UNIVERSITÄT DORTMUND

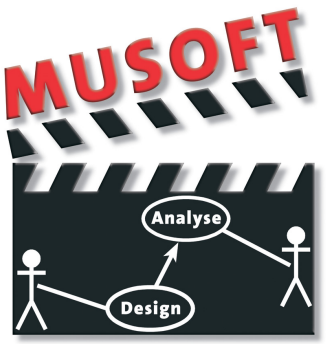
Fachbereich Informatik
Lehrstuhl für Software-Technologie

M E M O Nr. 133

MuSofT Bericht Nr. 2

Ergebnisbericht des Jahres 2002 des Projektes "MuSofT – Multimedia in der SoftwareTechnik"

Klaus Alfert Ernst-Erich Doberkat (Hrsg.) Gregor Engels



März 2003

Internes Memorandum des
Lehrstuhls für Software-Technologie
Prof. Dr. Ernst-Erich Doberkat
Fachbereich Informatik
Universität Dortmund
Baroper Straße 301

D-44227 Dortmund

ISSN 0933-7725

Ergebnisbericht des Jahres 2002 des Projektes “MuSofT– Multimedia in der SoftwareTechnik”

Klaus Alfert Ernst-Erich Doberkat Gregor Engels
(Hrsg.)

März 2003

Vorwort

Das Projekt *MuSofT – Multimedia in der Softwaretechnik* wird im Rahmen des Programmes Neue Medien in der Bildung seit Anfang März 2001 vom Bundesministerium für Bildung und Wissenschaft gefördert und endet im Dezember 2003. Das Projekt hat sich zum Ziel gesetzt, die Ausbildung im Bereich der Softwaretechnik in den Hochschulen durch den Einsatz Neuer Medien insbesondere in der Präsenzlehre zu unterstützen und zu verbessern.

Wie bereits im letzten Jahr wollen wir mit diesem Jahresbericht die Projektergebnisse dokumentieren. Nach zwei Jahren Laufzeit liegt das letzte Jahr vor uns, daher haben wir einen Schwerpunkt der Berichte der Teilprojekte auf die technische Realisierung gelegt.

Für die Belange von MuSofT, die über die Verantwortlichkeit einzelner Teilprojekte hinausgehen, haben wir von Projektbeginn an Koordinationsteams geplant und eingesetzt. Die Aufgaben dieser über die einzelnen Standorte hinweg tätigen Koordinationsteams mit den Namen KT1 bis KT6 lassen sich wie folgt charakterisieren:

KT1 erarbeitet einheitliche Richtlinien für die didaktische Konzeption von Lerneinheiten.

KT2 befasst sich mit der inhaltlichen und stilistischen Abstimmung von Lerneinheiten.

KT3 hat die Aufgabe, die Einsetzbarkeit der erstellten Materialien in anderen Studiengängen als der Kerninformatik, insbesondere auch in Ingenieurstudiengängen, zu untersuchen.

KT4 ist für die Bereitstellung eines Internetportals zuständig, über welches die erstellten Lerneinheiten und die dazugehörigen Werkzeuge angeboten werden.

KT5 koordiniert Medienproduktionen, die über Teilprojektgrenzen hinweg möglich sind.

KT6 beschäftigt sich mit den Grundlagen für die Nachhaltigkeit von MuSofT.

Zusätzlich zu KT1 bis KT4 haben wir haben im Laufe des Jahres 2002 KT5 und KT6 gebildet, um neu aufgekommene Fragestellungen zu bearbeiten. Von den sechs Koordinationsteams hat KT3 seine Arbeit bereits abgeschlossen. Die übrigen noch aktiven fünf Teams stellen ihre aktuellen Ergebnisse in dieser Sammlung vor.

Eine weitere teilprojektübergreifende Fragestellung beschäftigt sich mit Gender Mainstreaming für MuSofT. Diese umfassenden Aufgaben haben wir an externe Expertinnen vom Hochschuldidaktischen Zentrum der Universität Dortmund unter der Leitung von Frau Prof. Dr. Sigrid Metz-Göckel vergeben. Ihr Bericht wird uns in Kürze vorliegen und ebenfalls in dieser Berichtsreihe erscheinen.

Wir möchten nun abschließend allen beteiligten Kolleginnen und Kollegen dafür danken, dass sie die Beiträge für diesen zweiten Projektbericht geschrieben und begutachtet haben.

