Online-Übungssystem für die Programmierausbildung zur Einführung in die Informatik

Andreas Hoffmann, Alexander Quast, Roland Wismüller

Fachgruppe Betriebssysteme und verteilte Systeme,
Fachbereich 12 – Elektrotechnik und Informatik, Universität Siegen
Hölderlinstraße 3, 57068 Siegen
{andreas.hoffmann, roland.wismueller}@uni-siegen.de
alexander.quast@student.uni-siegen.de

Abstract: Online Programmierausbildung stellt mittlerweile einen wichtigen Teil des E-Learnings dar, insbesondere an Hochschulen. Es existieren viele verschiedene Systeme auf dem Markt, die sich speziell der online Programmierausbildung und den damit verbundenen Schwierigkeiten auf verschiede Weisen nähern. Das Projekt DUESIE (Das UEbungsSystem der Informatik Einführung) ermöglicht die komplette Abwicklung des Übungsbetriebes zur Veranstaltung „Einführung in die Informatik“ an der Universität Siegen. DUESIE unterstützt u.a. die automatische Auswertung und Bewertung von Programmieraufgaben in JAVA und SML sowie UML.

1 Einleitung

In der Informatikausbildung an Hochschulen ist es üblich, dass die Lehrveranstaltungen mit Übungsaufgaben gekoppelt sind. Das heißt, der Lehrstoff wird durch zusätzliche Übungen vertieft und veranschaulicht. Diese Übungen bestehen klassischer Weise aus verschiedenen Aufgabentypen, wie sie in einer Klausur ebenfalls auftauchen könnten. Dabei handelt es sich um verschiedene Arten von Textaufgaben die beantwortet werden müssen, sowie Multiple- und Single-Choice Aufgaben die durch einfaches Ankreuzen gelöst werden. Einen weiteren, aber weitaus wichtigeren Teil stellen die Programmieraufgaben dar. Diese nehmen eine zentrale Rolle in der Lehre/Ausbildung ein.

Zusätzlich ist das Aufzeigen von Fehlern und Schwachstellen im Programmcode durch die Tutoren von entscheidender Bedeutung. Jedoch ist das Verhältnis von Anzahl Tutoren/Korrektoren zur Anzahl der Studierenden oft so unausgewogen, dass eine genaue Betrachtung und Korrektur von Programmieraufgaben schlecht bis gar nicht mehr möglich ist. Daher ist eine zumindest teilweise automatische Korrektur und Kontrolle von Übungsaufgaben, gleich welcher Art, wünschenswert. An vielen Hochschulen werden bereits Systeme eingesetzt, die online Text und Multiple-Choice Aufgaben auswerten

können. Die meisten klassischen Systeme (z.B. Moodle[1]) sind aber für die Auswertung von Programmieraufgaben ungeeignet.

Daher wurden an verschiedenen Hochschulen neue Systeme entwickelt, die nun auch die Auswertung von Programmieraufgaben übernehmen ([KSZ02], [MOS07], [BEW04], [VHL01]). An der Universität Siegen wird im Rahmen einer Projektgruppenarbeit ein Onlinesystem entwickelt, das sich speziell den Anforderungen an ein Übungssystem für die Einführung in die Informatik widmet. Dieses System mit den Namen DUESIE (Das UEbungsSystem der Informatik Einführung) bietet die Möglichkeit den kompletten Übungsbetrieb online abzuwickeln. Dies reicht vom Erstellen eines Übungsblattes bis hin zur teilautomatischen Auswertung und Korrektur von Aufgaben. Unterstützt werden Single-Choice-, Multiple-Choice-, Matrix-Choice-, Freitext- und Lückentextaufgaben, sowie die Programmiersprachen SML (SMLNJ) und Java 5. Darüber hinaus ist DUESIE in der Lage, (digitale) UML Diagramme zu analysieren und auszuwerten.

2 Verwandte Arbeiten

Mehr als vielfältig stellt sich die Menge der bereits vorhandenen E-Learning-Plattformen und Systemen zur online Programmierausbildung dar. Ein Vertreter dieser Plattformen ist Moodle, das seit längeren auch an der Universität Siegen eingesetzt wird. Moodle stellt zwar ein mächtiges Lern-Management-System dar, mit dessen Hilfe Inhalte für große und viele verschiedene Gruppen publiziert werden können und Lerninhalte durch die Community erarbeitet werden, doch sind sie nicht für die speziellen Bedürfnisse an ein System zur Programmierausbildung ausgerüstet. Dieser Umstand und der wachsende Bedarf an solchen Systeme hat eine Reihe kleinerer Projekte hervorgebracht, die nachfolgend vorgestellt werden sollen.

2.1 Praktomat

Praktomat wurde an der Universität Passau entwickelt und steht momentan in der Version 4.2 zur Verfügung ([KSZ02], [Ze99]). Es wird zur Praktikumsverwaltung von Programmierveranstaltungen eingesetzt. Praktomat ist eine webbasierte Anwendung die zusätzlich externe Tools von anderen Anbietern einsetzt.

