichert werden.

2.2 Schema

Unser Schema beinhaltet drei Arten von Tabellen: die Tabellen, die ein LMS Objekt darstellen, die Tabellen, die eine Interaktion mit einem LMS Objekt darstellen und Tabellen welche Assoziation zwischen LMS Objekten festhalten. Jede Tabelle hat ein Element *id.* Dieses Element ist der Identifikator, auch Schlüssel genannt, einer Zeile. Wir führen die Art der Tabellen nacheinander ein. Ein Überblick des Schemas ist in der Abbildung 1 gezeigt.

Die Tabelle **user** beispielsweise stellt Benutzer dar. Sie enthält als weitere Elemente *firstaccess*[2] und *lastaccess*[3], die die Zeitstempel des ersten und letzten Zugriffs auf einen Gegenstand im LMS sind, und *lastlogin* und *currentlogin,* die die Zeitstempel des letzten und aktuellen Logins in das LMS sind. Es kann der Sonderfall eintreten, dass sich Teilnehmer in das LMS einloggen ohne auf Lernmaterial zuzugreifen.

In der Tabelle **course** werden die Kurse abgebildet. Das Element *timecreated* ist der Zeitstempel der Erstellung des Kurses. Meistens wird ein Kurs durch den Administrator des LMS erstellt. Das Element *startdate* ist das Datum, an dem der Kurs anfängt. Die Elemente *enrolstart* und *enrolend* sind die Daten, ab wann und bis wann Studierende sich im Kurs anmelden dürfen. Das Element *lastmodified* ist der Zeitstempel, wann der Kurs zuletzt durch Administratoren oder Dozenten verändert wurde.

In der Tabelle **resource** finden wir das Lernmaterial, das Dozenten zur Verfügung stellen. Das Element *type* ist der Typ des Lernmaterials und kann Werte wie „datei“, „url“, „picture“ usw. annehmen – je nachdem, was das LMS für Materialien enthält. Das Element *timecreated* ist der Zeitstempel, der angibt wann dieses Material im LMS erstellt wurde. Das Element *timemodified* gibt an, wann dieses Material zuletzt verändert wurde.

Die Tabelle **quiz** stellt eine beliebige Aufgabe dar. Das Element **qtyp** gibt dabei die genaue Art des Quiz an. Es kann Werte wie „Aufgabe“, „Test“, „Übung“, „SCORM“ usw. annehmen – je nach Art der Quiz, die das LMS zu Verfügung stellt. Was den Identifikator der Tabelle angeht, haben wir als Ausnahme einen doppelten Schlüssel. Die Verkettung der Elemente *qid* und *qtyp* bildet den Identifikator eines Quiz. So können Elemente, die in der LMS Datenbank in einzelnen Tabellen stehen, als Quiz zusammengefasst werden, ohne dass die Identifikatoren kollidieren. Ein Quiz kann eine oder mehrere Fragen enthalten, wie im Abschnitt zur Tabelle **question** noch erklärt wird. Das Element *title* ist der vom Dozent vergebene Titel. In den Elementen *timeopen* und *timeclosed* ist der Zeitrahmen festgelegt, in dem Studierende das Quiz beantworten dürfen. Die Elemente *timecreated* und *timemodified* sind analog, denen in der Tabelle **resource**.

---

[2] Es gibt einen Fehler in Moodle-Versionen vor 1.9.X durch den dieses Feld leer sein kann, siehe: http://bugs.moodle.org/browse/MDL-16897.

[3] Es gibt ein Problem in Moodle mit diesem Feld, siehe: http://bugs.moodle.com/browse/MDL-18426.

Die Tabelle **question** enthält die Fragen zu dem Quiz. Das Element *typ* ist die Art der Frage, wie „multiple-choice“, „true-false“ oder „shortanswer“ usw.. Dies ist davon abhängig, welche Fragentypen es im LMS gibt. Das Element *text* enthält den Aufgabentext. Die Elemente *title*, *timecreated* und *timemodified* entsprechen denen in der Tabelle **quiz**.