Dortmund und Paderborn,
im März 2003

Dr. Klaus Alfert
Prof. Dr. Ernst-Erich Doberkat
Prof. Dr. Gregor Engels

Inhaltsverzeichnis

Autorinnen und Autoren

Prof. Dr. Gregor Engels
Dipl.-Inform. Jan Hendrik Hausmann
Dipl.-Inform. Marc Lohmann
Dr. Annika Wagner
Arbeitsgruppe Informationssysteme
Fachbereich Mathematik/Informatik
Universität Paderborn

Dipl.-Inform. Dirk Jesko
Institut für Technische und
Betriebliche Informationsysteme
Otto-von-Guericke-Universität Magdeburg

Prof. Dr. Johannes Magenheim
Olaf Scheel
Arbeitsgruppe Didaktik der Informatik
Fachbereich Mathematik/Informatik
Universität Paderborn

Dr. Klaus Alfert
Prof. Dr. Ernst-Erich Doberkat
Dipl.-Inform. Corina Kopka
Dipl.-Inform. Jörg Pleumann
Lehrstuhl für Software-Technologie
Fachbereich Informatik
Universität Dortmund

Prof. Dr. Silke Seehusen
Dipl.-Ing. (FH) Nina Nissen
Fachbereich Elektrotechnik
Fachhochschule Lübeck

Dipl.-Inform. Peter Aschenbrenner
Prof. Dr. Andy Schürr
FG Echtzeitsysteme, FB 18
TU Darmstadt

Prof. Dr.-Ing. Fritz Schmidt
Institut für Kernenergetik und Energiesysteme
Universität Stuttgart

Prof. Dr. Udo Kelter
Praktische Informatik / Softwaretechnik
Fachbereich Elektrotechnik und Informatik
Universität-Gesamthochschule Siegen

MuSofT-Teilprojekt 1.1 Ergebnisbericht 2002

Jan Hendrik Hausmann, Marc Lohmann, Annika Wagner

Im Rahmen des Projektes MuSofT (Multimedia in der Softwaretechnik) ist es die Aufgabe des Teilprojektes 1.1. Lehreinheiten zum Thema Videogestützte Anforderungsdefinition zu erzeugen. Dazu wurden im Jahr 2001 bereits umfangreiche Vorarbeiten geleistet, die in einem ersten Einsatz in der Hochschullehre im Wintersemester 2001/2002 resultierten. In diesem Bericht wird die Weiterentwicklung der Lehreinheit im Jahre 2002 sowie Konzeptionen für zukünftige Arbeiten geschildert.

Inhaltsverzeichnis

1 Einleitung

Dieser Bericht beschreibt die Arbeit des MuSofT-Teilprojektes 1.1, dessen Aufgabe die Erstellung einer Lehreinheit zum Thema Videogestützte Anforderungsdefinition ist. Der erste

Jahresbericht zeigte dabei die konzeptionellen Grundlagen der Einheit auf und stellte erste Realisierungserfolge und einen ersten Einsatz der Materialien vor. In diesem Bericht wird nun beschrieben, welche Erkenntnisse aus diesem Einsatz gewonnen werden konnten und wie diese in einer Ergänzung und Verbesserung des Materials resultierten. In Abweichung zur vorgegebenen Struktur werden wir daher das Kapitel Evaluation vorziehen, da die Evaluationsergebnisse die Grundlage bilden, um die LE fortzuentwickeln.

Drei wesentliche Weiterentwicklungen der Lehreinheit sind im Berichtszeitraum festzustellen: Zum einen ergab die Evaluation des ersten Einsatzes der von uns geschaffenen Materialien, dass die Studierenden eine große Unsicherheit bei der Umsetzung von unstrukturierten (realen) Informationen in strukturierte Modelle hatten, besonders was die erwünschte/benötigte Detaillierung und Form der Ergebnisse anging. Um diesem zu begegnen, wurde die Dokumentenzentrierung als neue didaktische Leitlinie in die Lehreinheit eingebracht. Anhand von vorgegebenen strukturierten Dokumenten (Pflichtenhefte), die durch Beispiele illustriert werden, sollen die Studierenden eine stärkere Anleitung für die Form und das Ausmaß der von ihnen erwarteten Lösung erhalten. Die nächste zentrale Entwicklung war die planmäßige Erweiterung der Lehreinheit von übungszentrierten Materialien um Vorlesungsfolien. Schließlich wurde mit der Konzeption eines thematisch aufbauenden Moduls für das Hauptstudium begonnen.

Neben der Erstellung neuen Materials wurde darauf geachtet, dass bestehende Materialien so aufbereitet und dokumentiert werden, dass die Weitergabe über das MuSofT-Portal an andere Lehrende ermöglicht wird. Nur durch eine solche „Abrundung“ von Materialien kann sich ein nachhaltiger Erfolg des Projektes einstellen.

Der abschließende Ausblick stellt die Konzepte der Arbeit für das nächste Jahr vor. Auch hier wird der Aspekt der Nachhaltigkeit betont.

2 Vorstellung der Lehreinheit

Die Lehreinheit Anforderungsdefinition soll Probleme aufzeigen, die typischerweise beim Erfassen, Strukturieren und Notieren der Anforderungen von Benutzern an ein neues Softwaresystem entstehen. Sie soll darüber hinaus den Studierenden die Fähigkeiten vermitteln, diesen Problemen in einem praktischen Kontext zu begegnen.

