

Eine dienste- und komponentenbasierte Architektur zur elektronischen Durchführung von Prüfungen und zum Management von Lehrveranstaltungen

Thorsten Berger, Heinz-Werner Wollersheim

Erziehungswissenschaftliche Fakultät
Universität Leipzig
Karl-Heine-Str. 22b
04229 Leipzig
mail@thorsten-berger.net
wollersh@fakorz.uni-leipzig.de

Abstract: Die Erziehungswissenschaftliche Fakultät setzt erfolgreich eTesting zur Abwicklung von Klausuren unter Examensbedingungen und eine Portalapplikation zur Verwaltung von Lehrveranstaltungen ein. Der Artikel stellt die entwickelten Systeme vor. Nach einer kurzen Darstellung der Ausgangssituation werden vor allem Anforderungen und die Gesamtarchitektur des Systems skizziert. Der letzte Teil gibt eine Auswertung der Erfahrungen beim Einsatz von eTesting wieder.

1 Motivation

Gegenwärtig steht eine erfreulich große Anzahl von eLearning-Plattformen mit unterschiedlichem Funktionsumfang und Komfort zur Verfügung. So hilfreich die Möglichkeiten sind, Lernmaterialien ins Netz zu stellen und nutzerorientierte Lernprozesse durch Diskussionsforen und individualisierte Aufgaben zu stimulieren, so blieb bisher ein Problem ungelöst, das im universitären Alltag insbesondere in Fächern mit einer großen Anzahl von Studierenden zu massiven Belastungen führte: die Durchführung von schriftlichen Klausuren als Kriterium für die Vergabe eines Leistungsnachweises oder als Bestandteil einer akademischen (Zwischen-)Prüfung.

Am Beispiel einer Großvorlesung an der Erziehungswissenschaftlichen Fakultät mit 800 bis 1000 Teilnehmern lässt sich der Korrekturaufwand papierbasierter Klausuren verdeutlichen: Bei freier Aufgabenstellung und handschriftlich frei formulierten Antworttexten wird man, je nach Komplexität und Lesbarkeit, diesen Aufwand mit 0,5 bis 1 Stunde ansetzen müssen. Das ist eine für einen einzelnen Korrektor kaum zu bewältigende Arbeitsmenge. Eine mögliche Strategie zur Verringerung der Korrekturbelastung ist die Verwendung standardisierter schriftlicher Multiple-Choice-Aufgaben, die auch in Papierform eine wesentliche Beschleunigung bewirken. Allerdings fielen die Aufgabenstellungen recht umfangreich aus, so dass immer noch vier Mitarbeiter und 20 studentische Tutoren jeweils sechs bis acht Stunden benötigten, um ihren Stapel Prüfungen

durchzusehen. Das manuelle Zusammenführen und Abgleichen mit Einschreiblisten nahm danach noch ein bis zwei Tage in Anspruch. Außerdem erwiesen sich die papiernen Varianten als sehr unflexibel, überraschend fehleranfällig für Studierende und Korrektoren und inhaltlich qualitativ unzureichend, weil sich nicht jeder Sachverhalt für die Wissenskontrolle über Multiple-Choice eignet.

2 Anforderungen

2.1 Funktionale Anforderungen

Aus unserer Sicht stellten sich die Anforderungen an ein elektronisches Prüfungssystem in zwei großen Aufgabenfeldern dar:

1. Computergestützte Durchführung von Prüfungsklausuren (Prüfungssystem):
 - sichere Identifikation der Klausurteilnehmer
 - Sicherung gegen Betrug und Täuschung
 - Zugriff auf Prüfungsaufgaben nur unter klausurüblichen kontrollierten Bedingungen

2. Organisation und Management von Lehrveranstaltungen (Portalsystem):
 - Lehrveranstaltungen verwalten
 - Online-Einschreibungen, Übungsaufgaben, Foren, Download-Bereiche

Teilnehmer sollten Gelegenheit bekommen, sich von zu Hause auf die elektronischen Klausuren vorzubereiten, d.h. sich mit der Art der Fragestellungen und dem Handling der Oberfläche vertraut zu machen. Dies war eine wichtige Voraussetzung, um anfänglichen Akzeptanzschwierigkeiten entgegenzutreten zu können.

2.2 Nichtfunktionale Anforderungen

Als nichtfunktionale Anforderungen sind in erster Linie Sicherheit und Zuverlässigkeit des Prüfungssystems zu nennen. Das impliziert zum einen die Absicherung des Prüfungsservers gegen unbefugten Zugriff sowie eine Trennung sensibler Prüfungsdaten von (ebenso sensiblen) Einschreibungs- und Studentendaten. Erweiterbarkeit und Integrierbarkeit stellten weitere Anforderungen an das System dar. So sollte es u.a. möglich sein, zusätzliche Aufgabentypen hinzuzufügen und später das Portal mit anderen Hochschulinformationssystemen zu kombinieren.