Die Tabellen **forum** und **wiki** enthalten weitgehend die gleichen Elemente wie die Tabelle **resource.**

Die Tabellen, in denen die Interaktionen mit einem Objekt des LMS dargestellt werden, werden mit dem Namen des Objekts und einem angehängten _log bezeichnet. Alle Tabellen, die gerade vorgestellt wurden, haben eine entsprechende _log Tabelle, außer den Tabellen **user** und **group**. Alle _log Tabellen enthalten folgende Elemente:

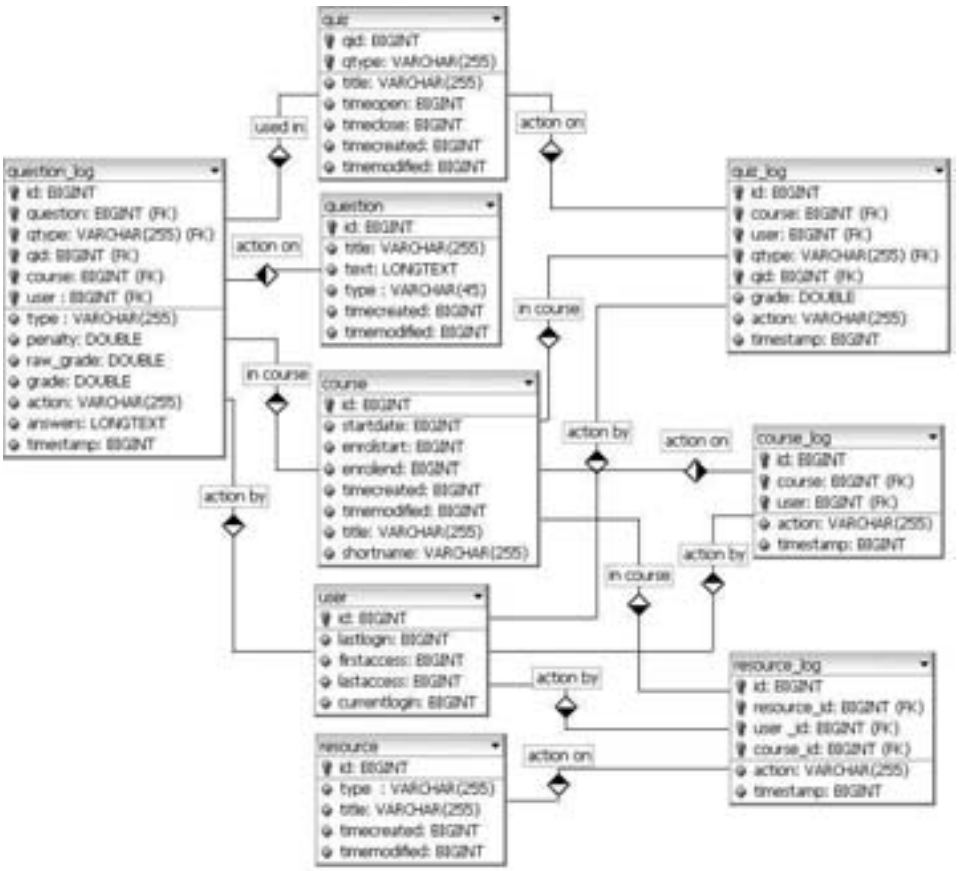

Abbildung 1: Vereinfachtes Schema des Datenmodells

Das Element *user* ist die user-id des Benutzers, der interagiert hat. Das Element *course* ist der Identifikator des Kurses, in dem interagiert wurde. Die Elemente *qid* und *qtype* in der **quiz_log** Tabelle bilden den Identifikator des betroffenen Quiz. Das Element *resource* in der **resource_log** Tabelle ist die resource-id des betroffenen Materials, usw.. Das Element *timestamp* ist der Zeitstempel, zu dem die Interaktion ausgeführt wurde. Das Element *action* gibt die Art der Interaktion an. Dieses Element kann Werte wie „view“, „modify“, „attempt“, „submit“ usw. annehmen, wobei die zwei Werte „attempt“ und „submit“ Interaktionen mit Quiz betreffen.

Die Tabelle **quiz_log** hat außerdem ein *grade* Element, das die Benotung enthält. Die Tabelle **question_log** hat drei weitere Elemente: *penalty*, *raw_grade* und *grade*. Das Element **grade** ist die Benotung für diese Frage in dieser Interaktion. Diese Benotung ist unterschiedlich zur Benotung des Elementes *raw_grade*, wenn es erlaubt ist, eine Frage mehrmals zu beantworten und wenn Abzugspunkte vorgesehen sind. Eine falsche Antwort kann mit Strafpunkten belegt werden, welche im Element *penalty* zu finden sind. Bei einem erneuten Versuch wird durch eine richtige Antwort eine Benotung vorgenommen. Das Element *raw_grade* enthält die Benotung ohne Abzüge. Das Berücksichtigen der Abzugspunkte ergibt den endgültigen Wert der Benotung des Elementes und wird in *grade* gespeichert.