Die Lehreinheit richtet sich an zwei unterschiedliche Zielgruppen. Einerseits sollen Studierende ohne Vorkenntnisse im Bereich des Software Engineering an die Problematik der Anforderungsdefinition herangeführt werden und Grundkenntnisse erwerben, andererseits sollen aber auch Vertiefungen für Studierende mit eben diesen Grundkenntnissen angeboten werden.

Für diese Ausrichtung der Lehreinheit auf zwei Zielgruppen mussten wir definieren, was unter Grundkenntnissen der Anforderungsdefinition zu verstehen ist. Dafür haben wir eine Gewichtung der im letzten Jahresbericht dargestellten inhaltlichen Lernziele vorgenommen. Danach ist im Rahmen der Vermittlung von Grundkenntnissen der Anforderungsdefinition folgendes zu leisten:

- Den Studierenden soll die Notwendigkeit einer Domänenanalyse bewusst gemacht werden. Grundlegende Techniken zur Beschreibung des Ergebnisses der Domänenanalyse sollen vermittelt werden. Techniken der Informationsbeschaffung bleiben unberücksichtigt.

- Methoden zur Spezifikation von Anforderungen sollen vermittelt werden.

- Durch eine explizite Anleitung sollen die Studierenden an die Dokumentation ihrer Entscheidungen herangeführt werden. Eine eigene Abschätzung wie viel und welche Art von Dokumentation an einer Stelle sinnvoll ist, wird nicht erwartet.

- Den Studierenden soll die Notwendigkeit einer Spezifikationsanalyse, also die (kritische) Reflexion der formal spezifizierten Modelle, bewusst gemacht werden. Inhaltlich werden im Rahmen dieser Analyse eher typische studentische Fehler diskutiert, als die Probleme einer realen Spezifikationsanalyse. Aufwands- und Risikobewertungen bleiben daher vollständig unbehandelt.

Diese Grundkenntnisse sollen im *Basismodul* unserer Lehreinheit vermittelt werden, auf dessen Entwicklung der Schwerpunkt unserer Arbeit während des Berichtszeitraums lag. Basis für die Weiterentwicklung der im letzten Jahr zu diesem Thema erstellten Materialien bildete die Evaluation, die im folgenden Kapitel beschrieben wird. Dieses Basismodul umfasst dabei Vorlesungsfolien sowie Übungsmaterialien (Haus- und Präsenzarbeitsaufgaben), wobei in beiden Lehrformen multimediale Elemente (Videos/Animationen) die zu spezifizierende Domäne darstellen.

Daneben wurde die Konzeption für das *aufbauende Modul* konkretisiert. Als spezielles Thema kristallisierte sich hier die *Zielorientierte Anforderungsdefinition* heraus. Dieses Thema ist als Aufbaumodul geeignet, da es ein grundsätzliches Verständnis für die Problematik der Anforderungsdefinition voraussetzt, darüber hinaus jedoch mit neuen Ideen und Aspekten eine intensivere Auseinandersetzung mit den Problematiken erlaubt. Das Thema ist ein aktuelles Forschungsthema, welches sowohl im Bereich des Requirements Engineering, aber auch in allgemeinen Softwaretechnik-Communities zunehmend Interesse und Verbreitung findet. Wir gehen daher davon aus, dass ein Lernmodul zu diesem Thema in Zukunft für verschiedenen Lehrenden von Interesse sein wird. Im Bezug auf die im letzten Bericht identifizierten Lernziele unterstützt diese Einheit besonders:

- **Zieldefinition.** Dies ist ein zentrales Konzept der Zielorientierten Anforderungsdefinition. Abstrakte Ziele müssen aufgestellt und alternative Realisierungen von diesen verglichen werden.

- **Spezifikation.** Im Lernmodul Zielorientierte Anforderungsdefinition sollen Spezifikationstechniken aus dem Bereich der temporalen Logik eingesetzt werden. Die hierfür benötigten Grundkenntnisse werden in der Einheit vermittelt.

- **Spezifikationsanalyse.** Es existieren umfangreiche Techniken, um zielorientierte Anforderungsdefinitionen auf Konflikte, Abhängigkeiten und Lücken hin zu untersuchen. Die Studierenden sollen lernen diese Techniken zu nutzen und die dahinter liegenden Konzepte verstehen.

Auch für dieses Aufbaumodul werden Materialien zur Durchführung von Vorlesungen und Übungen erstellt. Eine erste Version der Vorlesungsfolien liegt bereits vor.