Als Benutzer kann man seine bearbeiteten Aufgaben als unkompilierten Sourcecode per Weboberfläche auf den Server hochladen. Das System übersetzt den Code und unterzieht ihn sofort verschiedenen statischen Tests, die vorher festgelegt wurden. Ein Student kann seinen Code testen lassen und bekommt vom System eine Auswertung zu seiner Lösung angezeigt. Sollte eine gewisse Anzahl der Testfälle nicht funktionieren, oder das Programm gar nicht erst starten, kann er seine Lösung überarbeiten und zu einem späteren Zeitpunkt noch einmal abgeben. Diesen Vorgang kann er beliebig oft wiederholen bis das System seine Lösung akzeptiert. Praktomat unterstützt momentan die Program-

[1] http://moodle.org/

miersprachen Java, C++ sowie Haskell. Die automatisierten Tests die Praktomat durchführt, werden vorher von einem Tutor oder Dozenten im System angelegt und definiert. Zur Durchführung dieser Tests benutzt das System das Tool DejaGnu[2], welches interaktive Benutzereingaben simuliert.

Eine interessante Funktionalität des Systems ist die Kommentar-Funktion. Studierende haben die Möglichkeit, Lösungen anderer Kommilitonen einzusehen und zu kommentieren. Darüber hinaus bekommt jeder Studierende eine eigene, individuelle Aufgabe zugeteilt, um einfaches Abschreiben und Copy-Paste zu vermeiden.

2.2 ASB

Das ASB (Automatische Software-Bewertung) System wurde im Jahr 2006 an der Fachhochschule Trier innerhalb des Fachbereichs Informatik entwickelt [MOS07]. Das ASB System ist eine Webanwendung, die es Dozenten und Tutoren ermöglicht, Aufgaben zu ihren Vorlesungen zu erstellen, zu denen Studierende innerhalb eines festen Zeitraums Lösungen einreichen können. Bei ASB wird der Ansatz verfolgt, Studierenden in dem vorgegeben Zeitrahmen so oft Lösungen abgeben zu lassen, wie sie möchten. Die letzte abgegeben Lösung wird dabei immer durch die neue überschrieben. Zusätzlich haben die Studierenden auch hier die Möglichkeit, in einem mehrstufigen Peer-Review-Prozess, die Lösungen anderer Studierender zu bewerten.

ASB stellt in erster Linie ein Framework dar. Es übernimmt die Verwaltung der Aufgaben und Lösungen und führt die von den Tutoren oder Dozenten festgelegten Bewertungsmaßnahmen durch. Ein umfangreiches Zuordnungssystem stellt die Widerverwertbarkeit einzelner Aufgaben, Bewertungsschemata und Tests sicher. So werden z.B. Maßnahmen zur Überprüfung von Code auf richtige Formatierung nicht einer speziellen Aufgabe zugeordnet, sondern einer Aufgabensammlung. Auch werden Abgabezeiträume nicht direkt einer Aufgabe zugeordnet, sondern einem sog. Bewertungsprozess. Die Trennung von eigentlicher Aufgabe und Bewertungsschema stellt eine Widerverwertbarkeit der Aufgaben sicher. Es entsteht ein Archiv an Aufgaben, die sich auch später noch einsetzten lassen. Würde man z.B. den Abgabezeitraum fest in die Aufgabe integrieren, wäre ein erneutes Verwenden der Aufgabe kaum möglich, da das System den Zeitraum als abgelaufen ansehen würde.

Um die Bewertungsmaßnahmen durchzuführen, wird die studentische Software auf dem ASB Server ausgeführt. Dabei besteht die Gefahr, dass entweder absichtlich oder unabsichtlich, bösartige Software eingereicht wird. Daher werden die Programme auf dem Server mit eingeschränkten Rechten ausgeführt.

ASB unterstützt momentan standardmäßig drei verschiedene Hilfsprogramme die bereits integriert sind. Das sind der Java Compiler von IBM, CheckStyle[3] und FindBugs[4]. Der

[2] http://www.gnu.org/software/dejagnu/
[3] http://checkstyle.sourceforge.net
[4] http://findbugs.sourceforge.net

Java Compiler von IBM unterstützt einige hilfreiche Tests, die ihm den Vorzug gegenüber dem Sun Java Compiler gaben. Es werden Variablen- und Attributüberdeckungen gefunden, oder fälschliche Zugriffe auf statische Methoden. CheckStyle übernimmt auch hier, wie beim Praktomat, die Kontrolle von Formatierungskonventionen und Dokumentationsrichtlinien. Das Tool FindBugs untersucht Bytecode auf bekannte und häufige Fehlermuster. So werden z.B. Endlosschleifen durch fehlende Abbruchbedingungen erkannt.