Die Verknüpfung zweier LMS Objekte wird in einer Assoziations-Tabelle festgehalten. Der Name dieser Tabelle wird aus der Verkettung der Namen der LMS Objekte gebildet. Also gibt die Tabelle **course_user** alle Benutzer an, die sich in einem bestimmten Kurs angemeldet haben. Oder die Tabelle **quiz_question** gibt an, welche Fragen in einem bestimmten Quiz verwendet werden. Es ist möglich, dass eine Frage mehreren Quiz zugeordnet ist. Ähnlich gibt die Tabelle **course_quiz** alle Quiz an, die in einem bestimmten Kurs enthalten sind. Es ist möglich, dass ein Quiz in mehreren Kursen verwendet wird. Alle Assoziations-Tabellen beinhalten die zwei Identifikatoren der zwei Objekte. Für die Mehrheit der Assoziations-Tabellen gibt es keine weiteren Elemente. Eine Ausnahme ist die Tabelle **course_user**. Sie enthält drei weitere Elemente *enrolstart*, *enrolend* und *role*. Das Element *enrolstart* ist das Datum, an dem sich der Benutzer im Kurs angemeldet hat, und *enrolend* ist das Datum, zu dem er sich abgemeldet hat. Das Element *role* ist die Rolle des Nutzers. Das Element *role* kann beispielsweise Werte wie „Studierende“, „Dozent“, „Administrator“ usw. annehmen.

3 Systemarchitektur des Exports

Abbildung 2 zeigt das wichtigste Paket der Exportanwendung. Die Klasse `ExtractAndMap` ist die Hauptklasse der Anwendung. Durch diese werden die im LMS gespeicherten Daten extrahiert und auf die Tabellen des Datenmodells abgebildet. Zurzeit gibt es die abstrakte Klasse `ExtractAndMap` und eine Implementierung dieser Klasse für das LMS Moodle.

Diese abstrakte Klasse enthält eine Methode `start`, die den Extraktionsprozess anstößt und die Methode `getMiningInitial`, die initial benötigte Daten von der Analyse-Datenbank holt. Die `start` Methode kann mit einem Startparameter gesteuert werden. Dieser legt fest, ab welchem Zeitpunkt die Daten aus der LMS Datenbank gelesen werden sollen. Außerdem sind in der Klasse die Definition der abstrakten Methoden `getLMStables` und `clearLMStables` zu finden. Diese sind für den Zugriff auf die LMS Datenbank nötig. Zuletzt gibt es hier noch mehrere abstrakte `generate` Methoden. Diese haben wir in der Abbildung 2 durch eine Platzhaltemethode namens `generateTablename_Mining` dargestellt, welche für die Methoden `generate Course_mining, generateQuiz_mining, generateQuiz_log_mining` usw. steht. Jede `generate` Methode benutzt die extrahierten Daten, um eine Tabelle der Analyse-Datenbank zu generieren. Diese werden dann von der `saveMining Tables` Methode in die Datenbank geschrieben.

Um die Daten eines bestimmten LMS in unser Datenmodell exportieren zu können, reicht es, diese Klasse zu erweitern und die abstrakten Methoden entsprechend der Eigenschaften des LMS zu implementieren. Alle Methoden, die benötigt werden, sind in der abstrakten Klasse `ExtractAndMap` als abstrakte Methoden definiert und dokumentiert. Unsere Anwendung ist in Java geschrieben. Sie benutzt die Datenbank MySQL [My10] und das Persistenz-Framework Hibernate [Hy10].