3 Umsetzung

Es begann die Suche nach Ermöglichung einer computergestützten Klausur mit den genannten Anforderungen. Recherchen ergaben, dass es im Bundesgebiet nur sehr wenig Erfahrung mit eTesting gibt, obwohl durchaus Bedarf, insbesondere in Fakultäten mit hohen Immatrikulationszahlen, besteht.

Als wichtigste kommerzielle Applikationen existieren die Systeme LPLUS¹ Test-Management-System, I-Qbox², WebAssign³ und ATS Online⁴. Auf diese gehen wir hier nicht weiter ein, da sie aus finanziellen Gründen und aufgrund eingeschränkter Flexibilität für uns nicht in Frage kamen. Lösungen im Open-Source-Bereich, wie bspw. Tests in OLAT⁵, waren nicht mächtig und erweiterbar genug, als dass man sie hätte adaptieren können.⁶

Basierend auf den Ergebnissen der Recherche fiel die Entscheidung zugunsten einer Eigenentwicklung. Der Bedarf lässt sich folgendermaßen skizzieren: Benötigt wird ein Prüfungssystem sowie eine Verwaltungs- und Organisationsplattform (Portalsystem). Das Prüfungssystem darf aus Sicherheitsgründen nur zu Prüfungszwecken in Erscheinung treten und lediglich lose mit dem Portal gekoppelt sein. Aufgrund der Anforderung, in beiden Systemen Aufgaben bearbeiten zu können, sollte eine eigenständige Komponente „Aufgabenmodell“ entwickelt werden, welche sowohl in das Portal- und das Prüfungssystem als auch in andere Applikationen integrierbar ist.

Die Arbeiten begannen im November 2004 und führten zur ersten elektronischen Prüfung im darauf folgenden Februar. Das Projekt basiert auf der J2EE-Plattform und ist seit Herbst 2005 unter dem Namen „elatePortal“ als Open Source (GPL-Lizenz) verfügbar⁷. Im Folgenden soll die Architektur grob skizziert werden.

3.1 Aufgabenmodell

Das Aufgabenmodell ist ein Framework, das nach dem „Contract First“-Ansatz zunächst nur aus Interfaces besteht und die Funktionalität seiner Implementierung über eine API zur Verfügung stellt. Genauer betrachtet handelt es sich um eine Container-Lösung, in welcher ein sog. Tasklet-Container den Lifecycle von Tasklets steuert. Tasklets sind in Ausführung befindliche Aufgaben eines bestimmten Aufgabentyps. Einige Typen wurden im Modell vordefiniert, von denen einer der für Prüfungen benutzte Typ „Complex“ ist und wiederum aus verschiedenen Teilaufgabentypen bestehen kann. Das ermöglicht die Erweiterung des Aufgabenmodells um weitere Aufgabentypen auf zwei verschiedenen Ebenen.

¹ <http://www.lplus.de>

² <http://www.i-qbox.com>

³ <http://www.webassign.com>

⁴ <http://www.nccedu.com/ats/AtsOnline.asp>

⁵ <http://www.olat.org>

⁶ so auch die Evaluation von Edutech [Edu05]

⁷ <http://www.elateportal.de>

Für dieses Modell existiert eine Referenz-Implementierung, welche den Aufgabentyp „Complex“ mit den Teilaufgabentypen „Multiple Choice“, „Zuordnung“, „Lückentext“, „Freier Text“ und „Graphische Darstellung“ unterstützt. Die Verwendung des Komponenten-Frameworks Spring [WB05] und von Factories [Ga97] in dieser Implementierung garantieren die Erweiterbarkeit um weitere Aufgabentypen. Es ist weiterhin in nahezu jede J2EE-Applikation integrierbar, da Applikations-spezifische Implementierungen im Modell gekapselt wurden und durch den Spring-Ansatz ohne Änderungen an vorhandenem Code ausgetauscht werden können. Die Integration in den Prüfungsserver und das Portal geschah entsprechend mit relativ geringem Aufwand.

Der Ablauf einer solchen „Complex“-Aufgabe im Prüfungsszenario ist folgender. Der Studierende startet einen neuen Lösungsversuch. Auswahlalgorithmen stellen (randomisiert) aus einem in XML definierten Fragenpool die Klausur zusammen (bei MC-Aufgaben können auch Antwortkombinationen variiert werden). Diese Teilaufgaben sind hierarchisch in Blöcken und Kategorien organisiert und erlauben eine sehr flexible Zusammenstellung der Aufgaben. Die Bearbeitung durch Studierende erfolgt seitenweise und endet mit dem Abgeben oder dem Ablauf der Bearbeitungszeit. Das System versucht anschließend, die Klausur soweit wie möglich vorzukorrigieren. Multiple-Choice- und Zuordnungsaufgaben können vollständig, Lückentexte teilweise automatisch korrigiert werden. Freitext- und Graphikaufgaben werden Korrektoren vorgelegt.