Doch nicht nur statische Bewertungsverfahren wie zuvor beschrieben kommen zum Einsatz. Mit Hilfe des Testing-Frameworks JUnit[5] werden zusätzliche aufgabenspezifische Tests durchgeführt. Zum Überprüfen von grafischen Benuzeroberflächen wird die JUnit Erweiterug Abbot[6] eingesetzt. Abbot bietet Funktionen zum Auffinden von Komponenten grafischer Benutzeroberflächen sowie zur Bedienung der zu testenden Oberfläche an.

2.3 XLX

XLX steht für „eXtreme e-Learning eXperience“ und wurde von der Universität Münster ins Leben gerufen ([VHL01], [SVW06]). Das Projekt startete im Jahr 2001 und wird bis heute weiterentwickelt. XLX ist eine Webanwendung die den universitären Übungsbetrieb von Informatikfächern unterstützen soll und ursprünglich für den Einsatz in Datenbank-Vorlesungen konzipiert. Es konnten SQL Statements und Multiple-Choice Aufgaben online bearbeitet und korrigiert werden. Wie schon im Namen ersichtlich, wurde hier ein Schwerpunkt auf Gesichtspunkte des E-Learnings gelegt. Das System soll nicht etwa eine ohnehin vorhandene Präsenzübung unterstützen, sondern als komplette Abwicklungsplattform für den Übungsbetrieb dienen. Tutoren kommunizieren über das System mit den Studierenden und geben ihnen Ratschläge und Berichtigungen zu ihrer Arbeit. Die XLX Plattform passt sich dem individuellen Lernfortschritt und Tempo an. Dies geschieht dadurch, dass neue Übungsaufgaben erst freigeschaltet werden, nachdem die vorhergegangene Aufgabe erfolgreich bearbeitet wurde. Die übliche Vorgehensweise, bei der Übungsaufgaben innerhalb eines festen Zeitfensters (z.B. 1 Woche) abgeliefert werden müssen, wird hier bewusst aufgegeben um dem Studierenden ein freies Zeitmanagement zu ermöglichen.

Die Auswertung von Java Aufgaben basiert auf dem Java-Build-Tool ANT[7] der Apache Foundation. Der Studierende lädt seine Lösung über ein einfaches Webformular auf den Server hoch. Dort wird ein ANT Prozess angestoßen, der vorher definierte, sog. Tasks, startet. Diese Tasks bieten vielfältige Möglichkeiten den Code zu kompilieren und anschließend zu testen.

XLX besitzt eine dreischichtige Client-Server Architektur. Während Benutzerdaten der Studierenden, sowie Aufgaben und Lösungsabgaben in einer „kleinen“ MySQL Datenbank mit auf dem Web-Server verwaltet werden, benutzt die von Servlets gesteuerte Lo-

[5] http://www.junit.org
[6] http://abbot.sourceforge.net/
[7] http://ant.apache.org

gik für die SQL Übungsaufgaben eine IBM-Datenbank, die auf einem externen Datenbankserver liegt.

3. Das Projekt DUESIE

Im Wintersemester 2007/08 haben im Rahmen einer Projektgruppenarbeit im Fachbereich Informatik und Elektrotechnik der Universität Siegen die Arbeiten an einem eigenen Online-Übungssystem begonnen. Durch die Einführung der Bachelor- und Master-Studiengänge in der Informatik an der Universität Siegen, wurde nach einer maschinell unterstützten und zum Teil automatisierten Softwarelösung zur Abwicklung des Übungsbetriebs in der Informatik gesucht. Die „Einführung in die Informatik“ soll grundlegende Konzepte der Informatik vermitteln, wobei der Schwerpunkt auf der objektorientierten Programmierung mit Java, sowie objektorientierter Modellierung mit UML liegt. Ein weiterer Schwerpunkt liegt auf der funktionalen Programmierung mit SML.

Bisher wurden diese Vorlesungen von einer Übung begleitet bei der die Studierenden ihre Lösungen zu gestellten Übungsaufgaben in Papierform einreichten, bzw. freiwillig vorführten. Dabei waren nur für Lehramtsstudenten die Übungen verpflichtend, alle anderen Studierenden besuchten die Übungen freiwillig. Um nun alle Studierenden zu motivieren die Übungen durchzuführen, ist die Abgabe für alle verpflichtend und Voraussetzung zur Prüfungszulassung.

Bei einer durchschnittlichen Teilnehmerzahl von ca. 100 Studierenden wird klar, welch enormer personeller Aufwand betrieben werden musste, um diese „Papier-Lösungen“ zu korrigieren und zu sichten. Während Textaufgaben noch relativ leicht von Tutoren bewertet werden können, verhält es sich bei Programmieraufgaben ungleich schwerer. Größere Programme können nur noch stichprobenartig getestet werden, oft auch nur auf ihre Lauffähigkeit. Programmierstil und Effizienz hingegen nur unzureichend.

Daher wurde das Projekt DUESIE ins Leben gerufen, um den Übungsbetrieb größtenteils online abzuwickeln. DUESIE sollte nicht etwa die Präsenzübung ersetzen, sondern eine automatisierte Korrektur der Übungen und damit eine Entlastung der Tutoren erzielen. Die dadurch gewonnene Zeit soll wiederum in besseres Feedback für die Studierenden investiert werden.