3 Evaluation

Teile der Lehreinheit Videogestützte Anforderungsdefinition wurden bereits im Wintersemester 2001/2002 eingesetzt. Es handelte sich dabei hauptsächlich um Übungsaufgaben, die die Studierenden Anforderungsdefinitionen anhand von multimedial illustrierten Beispielen durchführen ließen. Diese Übungen waren eingebettet in den Kontext einer Vorlesung, die das Thema Anforderungsdefinition bisher ohne den Einsatz multimedialer Elemente zu vermitteln suchte. Diese Vorlesung und der darin eingebundene Einsatz der MuSofT-Materialien wurde evaluiert. Die Absicht hierbei war es, Ansatzpunkte zu finden, in denen eine Verbesserung der didaktischen Qualität möglich wäre. Diese Verbesserungen sollten dann in den neu zu erstellenden Folien zu unserer Lehreinheit integriert werden. In den folgenden Abschnitten wird dargestellt, wie der Einsatz konkret aussah, welche Evaluationen durchgeführt wurden und welche Schlussfolgerungen aus diesen Evaluationen zu ziehen sind.

3.1 Einsatz der Materialien

Die Übungsmaterialien des Basismoduls sind in der Vorlesung Techniken des Softwareentwurfes I im Wintersemester 2001/2002 und im Wintersemester 2002/2003 eingesetzt worden. Sie wurden weiterhin zum Sommertrimester 2002 der Bundeswehruniversität in München zur Verfügung gestellt, ein Feedback über den Einsatz oder den Erfolg dieses Einsatzes haben wir von dort leider nicht erhalten. Die Materialien bilden weiterhin einen Teil der Vorlesung Einführung in Softwareengineering an der Technischen Universität in Braunschweig und der Vorlesung Methodische Softwareentwicklung an der Universität Kassel im Wintersemester 2002/2003.

Die Konzepte für die Hauptstudiumsvorlesung (Aufbaumodul) wurden an der Universität Paderborn erprobt, indem eine Auswahl möglicher aufbauenden Themen zur Anforderungsdefinition genutzt wurde, um ein Seminar fürs Hauptstudium im Sommersemester 2002 durchzuführen. Dieses bot die Möglichkeit, Studenten mit den weitergehenden Ansätzen zu konfrontieren und herauszufinden, inwieweit sich interessante Lernmodule daraus konstruieren lassen. Eine formale Evaluation fand hier nicht statt.

3.2 Durchführung der Evaluation

Der Einsatz der Materialien im Grundstudium wurde und wird begleitet von einer Reihe evaluierender Maßnahmen. Grundsätzlich muss man dabei unterscheiden zwischen dem Feedback, das von Studenten gegeben wird, einer Erfolgskontrolle im Sinne der Lernziele sowie einer unabhängigen Begutachtung des Lehrmaterials. Wir haben alle diese Formen der Evaluation durchgeführt.

Für Veranstaltungen an der Universität Paderborn wird grundsätzlich eine studentische Evaluation von der Fachschaft durchgeführt. Da diese jedoch nur ein sehr allgemeines Stimmungsbild der Vorlesung vermittelt, wurde in den Vorlesungen TSEI im letzten und in diesem Wintersemester eine spezielle Befragung der Studenten zu den eingesetzten MuSofT-Materialien durchgeführt. Die Ergebnisse dieser Befragung vom letzten Jahr sind in einem auf dem MuSofT-Server verfügbaren Dokument zusammengefasst. In diesem Jahr wurde die studentische Befragung erneut durchgeführt, die Ergebnisse sind verfügbar, jedoch war eine

detaillierte Auswertung aus Zeitgründen noch nicht möglich. Eine ähnliche Befragung ist an der Universität Braunschweig geplant, diese kann jedoch erst nächstes Jahr durchgeführt werden, da der Einsatz der MuSofT-Materialien dort noch nicht abgeschlossen ist. Neben diesen offiziellen Befragungen haben wir auch freie Meinungsäußerungen der Studenten (über die Mailinglisten oder die dafür eingerichtete Newsgroup) zu den Veranstaltungen gesammelt, um ein Feedback von dieser Seite zu erhalten.

Den Erfolg der Vorlesung TSEI und insbesondere den der MuSofT-Einheit Anforderungsdefinition konnten wir nachvollziehen, indem wir die Anschlussveranstaltung dieser Vorlesung, das Softwarepraktikum, beobachtet und ausgewertet haben. Im Softwarepraktikum sollen Gruppen von Studenten relativ selbstständig eine Softwareentwicklung betreiben. Wir haben 4 Gruppen in der Anfangsphase dieser Entwicklung (der Anforderungsdefinitionsphase) begleitet, um zu erkennen, ob die vorhergehende Vorlesung die Studenten mit den notwendigen Kenntnissen und Fähigkeiten zur Bewältigung dieser Aufgabe ausgestattet hat. Weiterhin wurden studentische Lösungen zu Übungsaufgaben (fertige Pflichtenhefte), die aus dem MuSofT-Fundus stammen, gesammelt, um daran Lernerfolge und Probleme detailliert diagnostizieren zu können.

Einen dritten Baustein der Evaluation stellt der Peer-Review dar. Um die MuSofT-Materialien in drei Universitäten einbringen zu können, mussten drei Dozenten (wobei zwei keine Verbindungen zum MuSofT-Projekt haben) das gegebenen Lehrmaterial beurteilen und unter Einbeziehung der Ergebnisse der studentischen Evaluation vom Vorjahr sowie ihrer speziellen Anforderungsprofile weiterentwickeln. Diese Weiterentwicklungen sind nun in die MuSofT-LE 1.1 eingeflossen.