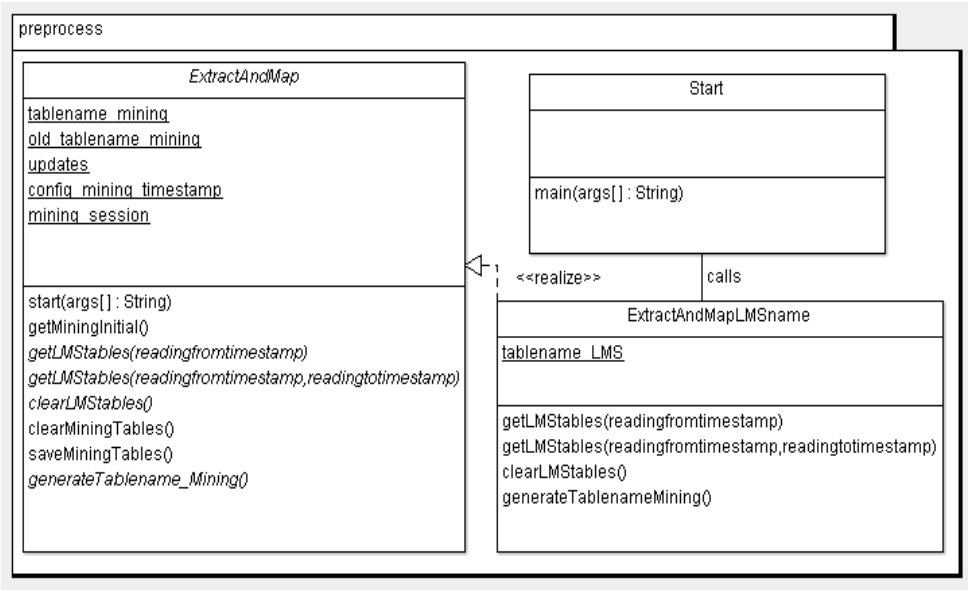

Abbildung 2: Systemarchitektur

3.1 Moodle Implementierung

Der Hauptteil der Anpassung des Tools an ein neues LMS liegt im Schreiben der `generate` Methoden. In diesen findet das Mapping der LMS-Daten auf die Tabellen der Analyse-Datenbank statt. Dabei wird die Struktur der Daten so verändert, dass sie sich besser für Analysen eignet.

Ein gutes Beispiel für eine solche Umgestaltung ist die Moodle Log Tabelle, in der alle Interaktionen der Nutzer mit allen Objekten in Moodle gespeichert werden. Dies führt dazu, dass die Tabelle ständig wächst und meist zu groß für eine effektive Analyse wird. Davon abgesehen wird die Tabelle wegen des großen Datenaufkommes meist regelmäßig gelöscht, was den Verlust von Rohdaten bedeutet, die für Analysen interessant gewesen wären. Wir begegnen dem Problem, indem wir die Interaktionen der Nutzer mit verschiedenen Objekten des LMS in verschiedene Tabellen schreiben. Das Tool verteilt die Daten aus der Log-Datei in mehrere Tabellen, wodurch die Analyse wesentlich vereinfacht wird.

Ein anderer Aspekt der Moodle-Implementierung ist das Zusammenlegen von Tabellen. In Moodle sind *assignment* (Aufgaben) und *quiz* (Tests) verschiedene Objekte. Dies ist jedoch nicht in allen LMS so. Da wir das Tool jedoch möglichst offen für verschiedene LMS halten wollen, vereinigen wir die *assignment* und *quiz* Tabellen von Moodle.

4 Fallbeispiel

Wir haben unsere Export-Anwendung benutzt, um den Kurs „Formale Grundlagen der Informatik“ im Wintersemester 2009/10 zu analysieren. Dieser Kurs wird im ersten Semester des Studiengangs Medieninformatik Bachelor im Präsenzstudium angeboten. Im virtuellen Lernraum Moodle werden Materialien und Aufgaben für die Studierenden bereitgestellt. Die Benutzerdaten dieses Kurses wurden schon mit aufwendigem Preprocessing im WS07/08 analysiert [MY08]. Ziel der aktuellen Analyse ist es zu erproben, inwieweit sich die damalige Analyse komfortabler wiederholen lässt und zu prüfen, ob sich die gleichen Trends, die im WS07/08 zur Nutzung des Lernmaterials beobachtet wurden, bestätigen lassen.

4.1 Kontext des Kurses

Der Kurs „Formale Grundlagen der Informatik“ findet jede Woche in der Form eines vierstündigen seminaristischen Unterrichts statt. Um dieses Modul zu bestehen, schreiben die Studierenden die Klausur in zwei Teilen. Der erste Teil wird Mitte des Semesters geschrieben und der zweite Teil am Ende. Die jetzige Analyse betrifft wie die vorherige [MY08] die erste Hälfte des Semesters.