3.1 Konzeption

DUESIE ist, wie auch die meisten anderen Systeme zur online Programmierausbildung, eine reine Webanwendung und komplett serverseitig implementiert. Es wurde Wert darauf gelegt, plattformunabhängig zu sein und keine zu speziellen Anforderungen an den Client zu stellen. Um DUESIE zu benutzen, braucht der Studierende lediglich einen internetfähigen Rechner mit Webbrowser. Nur für die Bearbeitung von UML Aufgaben wird eine installierte Java-Runtime-Umgebung benötigt. Zudem wurde Wert auf Barrierefreiheit gelegt. Es gibt eine implementierte Sprachausgabe für spezielle Captcha-Grafiken sowie Unterstützung für Menschen mit Sehbehinderungen.

Um DUESIE nutzen zu können, muss der Studierende sich lediglich einmal im System registrieren. Dies geschieht über seinen Account beim ZIMT[8] der Universität Siegen. DUESIE greift auf diese Anmeldedaten zurück, es wird kein neuer Login benötigt. Nachdem der User die einmalige Registrierung erledigt hat, bekommt er Zugriff auf seinen persönlichen Bereich im System.

DUESIE verfügt über ein komplettes User/Rechte Management, welches zwischen Studierenden, Tutoren, Dozenten und Administratoren unterscheidet (vgl. [BEW04], [SVW06]). Die Benutzeroberfläche enthält je nach User mehr oder weniger Menüpunkte, unterscheidet sich aber nicht in Design oder Erscheinung. Es wurde darauf geachtet, die Oberfläche einheitlich und übersichtlich zu gestalten. Die Navigation durch das System ähnelt anderen bekannten Webprojekten, um auch unerfahrenen Benutzern den Umgang mit dem System zu erleichtern.

Übungsaufgaben können von Dozenten einfach und direkt im System angelegt werden. Um dies optisch optimal umzusetzen, ist ein WYSIWYG Editor[9] integriert, der eine komplette Textverarbeitung (ähnlich MS-Word) ersetzt. So können auch Formeln und Sonderzeichen, sowie verschiedene Schriftarten verwendet werden (insb. Monospaced-Fonts für Sourcecode). Übungsblätter (Zusammenstellungen von Fragen) können aus einem Pool von bereits vorher angelegten und neuen Fragen zusammengestellt werden. Zusätzlich wurde eine Exportfunktion für PDF Dateien integriert. Übungsblätter können so auch zur Bildschirmansicht oder zum Ausdruck aufbereitet werden.

Dozenten können Übungsblätter, bestehend aus den Aufgabentypen Freitext, Single-Choice, Multiple-Choice, Matrix-Choice, Lückentext, UML-Modellierungen, Java- und SML- Programmieraufgaben erstellen.

Wenn ein Dozent ein Übungsblatt zur Bearbeitung freigibt, hat der Studierende eine vorgegebene Zeit, um dieses Übungsblatt zu bearbeiten. Wenn der User eine Aufgabe bearbeitet hat, wird diese als „bearbeitet“ markiert. Hierin unterscheidet sich DUESIE von anderen Systemen. DUESIE gibt dem User nicht direkt an, ob seine Lösung richtig ist. Das „Abgeben“ erfolgt immer auf Übungsblattebene und nicht auf Aufgabenebene. Er kann zwar einzelne Aufgaben so oft abändern wie er will, erfährt aber nicht, ob seine neue Lösung besser oder schlechter ist als die alte. Wenn der Studierend so viele Aufgaben gelöst hat wie er kann oder will, kann er sein Übungsblatt abgeben. Danach erfolgt erst die Auswertung aller Aufgaben. Diese Herangehensweise soll verhindern, dass Studenten die Präsenzübungen komplett auslassen und einfach über das „Try-and-Error-Verfahren“ richtige Lösungen erraten, was z. B. bei den Multiple-/Single-Choice Aufgabentypen sehr einfach möglich wäre.

DUESIE ist kein E-Learning-System wie andere Plattformen. Es soll viel mehr die Auswertung und Korrektur von Übungen so weit es geht automatisieren. Die Umstellung

[8] Zentrum für Informations- und Medientechnologie, ehem. Hochschulrechenzentrum der Universität Siegen
[9] Abk. für Editoren mit grafischer Benutzeroberfläche die keine Kenntnisse der verwendeten Sprache (z.B. HTML) voraussetzen. DUESIE verwendet den Editor TinyMCE (http://tinymce.moxiecode.com)

von einem Übungssystem zu einem E-Learning- System, oder deren Integration, ist dennoch für zukünftige Versionen denkbar.