3.3 Ergebnisse der Evaluation

Die Ergebnisse der studentischen Befragung vom WS 2001/2002 in Paderborn zeigten ein in vielen Fällen positives Ergebnis, wiesen jedoch auch auf Schwachstellen hin. Eine solche Schwachstelle war die große Unsicherheit der Studenten, welcher Detaillierungsgrad bei der Bearbeitung von Aufgaben als Lösung angemessen sei. Weiterhin ergab die Beobachtung der Gruppen in den Folgeveranstaltungen, dass diese zwar erfreulicherweise die benötigten Methoden, Sprachen und Tools sehr selbstverständlich (und größtenteils richtig) einsetzen konnten, sie jedoch auf einige der formalen Aspekte einer Softwareentwicklung (Erstellung eines standardisierten Pflichtenheftes) nur unzureichend vorbereitet waren.

Insbesondere durch die gute Leistung der Studierenden in der Folgeveranstaltung sehen wir uns in den von uns umgesetzten Lehrkonzepten bestätigt. Wir sahen jedoch in der Integration des Pflichtenheftes in die Inhalte der Lehreinheit die Chance, auch das Problem der Unsicherheit über den Detaillierungsgrad und die Form von studentischen Lösungen anzugehen. Wir haben daher die Veranstaltung stärker auf ein dokumentenzentriertes Vorgehen ausgerichtet.

4 Technische Realisierung

Die Weiterentwicklung der Lehreinheit 1.1 hin zu Vorlesungsinhalten sowie das Integrieren des dokumentenzentrierten Ansatzes machen zahlreiche Arbeiten notwendig, die im Folgen-

den vorgestellt werden. Wir unterscheiden dabei zwischen der Konzeptionen und der Erstellung von Materialien sowie der Integration und Dokumentation dieser Materialien.

4.1 Konzeption

Im Mittelpunkt des Basismoduls steht das Pflichtenheft. Das Pflichtenheft umfasst die Ergebnisse der Domänenanalyse ebenso wie die Spezifikation der Anforderungen. Die Studierenden werden dazu angeleitet, ein Pflichtenheft basierend auf einer strukturierten Vorlage (sog. Template) zu erstellen. Dieses Template gibt eine Gliederung für das Pflichtenheft vor und beschreibt zu jedem Abschnitt die Aufgabe, die er zu erfüllen hat. Dadurch wird dem Studierenden ermöglicht nachzuvollziehen, mit welchem *Ziel* er etwas tut. Außerdem enthält es konkrete Arbeitsanweisungen und bietet auf diese Art und Weise dem Lernenden Hilfen an, *wie* bestimmte Aufgaben zu erfüllen sind.

Neben dem Template zur Erstellung eines Pflichtenheftes wird dem Lernenden ein Beispielpflichtenheft zur Verfügung gestellt, das mit Hilfe des Templates für ein komplexeres Projekt erstellt wurde. Dieses Beispielpflichtenheft kann als Illustration der abstrakten Arbeitsanweisungen im Template angesehen werden.

Nach dem gleichen Konzept verfahren wir bei der Vermittlung der Spezifikationsanalyse. Hierfür ist ein eigener Dokumententyp vorgesehen, der Review des Pflichtenheftes. Auch hier wird wieder über ein Template eine praktische Anleitung bereitgestellt, eine solche Analyse durchzuführen, und durch ein Beispieldokument illustriert.

Unser didaktisches Konzept stellt also zu erstellende Dokumente in den Mittelpunkt. Wir bezeichnen es daher auch als dokumentenzentrierten Ansatz für die Ausbildung. Dem Anfänger, der über keinerlei Erfahrung in größeren Softwareentwicklungsprojekten verfügt, bieten sich so wesentlich mehr Orientierungsmöglichkeiten um sich schrittweise einer Lösung anzunähern.

4.2 Medienerstellung

Zur Verwirklichung des dokumentenzentrierten Ansatzes mussten Vorlesungsfolien geschaffen werden, die diesen Ansatz unterstützen. Templates und Beispieldokumente müssen den Studierenden ebenso zur Verfügung gestellt werden wie Aufgabenblätter, die dieses Vorgehen unterstützen. Weiterhin sollte auch in der Vorlesung die videogestütze Vorgehensweise verfolgt werden, die bereits im letzten Jahr bei den Übungen erfolgreich eingesetzt wurde. Im Folgenden werden die Arbeiten zu diesen Themen detaillierter beschrieben:

Für eine Lehrveranstaltung zum Thema Videogestütze Anforderungsdefinition wurde ein *Foliensatz* entwickelt. Aufgabe dieses Foliensatzes ist es, zur Erstellung der Dokumente anzuleiten und ihren Aufbau zu erläutern. Als illustrierendes Beispiel werden innerhalb des Foliensatzes Ausschnitte aus dem Beispielpflichtenheft verwendet. Der Foliensatz umfasst Folien für fünf 90-minütige Veranstaltungen. Folgende Themen werden behandelt:

- Überblick über das Pflichtenheft
- Modell des Problembereichs (inkl. Einführung UML Objekt- und Klassendiagramme)

- Geschäftsprozessmodellierung (inkl. Einführung UML Use Case- und Aktivitätendiagramme)
- Produktfunktionen
- Qualität des Pflichtenheftes

Dabei erläutern die ersten vier Vorlesungen das Template für das Pflichtenheft, während die letzte Vorlesung in den Review des Pflichtenheftes gemäß diesem Template einführt.