3.2 elatePortal

Das elatePortal ist eine J2EE-Portalapplikation [LM04], die auf dem Open-Source-Portalserver Apache Jetspeed-2¹ basiert und als Plattform zur Verwaltung von Lehrveranstaltungen eingesetzt wird. Auch hier wurde bewusst auf eine Eigenentwicklung gesetzt. Der Fokus lag auf einer sauberen Architektur, die ebenfalls für zukünftige Projekte genutzt werden kann. Einer der Hauptvorteile, die sich durch Portaltechnologie ergeben, ist eine umfangreichere Erweiterbarkeit als bei den meisten derzeit verfügbaren LCMS. JSR168-kompatible Applikationen [AH03] lassen sich ohne Probleme in das Portal integrieren. Des Weiteren bietet Jetspeed-2 von Haus aus eine grundlegende Infrastruktur. Dazu gehören ein feingranulares, rollenbasiertes Rechtemodell, LDAP-Authentifizierung, Administrationstools, Navigationsstrukturen und eine homogene, von Nutzern anpassbare, Oberfläche. Nachteilig wirken sich die höhere Komplexität und der stärkere Overhead auf dem Server im Gegensatz zu normalen Webapplikationen aus.

Das elatePortal verwendet das Komponenten-Framework Spring in seiner Business-Schicht und stellt die Funktionalität über eine API den eigenen Portlets sowie anderen Applikationen zur Verfügung. Eine Architektur-Skizze ist in [Be05] zu finden.

Das Projekt wird kontinuierlich weiterentwickelt. Geplant sind u. a. die Unterstützung des Student-Lifecycle in Bachelor-/Master-Studiengängen und eine CMS-Integration.

¹ <http://portals.apache.org/jetspeed2>

3.3 Prüfungsserver

Ein eigenständiger Prüfungsserver wurde hauptsächlich aus Sicherheitsgründen entwickelt. Seine Installation erfolgt nur zur Prüfung und der Zugriff ist auf das Computerkabinett beschränkt. Seine Daten sind strikt vom Portal getrennt. Nach Ende einer Prüfungsperiode kann er archiviert und wieder gelöscht werden. Der Datenaustausch mit dem Portal erfolgt über eine Web-Service-Schnittstelle und dient in erster Linie zur Authentifikation von Prüfungskandidaten. In Zukunft ist eine genauere Spezifizierung dieser Schnittstelle geplant, um die Austauschbarkeit beider Systeme zu gewährleisten und auch komplexere Überprüfungen (bspw. des genauen Prüfungstermins, zu dem sich der Prüfungskandidat angemeldet hat) durchführen zu können. Diese werden z. Zt. von der Prüfungsaufsicht übernommen.

4 Erfahrungen

Zu Vergleichszwecken mit der Papierklausur lässt sich der Gesamtaufwand in die drei Phasen Vorbereitung, Durchführung und Korrektur einteilen. Die Vorbereitung der Prüfung ist zunächst durch Anlegen eines Aufgabenpools und eine zusätzliche Informationsveranstaltung umfangreicher, jedoch relativiert sich dieser Aufwand stark durch die Wiederverwendbarkeit des Pools. Für die Durchführung der Klausur steht nur ein kleines Computerkabinett mit 30 Rechnern zur Verfügung. Dies ergibt die Notwendigkeit, die gesamte Gruppe der Studierenden in Prüfungsgruppen aufzuteilen und impliziert wiederum einen höheren Zeitaufwand durch viele kleine Termine. Der Vorteil liegt hier in einer Flexibilisierung der Zeit, einem geringeren Organisationsaufwand und einer Verbesserung der Prüfungsqualität, welche durch eine größere Anzahl qualitativ unterschiedlicher Aufgabentypen gegenüber der Papierklausur wesentlich gesteigert werden konnte. Eine drastische Reduzierung ergibt sich bei der Korrektur der Klausuren, da eine Korrektursitzung mit den Tutoren nun nicht mehr länger als zwei Stunden dauert und eine sofortige Auswertung der Ergebnislisten möglich ist.

Weiterhin fällt auf, dass die Computerliterate der aktuellen Studierendengeneration bereits ausreichend hoch ist, so dass keiner mit den Prüfungen Probleme hatte oder sie gar ablehnte. Damit fällt nach vier Semestern Einsatz die Bilanz des Systems unter den genannten Bedingungen uneingeschränkt positiv aus.

Literaturverzeichnis

- [AH03] Abdelnur, S.; Hepper, S.: Java Portlet Specification, 2003.
- [Be05] Berger, T.: Der UebManager und das elatePortal als eTesting-Systeme. <http://www.thorsten-berger.net/fobis/vortrag.pdf> (Juni 2006), 2005; S. 22.
- [Edu05] Edutech: Evaluation of Open Source Learning Management Systems. <http://www.edutech.ch/lms/ev3> (Juli 2006), 2005.
- [Ga97] Gamma, E. et.al.: Design Patterns: Elements of Reusable Object-Oriented Software. 1997.
- [LM04] Linwood, J.; Minter, D.: Building Portals with the Java Portlet API. Apress, 2004.
- [WB05] Walls, C.; Breidenbach, R.: Spring in Action. Manning, 2005.