3.2 Umsetzung

Das System DUESIE ist zum größten Teil mit PHP 5 entwickelt worden. Alle Webseiten werden dynamisch erzeugt. Dabei wurde darauf geachtet, Programmlogik von Userinterface zu trennen. Es ist also möglich, über den Einsatz von HTML-Templates und CSS (Cascading Style Sheets) die gesamte Oberfläche getrennt von der Funktionalität auszutauschen. Damit ist die spätere Personalisierung für andere Einsatzzwecke sichergestellt. Um einige Sonderfunktionen zu realisieren, wurde zudem Javascript eingesetzt. Damit lassen sich dynamische Seitenänderungen bewerkstelligen, ohne komplette Seiten vom Server nachladen zu müssen.

Zur Datenhaltung wird eine MySQL Datenbank eingesetzt. Durch den Einsatz einer Datenbankklasse ist aber die Verwendung von anderen Datenbanken möglich.

Wie schon erwähnt werden verschiedene Aufgabentypen unterstützt. Alle Arten von Textaufgaben werden durch eine in PHP geschriebene Logik korrigiert. Beim Bewerten von Freitextaufgaben ist jedoch eine manuelle Nachsicht erforderlich, da Freitextaufgaben lediglich nach bestimmten Schlüsselwörtern durchsucht werden. Semantische und logische Prüfung von Freitextaufgaben ist per Software momentan noch nicht möglich. Um Java Programme, die von Studierenden per Upload eingereicht werden, zu überprüfen, bedient auch DUESIE sich zusätzlicher Technologien.

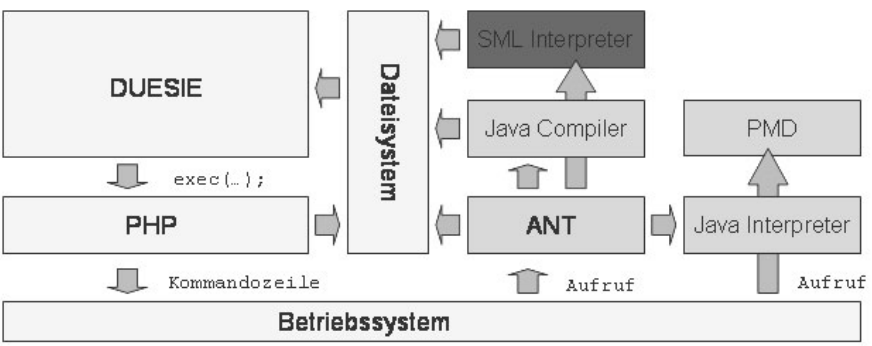

Abbildung 1: Schematische Darstellung der externen Tools im System

Wie in Abb. 1 zu erkennen, benutzt DUESIE mehrere externe Tools, um Programmieraufgaben zu verarbeiten. Am Beispiel einer Java Programmieraufgabe lässt sich der Ablauf erklären. Nachdem ein User seine Lösung auf den Server hochgeladen hat, wird von

PHP über einen Systemaufruf das Java Build-Tool ANT gestartet. ANT ist mit dem Build-Tool MAKE[10] vergleichbar. Es wird bei der Softwareentwicklung eingesetzt um den Kompilierprozess sowie andere Schritte zu automatisieren. ANT arbeitet eine XML Datei ab, in der festgelegt wird, welche Aufgaben ANT erledigen soll. Klassischer Weise steht an erster Stelle das Kompilieren des Sourcecodes. Anschließend werden z.B. die Binärdateien in eigene Verzeichnisse verschoben und Testläufe durchgeführt.

ANT steuert auch bei DUESIE den kompletten Kompilierprozess, führt alle Testläufe durch und erstellt entsprechende Logdateien. Diese Logdateien sind Basis für die Bewertung einer Aufgabe. Die Testdurchläufe werden beim Erstellen einer Programmieraufgabe vom Dozenten festgelegt. Dabei hat er verschiedene Möglichkeiten, Testläufe zu gestalten. Bei einfachen Aufgaben, wie sie am Anfang des Studiums auftauchen, ist oft eine Konsolenausgabe einer Berechnung verlangt. In diesem Fall kann der Dozent beliebig viele Testläufe erstellen, in denen er die richtige Funktion des Programms testen kann. Er kann Programmeingaben (bzw. simulierte Benutzereingaben) und erwartete Ausgaben definieren. Diese Testlaufdaten werden von DUESIE in eine XML-Datei geschrieben, die im Dateisystem abgelegt wird. ANT benutzt diese XML-Datei als Grundlage für alle eingereichten Programme zu dieser Aufgabe.

Im späteren Verlauf des Studiums werden jedoch auch anspruchsvollere Aufgaben gestellt, bei denen mehrere Klassen zu programmieren sind und evtl. gar keine Konsolenausgaben mehr erzeugt werden. Zudem werden dann auch grafische Benutzeroberflächen Bestandteil der Übungen sein. Für diese Aufgaben werden sog. Test-Driver definiert. Der Studierende muss nun fehlende Klassen und Methoden implementieren. Anschließend lädt er seinen Code wieder auf den Server. Auch hier befindet sich der Test-Driver, den ANT startet. Der Test-Driver arbeitet nun auf dem vom User eingereichten Programm und stellt fest, ob alle Tests erfolgreich durchgeführt werden können. Zudem ist es möglich, zum Bewerten einen erweiterten Test-Driver zu hinterlegen, der auch Tests durchführt, die dem Studierenden vorher nicht bekannt waren. So kann bewertet werden, in wie weit sich ein Studierender mit dem Programm auseinandergesetzt hat und auch andere Testfälle von sich aus berücksichtigt hat.