Sowohl Templates als auch Beispieldokumente wurden als Word-Dokumente realisiert. *Templates* arbeiten mit Feldern, die ein leichtes Umsetzen der konkreten Arbeitsanweisungen ermöglichen sollen. Die Beispieldokumente sind durch studentische Mitarbeiter des Projektes erstellt worden, die die entsprechende Lehrveranstaltung direkt vorher gehört hatten. Ihr Wissensstand ist also von dem der Teilnehmer der Lehrveranstaltung noch nicht weit entfernt. Auf diese Art und Weise konnten bei der Erstellung auftretende typische studentische Fehler ermittelt werden. Diese Fehler bilden die Grundlage des Templates für den Review des Pflichtenheftes. In diesem Template werden die Studierenden angeleitet in dem von ihnen angefertigten Pflichtenheft nach typischen Fehlern zu suchen.

Im *Beispielpflichtenheft* geht es um die Entwicklung einer Lagerverwaltungssoftware für ein horizontales Karusselllager, das im Kreiskrankenhaus Heidenheim als zentrales Medikamentenlager eingesetzt werden soll. Zur Einführung des Beispiels wird innerhalb der Vorlesung ein *Video* über das Karusselllager gezeigt, das den Studierenden auch zur Verfügung gestellt wird. Aufgabe dieses Videos ist es, die Domänenanalyse zu motivieren. Das Beispiel wurde gewählt, da ein entsprechendes Video durch Kontakte der MuSofT-Partner in Magdeburg zur Verfügung gestellt werden konnte. Ursprünglich war dies als eins der durchgehenden Fallbeispiele in MuSofT angedacht, es scheint jedoch, dass wir die einzige Gruppe sind, die dieses Beispiel nun einsetzt. Es hat sich herausgestellt, dass sich das Video gut in den Kontext der Vorlesung einfügt und die entsprechenden Konzepte motiviert.

Ein wesentlicher Punkt bei der Erstellung der Folien war die Etablierung eines einheitlichen, auf MuSofT zugeschnittenen *Layouts*. Dabei sollte auch die Kritik der Studierenden aus dem letzten Semester berücksichtigt werden, wo aufgrund eines dunklen Hintergrundes die Folienprojektion nicht gut lesbar war. Für die Folien wurde ein weißer Hintergrund mit schwarzer Schrift gewählt wobei rote und blaue Elemente als Hervorhebungen eingesetzt werden. Um eine Orientierung in den Folien zu erleichtern und ein einheitliches Aussehen zu erreichen wurde weiterhin eine einheitliche Symbolik etabliert, für die grafische Elemente gestaltet wurden. Auf diese Weise können Verweise auf Übungen oder externe Literaturquellen schnell in den Folien gefunden werden. Eine Beispielfolie ist in Abb. 1 dargestellt.

Neben den Dokumenten und den Folien existiert eine *Aufgabensammlung*. Aufgrund der unterschiedlichen Konzepte der Lehrveranstaltungen an der TU Braunschweig und der Universität Paderborn gibt es auch zwei Konzepte für vorlesungsbegleitende Aufgaben. An der Universität Paderborn ist der Stundenumfang der Lehrveranstaltung größer. Daher ist das Lernziel in der Ergebnistaxonomie auch höher angesetzt. Die Studierenden sollen hier in die Lage versetzt werden, ein Pflichtenheft zu erstellen und einen Review desselben durchzuführen. An der TU Braunschweig ist das Lernziel auf das Verständnis des Aufbaus des Pflichtenheftes und die Fähigkeit ein solches zu lesen reduziert. Die Erstellung eigener Dokumente ist hier erst Lernziel des Softwarepraktikums im darauf folgenden Semester.