Die folgenden Lernmaterialien werden am Anfang des Semesters im Lernraum System Moodle zur Verfügung gestellt:

- *Folien1*, *Folien2*, usw. bis *Folien8*: Folien, die Kursinhalte und ungelöste Übungen beinhalten.
- *Buch*: Link zur Webseite des Buches [IAT10], das für diesen Kurs benutzt wird.
- *DP*: Skript „Entwurfmuster für Automaten“ [Me08].
- *JFLAP*: Link zur Software JFLAP oder zur Website der Software [Jf10].
- *Aufgabe1*, *Aufgabe2*, usw. bis *Aufgabe8*: acht Aufgaben für die Selbstevaluierung. Abgabetermin der Aufgabe 1 ist in der ersten Woche des Kurses, Abgabetermin der Aufgabe 2 ist in der zweiten Woche vom Kurs, usw., Abgabetermin der Aufgabe 8 ist eine Woche vor der ersten Teilklausur. Die Punkte dieser Aufgaben zählen für die Klausur nicht.
- *Klausur1*, *Klausur2*, *Klausur3*: 3 Probeklausuren.

Die Registrierung in Moodle und Benutzung der Lernmaterialien ist freiwillig. Für Dozenten ist es interessant zu wissen, ob das Material überhaupt angeschaut wird und ob es zwischen der Benutzung des Materials in Moodle und den Ergebnissen der Klausur einen Zusammenhang gibt. Der Zweck der Analyse der Benutzerdaten ist es solche Informationen zu gewinnen.

4.2 Datenanalyse

Im WS09/10 haben sich 57 Studierende in Moodle registriert. Wird das Lernmaterial benutzt? Dies lässt sich nun mit einfachen Abfragen leicht beantworten. Wegen des aufwendigen manuellen Preprocessings wurde diese Frage für die Folien im WS07/08 nicht beantwortet. Abbildung 3 zeigt die Anzahl der Studierenden, die auf die Ressourcen zugegriffen haben.

Wurden die Selbstevaluierungsaufgaben wahrgenommen? Abbildung 4, Zeile 2 zeigt wie viele Studierende auf die Aufgaben zugegriffen haben. Die Spalte *Mind.1* liefert die Anzahl der Studierenden, die mindestens auf eine Aufgabe zugegriffen haben, die Spalte *Alle* zeigt die Anzahl der Studierenden, die auf alle Aufgaben zugegriffen haben. Die dritte Reihe zeigt ähnliche Zahlen für die Anzahl der Studierenden, die versucht haben, die Aufgaben zu lösen. Alle Zahlen lassen sich bequem gewinnen, bis auf die Zahlen in der letzte Spalte. Diese Abfrage ist aufwendig zu schreiben, aber nicht grundsätzlich schwierig. Sie wurde wegen des Aufwandes im WS07/08 nicht behandelt.

Von den 57 angemeldeten Studierenden haben 48 die erste Teilklausur geschrieben. Von diesen 48 haben neun Studierende weniger als 25 Punkte von 50 erreicht. Wie sehen die Ergebnisse aus? Lassen sich die Studierenden in Gruppen einteilen, je nachdem welche Ressourcen sie benutzt haben? Abbildung 5 gibt einen Überblick der Ergebnisse. Die Zeile *Allgemein* gibt das Minimum, Maximum, den Durchschnitt und die Standardabweichung für die Punkte der ersten Teilklausur für alle Teilnehmer an. Die Zeile *Buch* gibt diese Zahlen nur für Studierende an, die auf die Webseite des Buches zugegriffen haben. Die anderen Zeilen sind ähnlich. Die Zeile *Keine Klausur* betrifft Studierende, die auf überhaupt keine Probeklausur zugegriffen haben und die Zeile *Mind. 1* betrifft Studierende, die versucht haben mindestens eine Aufgabe zu lösen.