Zusätzlich zu diesen vordefinierten Tests kann das Tool PMD[11] eingesetzt werden. PMD untersucht Java Code auf mögliche Fehler, sowie nicht benutzten, suboptimalen, verkomplizierten und doppelten Code. PMD wird als externe Anwendung gestartet und erzeugt eine XML-Reportdatei die von DUESIE eingelesen und ausgewertet wird. Als Grundlage für seine Untersuchungen benutzt PMD sog. „Rulesets“. Diese Rulesets sind einzeln für verschiedene Zwecke vordefiniert und lassen sich je nach Bedarf zu –und abschalten. Zusätzlich erlaubt PMD es, eigene Rulesets zu entwerfen und zu verwenden. Jedoch ist die Erstellung eigener Rulesets sehr aufwendig. Mit PMD steht dem System ein mächtiges Werkzeuge zur Verfügung, um Code auch stilistisch zu bewerten und nicht nur auf Lauffähigkeit zu prüfen.

[10] www.gnu.org/software/make/
[11] http://pmd.sourceforge.net

Darüber hinaus liefert PMD ein weiteres Werkzeuge mit, das Tool CPD. CPD steht für „Copy-Paste-Detection“ und ermittelt duplizierten Sourcecode in beliebig vielen Eingabedateien. So ist es möglich, nach Ablauf der Abgabefrist, alle eingereichten Programme der Studierenden auf teilweisen oder komplett duplizierten Code zu testen. Sollte dies der Fall sein, wird dem Tutor der jeweiligen Personen eine Nachricht angezeigt. Daraufhin kann der Tutor sich selbst von den gefundenen Übereinstimmungen überzeugen.

Ein weiterer, wichtiger Aufgabentyp, ist die Objektorientierte Modellierung mit UML. Der Entwurf von Software als UML Diagramm stellt bei allen Softwareprojekten einen wichtigen und elementaren Schritt dar. Daher sollte die Modellierung mit UML ebenfalls Bestandteil der Programmierausbildung sein. Bisher war es so, dass UML Diagramme von Studierenden immer in Papierform eingereicht wurden. Entweder waren die Diagramme per Hand gezeichnet oder mit einer Software erstellt und anschließend ausgedruckt worden. Tutoren mussten diese Papierlösungen, wie auch andere Textaufgaben, mühsam durchlesen und korrigieren.

Um nun UML-Aufgaben direkt in DUESIE zu integrieren, waren zwei Probleme zu lösen. Zum einen musste ein Weg gefunden werden, UML Diagramme einzureichen und zum andern musste ein Korrekturverfahren entwickelt werden. Beides wurde in DUESIE gelöst. Um UML-Diagramme zu erstellen, wird das Open-Source Programm ArgoUML[12] verwendet. Das heißt, ein Studierender muss dieses Programm benutzten, um ein UML-Diagramm zu erzeugen und einzureichen. Vorteil dieser Lösung ist, dass das Dateiformat einheitlich ist und somit die Lösungen vergleichbar werden. ArgoUML muss nicht auf dem Rechner installiert werden, an dem der Student DUESIE benutzt. Optional kann dies geschehen, ist aber nicht notwendig, da sich die in Java geschriebene Software per Java-Web-Start starten lässt. Diese Technologie ermöglicht es, Java-Software ohne Installation, plattformunabhängig auf einem Rechner zu starten. Dazu wird lediglich ein Link geöffnet, der dann die Anwendung startet. Diese wird temporär auf dem lokalen Rechner abgelegt, benötigt aber keine Installation und ist auf jedem Rechner mit Java Laufzeitumgebung ausführbar.

ArgoUML bietet umfangreiche Funktionen zum Erstellen von UML-Diagrammen ist aber einfach zu bedienen. Nachdem der Studierende seine Lösung mit ArgoUML erstellt hat, speichert er diese an einem beliebigen Ort auf dem lokalen Rechner ab. Anschließend lädt er diese Projektdatei wieder über ein Webformular auf den DUESIE-Server hoch. Der Vorteil des ArgoUML Projektformats ist es, dass es sich um XML Dateien handelt, die von einem Container umgeben sind. DUESIE entpackt nun diese Projektdatei und kann anhand der XML Dateien das UML-Diagramm genau rekonstruieren.