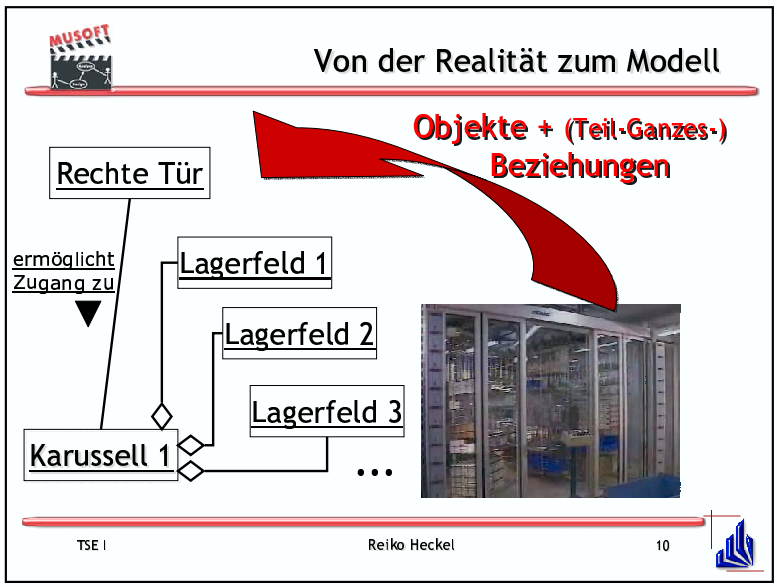


Abbildung 1: MuSofT-Folie der Vorlesung TSE I

Die Aufgabensammlung, die an der TU Braunschweig zum Einsatz kommt, enthält Verständnisfragen zum Beispielpflichtenheft und regt zur intensiven Beschäftigung mit demselben und Diskussion über dasselbe an. Hier wird der Tatsache Rechnung getragen, dass das Pflichtenheft aufgrund seines Umfanges von 43 Seiten unmöglich vollständig im Rahmen der Vorlesung diskutiert werden kann.

An der Universität Paderborn kommen hingegen Aufgaben zum Einsatz, bei denen die Studierenden für ein weniger komplexes Beispiel ein eigenes Pflichtenheft und einen eigenen Review erstellen sollen. Bei diesem weniger komplexen Beispiel handelt es sich um die Konzeption einer im Internet spielbaren Version des Brettspieles „Mississippi Queen“. Dies entspricht der geplanten ersten Fallstudie unseres Vorjahresberichts. Hier kommen also die bereits entwickelten multimedialen Materialien in Form von Animationen zur Erläuterung des Spielablaufes und Originaldokumenten aus der Domäne zum Einsatz. Ein Video für einen beispielhaften Spielablauf ist noch nicht realisiert.

Zusammenfassend ist zum Stand der Aufgabensammlung festzustellen, dass es uns gelungen ist, aus einem grundsätzlichen gut dokumentierten Materialpool Aufgaben für unterschiedliche Anforderungsprofile zu entwicklen. Es stehen nun thematisch ähnliche, in ihren Ansprüchen jedoch stark variierende Aufgabentypen zur Verfügung, die von anderen Lehrenden eingesetzt werden können. Musterlösungen dokumentieren unseren Erwartungshorizont und erleichtern die Wiederverwendung. Das gegebene Material kann weiterhin zur Konstruktion neuer Aufgabentypen herangezogen werden.

Für das Aufbaumodul sind erste Realisierungen verfügbar. Ein Foliensatz stellt die wesentlichen Aspekte des Themengebietes „Zielorientierte Anforderungsdefinition“ dar. Dieser

muss jedoch noch um Übungsaufgaben und ein multimediales Beispiel ergänzt werden. Als Beispiel ist hier die Integration der Aufnahmen aus dem Bertelsmann Logistik-Zentrum vorgesehen. Dies entspricht der Fallstudie III, wie sie in unserem letzten Jahresbericht beschrieben wurde. Wir beteiligen uns daher aktiv an der Gruppe, die diese Aufnahmen planen und durchführen soll. Solange diese Aufnahmen jedoch nicht vorliegen, können weitere Arbeiten am Aufbaumodul nicht konkret umgesetzt werden.

4.3 Integration und Nachhaltigkeit

Da das Ziel von MuSofT primär in der Weitergabe von Lehrmaterial an andere Hochschulen zu sehen ist, halten wir es für zentral, die produzierten Medienobjekte so aufzubereiten, dass diese Weitergabe auch praktisch handhabbar wird. Dies berührt prinzipiell eine inhaltliche Einbettung, die Dokumentation des Materials, eine technische Integration und Aufbereitung und die Klärung rechtlicher Fragen. Ohne diese flankierenden Maßnahmen kann Lehrmaterial nur „prinzipiell“ ausgetauscht werden, eine realistische Nutzung ist jedoch nicht zu erwarten.

4.3.1 Inhaltliche Einbettung

Wie bereits im vorhergehenden Kapitel erläutert, ist ein Grossteil der Materialien für das Basismodul inhaltlich in einer Weise integriert und miteinander verzahnt, dass er in verschiedenen Vorlesungen bereits Verwendung findet. Die Vorlesungen vermitteln das dokumentenorientierte Vorgehen, für das Aufgaben und vertiefende Aufgaben verfügbar sind. Darüber hinaus gibt es noch einige Medienobjekte, die Zusatzmaterial darstellen, um neue Aufgaben konstruieren zu können. Das Angebot wird abgerundet durch Klausurfragen, die das gelernte Wissen überprüfen sollen. Damit steht einem Lehrenden zu diesem Thema eine umfassende und aufeinander abgestimmte Sammlung von Materialien zur Verfügung, die in anderen Veranstaltungen Verwendung finden können.