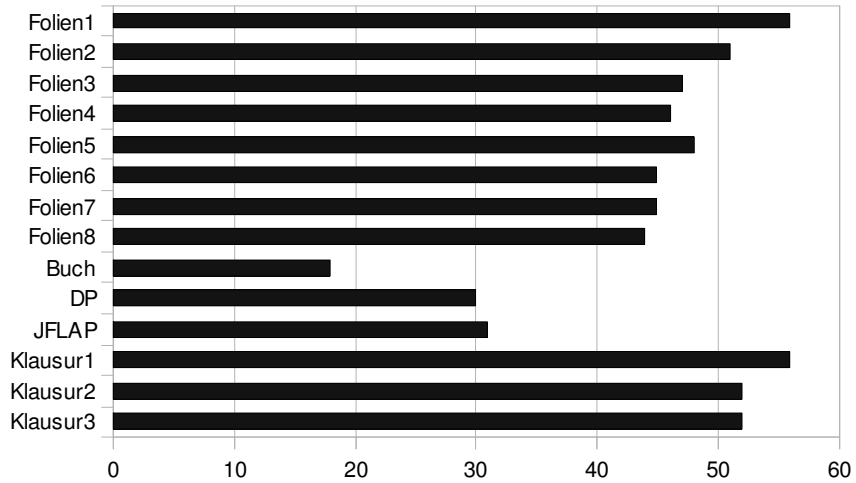

Abbildung 3: Zugriff auf Lernmaterialien

Auf.1	*Auf.2*	*Auf.3*	*Auf.4*	*Auf.5*	*Auf.6*	*Auf.7*	*Auf.8*	*Mind.1*	*Alle*
51	41	35	30	28	29	27	21	51	9
35	27	26	18	19	20	20	12	46	0

Abbildung 4: Aufgaben: Zugriff und Versuch

Im WS07/08 waren 81 Studierende in Moodle registriert, 60 hatten die erste Teilklausur geschrieben und acht Studierende hatten weniger als 25 Punkte erreicht. Der allgemeine Durchschnitt lag bei 36,45. Proportional gerechnet, haben mehr Studierende im WS09/10 die Klausur geschrieben und die Lernressourcen wurden mehr benutzt.

Dennoch zeigt der Zugriff auf die Ressourcen im WS09/10 das gleiche Muster wie im WS07/08: es wird mehr auf *JFLAP* und *DP* zugegriffen als auf die Webseite vom *Buch*. Die Studierenden, die keine Aufgabe gelöst haben, erreichten in der Klausur weniger Punkte. Je weiter das Semester fortschreitet, desto weniger greifen die Studierenden auf Aufgaben zu, und desto weniger versuchen sie sie zu lösen. Wir haben untersucht, ob sich eine treue Gruppe von Studierenden allmählich im Semester bildet, die systematisch die Aufgaben für Selbstevaluierung löst. Die Antwort ist leicht positiv, zeigt aber auch eine gewisse Fluktuation.

Ressource	*Minimum*	*Maximum*	*Durchschnitt*	*S. Abweichung*
Allgemein	9	50	34,4	10,9
Buch	14	50	37	11,94
DP	9	50	34,1	11,7
JFLAP	9	50	35,25	11,11
Mind. 1 Aufgabe	9	50	35,45	10,74
Keine Aufgabe	17	40	26,33	7,27

Abbildung 5: Überblick der Ergebnisse in der ersten Teilklausur

5 Fazit und Ausblick

In diesem Beitrag haben wir ein Datenmodell vorgeschlagen, das in LMS gespeicherte Daten einheitlich abbildet, damit die Datenanalyse durch geringeren Aufwand beim Preprocessing erleichtert werden kann. Wir haben eine Systemarchitektur für die Export-Anwendung entworfen und eine Implementierung für das LMS Moodle erstellt. Ferner haben wir unsere Anwendung benutzt, um die Daten des Kurses „Formale Grundlagen der Informatik“ vom WS09/10 zu analysieren. Die Daten des gleichen Kurses wurden schon im WS07/08 analysiert [MY08]. Die jetzige Analyse mit dem konzipierten Tool ist wesentlich leichter zu handhaben. Es konnten Fragen beantwortet werden, die in der ersten Analyse [MY08] wegen des aufwendigen Preprocessings nicht behandelt wurden. Die aktuelle Analyse zeigt die gleichen Trends in der Benutzung des Materials wie im WS07/08, allerdings ist eine stärkere Beteiligung der Studierenden zu beobachten.