Beim Erstellen einer UML-Aufgabe, hat der Dozent eine Art „minimale Musterlösung“ im System hinterlegt. Diese Lösung, ebenfalls ein mit ArgoUML erstelltes Diagramm, enthält alles, was eine richtige[13] Lösung mindestens braucht, aber nicht mehr. Dazu kann alles gehören, was in einem UML Diagramm vorkommen kann. Klassen, Assoziationen, Attribute usw. DUESIE errechnet nun auf Basis der „Musterlösung“ Übereinstimmun-

[12] http://argouml.titris.org
[13] Im Fall von UML kann nur bedingt von „richtig“ oder „falsch“ gesprochen werden.

gen, fehlende Teile und unnötige Teile in der Studentenlösung. Diese werden als Report im System abgelegt und dienen anschließend der Bewertung. DUESIE schlägt dem Tutor, der Zugriff auf alle Lösungen hat, eine Bewertung vor, diese ist aber, wie auch bei allen anderen Aufgabentypen, nicht verpflichtend. Der Tutor oder Dozent hat immer noch die Möglichkeit die Bewertung von Hand zu korrigieren, sollte sie vom System falsch gesetzt worden sein.

3.3 Bewertungsverfahren

DUESIE bewertet die eingereichten Aufgaben nach einem bestimmten Schema und zwar so, dass eine möglichst hoher Grad an Widerverwendbarkeit gegeben ist. Damit lässt sich das Bewertungsschema einerseits für jede im System verwaltete Vorlesung einzeln festlegen und zusätzlich noch beim Erstellen eines Übungsblattes. Beim Erstellen eines Übungsblattes muss der Dozent sog. Gewichtungen für die einzelnen Aufgabentypen festlegen. Eine Programmieraufgabe würde durch ihren großen Arbeitsaufwand sicherlich eine stärkere Gewichtung erhalten als eine Multiple-Choice Aufgabe.

Die abgegebene Lösung des Sudierenden wird nach der Abgabe korrigiert und anhand der erreichten Punktzahl in eine der folgenden Kategorien eingeteilt:

- vollkommen richtig: 85-100%
- guter Versuch: 50-85%
- Versuch: 10-50%
- falsch / keine Lösung: 0-10%

Die Prozentwerte für die Staffelung hinterlegt der Dozent individuell beim Anlegen einer neuen Vorlesung im System. Diese Werte sind nachträglich nicht mehr veränderbar, um ein wechselndes Bewertungsschema während einer laufenden Veranstaltung auszuschließen. DUESIE bewertet selbständig die eindeutigen Aufgabentypen wie Lückentexte und Multiple-Choice Aufgaben. Bei anderen Aufgaben schlägt es eine Bewertung vor, die ein Tutor erst bestätigen muss. Ein Freitext z.B. kann alle Schlüsselworte enthalten und damit 100% der Punkte vom System bekommen, obwohl der Text keinen Sinn ergibt oder inhaltlich falsch ist. Zudem hat der Tutor die Möglichkeit, zu jeder Aufgabe einen Kommentar zu hinterlassen, der dem Studierenden nach der Auswertung der Aufgaben angezeigt wird. Damit kann ein Feedback über die Präsenzübung hinaus gegeben werden. Um Missbrauch zu vermeiden, kann kein Tutor abgegebene Aufgaben verändern. Auch bei der Korrektur von Textaufgaben wird immer eine Kopie der Originalversion behalten.

3.3 Korrektur

DUESIE kann zwar eigenständig Aufgaben bewerten und korrigieren, doch bedarf es einer manuellen Nachkontrolle. DUESIE bewertet und korrigiert eigenständig alle Aufgaben. Dabei hat der Tutor jederzeit die Möglichkeit, die vom System ermittelte Bewer-

tung manuell zu korrigieren. Falls Aufgaben existieren, bei denen eine vollständig computergestützte Auswertung nicht möglich ist (Freitexte, etc.), kann der Tutor die vorgeschlagene Bewertung akzeptieren, oder er ändert sie entsprechend ab, in dem er sie in eine andere Bewertungskategorie einteilt.

Sobald Lösungen von Studierenden eingereicht werden, auch schon vor Ablauf der Abgabefrist, kann der Tutor mit der Korrektur beginnen. Wenn die Abgabefrist abläuft, werden alle Aufgaben vom System automatisch als abgegeben markiert. Der Studierende hat dann keine Möglichkeit mehr Aufgaben einzureichen. Auch unfertige oder nicht bearbeitete Aufgaben werden abgegeben. Der Tutor kann sich mit Hilfe verschiedener Filter durch die Aufgaben arbeiten. Entweder er kontrolliert alle Aufgaben eines Typs, oder direkt ganze Aufgabenblätter.

Bei Programmieraufgaben kann er sich den eingereichten Sourcecode anzeigen lassen, bzw. herunterladen. Zusätzlich hat er Zugriff auf alle Logdateien die der Compiler und ANT erzeugt haben. Damit kann er den Bewertungsvorschlag des Systems nach seinem Ermessen akzeptieren oder abändern.

Bei Modellierungsaufgaben mit UML hat er die Möglichkeit, mit ArgoUML direkt Lösungen der Studierenden per Link zu öffnen und sich anzeigen zu lassen.

3.4 Sicherheit

Bei der Verarbeitung von fremdem Sourcecode besteht immer die Gefahr, dass entweder absichtlich oder versehentlich Schadcode auf den Server hochgeladen wird. Um diese Gefahr auszuschließen, werden verschiedene Sicherheitsmechanismen verwendet. Zum einen ist es nicht möglich Binärdaten, egal welcher Art, auf den Server hochzuladen. Sollte Schadcode auf dem Server kompiliert worden sein und ausgeführt werden, verhindert der Java Security-Manager[14] ungewollten Zugriff auf nicht freigegebene Ressourcen. Der Security-Manager von Java wird z.B. bei Java Applets eingesetzt, um deren Zugriff auf lokale Ressourcen zu steuern. Ebenso lässt er sich für normale Java Anwendungen zuschalten. Der Manager benutzt eine Policy-Datei, in der festgelegt wird, welchen Zugriff die Anwendung auf lokale Ressourcen erhält. Im Normalfall wird gar kein Zugriff auf das Dateisystem gewährt. Sollte eine Übungsaufgabe Dateizugriff erfordern, so kann explizit ein sicheres Verzeichnis und genau eine Datei angeben werden auf der gearbeitet werden darf. Zusätzlich wird mit Hilfe des Linux-Tools „chroot“ das auszuführende Programm in einem eigenen Verzeichnis „eingesperrt“ von dem aus kein Prozess auf andere Dateien oberhalb der aktuellen Ebene zugreifen darf.

Sollte ein Programm in eine Endlosschleife geraten, absichtlich oder versehentlich, könnte dies bei einer ausreichend großen Anzahl an Prozessen zu einem starken Performanceeinbruch führen. Daher wird ANT angewiesen, Programme nach einer gewissen

[14] http://java.sun.com/j2se/1.4.2/docs/api/java/lang/SecurityManager.html

4 Ausblick und Fazit

Die online Programmierausbildung ist ein Feld, das für viele Hochschulen und andere Institutionen immer wichtiger und interessanter wird. Es wurde erkannt, welches Potential in diesem Bereich des E-Learnings vorhanden ist und man versucht durch verschiedenste Herangehensweisen dieses Potential auszuschöpfen.

In Zukunft soll DUESIE auch für weitere Programmiersprachen (C++, Visual Basic, etc.) und weitere Aufgabentypen wie z.B. Zeichnungen erweitert werden. Des Weiteren ist aktuell eine parallele Entwicklung zu DUESIE geplant, die ein online gestütztes Klausursystem umsetzen wird. Damit wären dann auch computergestützte Klausuren innerhalb der Informatik-Ausbildung möglich, die mehr als nur die reinen Multiple-/Single Choice Aufgaben beinhalten. Dazu sind aber die erhöhten Sicherheitsanforderungen zu bedenken [Ho07].

Literaturverzeichnis

[BEW04] F. Behringer, D. Engeldinger, K. Weicker: Web-basierte Administration des Übungsbetriebs mit ECLAUS, Fachtagung "e-Learning" der Gesellschaft für Informatik, 2004

[Ho07] A. Hoffmann: Ein prozessorientiertes und dienstbasiertes Sicherheitsmodell für elektronische Prüfungen an Hochschulen. In Ch. Eibl, J. Magenheim, S. Schubert, M. Wessner, Hrsg. DeLFI 2007: Deutsche e-Learning Fachtagung Informatik, Lecture Notes in Informatics, S. 297 f., Bonn, 2007. Gesellschaft für Informatik

[KSZ02] J. Krinke, M. Störzer, A. Zeller: Web-basierte Programmierpraktika mit Praktomat, Proc. Workshop Neue Medien in der Informatik-Lehre, Dortmund, Germany, Oktober 2002.

[MOS07] Thiemo Morth, Rainer Oechsle, Hermann Schloss, Markus Schwinn: Automatische Bewertung studentischer Software, Workshop "Rechnerunterstütztes Selbststudium in der Informatik". In: Ch. Rensing, G. Rößling, Hrsg.: Proceedings der Pre-Conference Workshops der 5. e-Learning Fachtagung Informatik, S. 109-116, Logos Verlag, Berlin, 2007

[SVW06] J. Schwieren, G. Vossen, P. Westerkamp: Using Software Testing Techniques for Efficient Handling of Programming Exercises in an E-Learning Platform (Electronic Journal of e-Learning (EJEL) , Volume 4 Issue 1 March), 2006

[VHL01] Gottfried Vossen, Bodo Hüsemann, Jens Lechtenbörger: XLX - Eine Lernplattform für den universitären Übungsbetrieb (Arbeitsberichte des Instituts für Wirtschaftsinformatik, Universität Münster, ArbeitsberichtNr. 79), 2001

[Ze99] A. Zeller: Funktionell und verständlich programmieren - so lernen es die Passauer. Softwaretechnik-Trends 19(3):29-34, August 1999.