Wie in [BY09] erwähnt, nimmt die Anzahl der Arbeiten zu, die in LMS gespeicherten Daten zu analysieren. Wir hoffen, dass unsere Arbeit helfen wird, weitere Ergebnisse zu erzielen und bewährte Verfahren auf diesem Gebiet zu fördern.

Bis jetzt haben wir unser Datenmodell größtenteils dazu verwendet die Nutzung von Ressourcen und Aufgaben und deren Auswirkung auf die Klausurnote zu analysieren [KMW10b]. Prinzipiell sind aber auch viele weitere Anwendungen möglich, wie zum Beispiel Analysen zur Navigation innerhalb des LMS, Beteiligung in kooperativer Zusammenarbeit oder Klassifizierung von Aufgaben und Ressourcen nach deren Verwendung und Nutzen. Die für solche Analysen notwendigen Informationen liegen im Datenmodell vor.

Wir setzen unsere Arbeit in mehrere Richtungen fort. Eine wichtige Richtung ist die Entwicklung weiterer Fallbeispiele mit LMS-Nutzern, um unser Datenmodell weiter zu erproben und den Katalog von relevanten Fragen zu konsolidieren und zu erweitern. Die besten Techniken, um diese Fragen zu beantworten, sollen ausgearbeitet werden. Eine zweite wichtige Richtung ist die Implementierung eines grafischen Tools für die Auswertung der Ergebnisse. Diese grafische Anwendung soll als Dienst zwischen Nutzer, Datenmodell und Analyse-Tools dienen, um die Analyse zu erleichtern. Ferner wollen wir erforschen, in wieweit unser Datenmodell auf andere Lernsoftware übertragbar ist. Als erstes wollen wir dazu weitere Lernportale untersuchen. Unser Projekt ist Open Source, und wir werden unsere Dokumentationen und den Quellcode zur Verfügung stellen [KMW10a].

Literaturverzeichnis

[BY09] Baker, S.J.D.R.; Yacef, K.: The State of Educational Data Mining in 2009: A review and Future Visions. In: JEDM Journal of Educational Data Mining, 1(1), 2009; S. 3-17.

[HK06] Han, J.; Kamber, M.: Data mining: concepts and techniques. Morgan Kaufman publishers, 2006.

[Hy10] Hibernate relational persistence framework. www.hibernate.org, last access 30.01.2010

[IAT10] Introduction to Automata Theory, Languages, and Computation, http: //infolab.stanford.edu/~ullman/ialc.html, last access 21.01.2010

[IMS10] IMS Question and Test Interoperability Results Reporting, http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_resultv2p1pd2.html, last access 25.04.2010

[Jf10] Jflap: http://www.jflap.org/ last access 21.01.2010

[KMW10a] Krüger, A.; Merceron, A.; Wolf, B.: Nutzerdaten und Nutzerprofile in Lernraumsystemen. http://learn.beuth-hochschule.de/datamining, last access 12.02.2010

[KMW10b] Krüger, A.; Merceron, A.; Wolf, B.: A Data Model to Ease Analysis and Mining of Educational Data. In (Baker, R.; Merceron, A.; Pavlik, P. Hrsg.): Proc Third International Conference on Educational Data Mining EDM2010, Pittsburgh, USA, 2010.

[Me08] Merceron, A.: Unterstützung des Erlernens von endlichen Automaten mit Hilfe von Mustern. In: Proceedings Wokshop für Modellierung in Lehre und Weiterbildung (Desel, J.; Gliz, M. Hrsg.): Modellierung 2008, Berlin, Germany, 2008; S. 27-36.

[MY08] Merceron, A.; Yacef, K.: Interestingness Measures for Association Rules in Educational Data. In (Baker, R.; Barnes, T.; Beck, J. Hrsg.): Proc First International Conference on Educational Data Mining EDM08, Montreal, Canada, 2008; S. 57-66.

[My10] MySQL open source database: http://www.mysql.com/, last access 30.01.2010

[RV07] Romero, C., Ventura, S.: Educational Data Mining: A Survey from 1995 to 2005. Expert Systems with Applications 33, 2007; S. 125-146.

[Vi09] Vialardi Sarcin, C. et al.: Recommendation in Higher Education Using Data Mining Techniques. In (Barnes, T. et al. Hrsg.): Proc Second International Conference on Educational Data Mining, 2009, Cordoba, Spain, 2009; S. 190-199.