4.3.2 Dokumentation

Jedem Softwaretechniker ist bewusst, dass neben dem eigentlichen Programmcode eine Dokumentation eines Programmes unerlässlich ist, um den Sinn eines Programms erfassen zu können. Eine ähnliche Situation ergibt sich für Lehrmaterialien. Der Lehrende, der fremde Materialien einsetzen will, muss nicht nur das Material zur Verfügung gestellt bekommen, sondern auch die Zusammenhänge und das übergreifende Konzept verstehen, das eben nicht auf den Folien steht. Um diese Art von Dokumentation zu erstellen, wurde bei der Vorlesung der MuSofT-Folien in Braunschweig der Ton aufgezeichnet. Die einzelnen Folien wurden mit den entsprechenden Erläuterungen synchronisiert. Dieses mit Hilfe des Internet Explorers abspielbare Hörbuch zur Vorlesung richtet sich nicht nur als Wiederholungsangebot an Studierende, sondern ist auch als Anleitung für Dozenten gedacht, die das erzeugte Material übernehmen wollen.

Für Übungsaufgaben ist eine schriftliche Dokumentation in Form von Lösungshorizonten und Musterlösungen verfügbar. Damit können Lehrende die Aufgaben verstehen und Adaptionen gemäß ihren eigenen Anforderungen durchführen.

4.3.3 Rechtliches

Ein weiterer wichtiger Punkt, der im Jahre 2002 angegangen wurde, ist die rechtliche Absicherung des Medieneinsatzes. Das von uns verwendete Spiel Mississippi Queen ist eine Kreation des Spieleautors Werner Hodel und wurde produziert vom Goldschläger Verlag. Da es inzwischen nicht mehr produziert wird, sind die Urheberrechte wieder an Herrn Hodel zurückgefallen. Der Autor hat zugestimmt, dass Elemente des Spiels Verwendung in unseren Lehrveranstaltungen finden. Eine detaillierte Vereinbarung steht noch aus, jedoch sind die bisherigen Kontakte sehr positiv verlaufen, so dass wir davon ausgehen, dies ohne weitere Komplikationen durchführen zu können. Bei den Videos des Regallagers handelt es sich um öffentlich verfügbare Werbevideos der Firma Mannesmann, eine abschließende rechtliche Klärung des Einsatzes dieser Medien steht noch aus.

4.3.4 Technische Integration

Unter der Technischen Integration verstehen wir die Vorbereitung auf die tatsächliche technische Weitergabe der Daten an andere Lehrende. Hierfür ist das von KT4 entwickelte MuSofT-Portal gedacht. Wir haben dafür bei der Erstellung von Materialien auf die von KT4 verbreitete Liste mit Dateiformaten geachtet: Es werden Powerpoint-Folien eingesetzt, von denen alternativ eine pdf Version verfügbar ist. Beispielmaterialien (Templates, Beispielpflichtenhefte) wurden parallel als Microsoft Word Dokument, Rich Text Format und als Ascii Text produziert. Die Animationen von Mississippi Queen sind im Macromedia Flash Format gespeichert und das Video des Karusselllagers ist als DivX codierter AVI verfügbar (hier steht eine Konvertierung in alternative (Streaming-)Formate noch an).

Die Annotation mit Metadaten und die Modularisierung, die für ein Einstellen der Materialien ins Portal notwendig sind, werden einen Schwerpunkt unserer Arbeit des nächsten Jahres bilden. Die Modularisierung wird sich dabei an der inhaltlichen Aufteilung des Themengebietes in einzelne Foliensätze orientieren, eine feinere Aufteilung muss auf Sinnhaltigkeit hin geprüft werden.

5 Planung für das nächste Jahr

Wir erwarten, dass der wesentliche Realisierungsaufwand für das Basismodul hinter uns liegt. Die Evaluation der aktuell eingesetzten Materialien gibt erneut Hinweise zu möglichen Verbesserungen, diese werden natürlich eingearbeitet, wir erwarten allerdings keine grundlegenden Umgestaltungen mehr. Einige offene Punkte in den Realisierungen wurden bereits im Text angesprochen, so muss etwa die Applikation zur Demonstration der Software-Qualitäten noch verbessert werden und ein Video zum Ablauf des Spiels Mississippi Queen ist zu erstellen.

Die Arbeiten zum Aufbaumodul werden mit der Planung und Durchführung der Dreharbeiten bei Bertelsmann fortgesetzt. Die dabei entstehenden Videos müssen dann eine inhaltliche Integration in die Vorlesung erfahren. Ob wir einen Einsatz dieser Lehmaterialien durchführen und evaluieren können, ist im Augenblick schwer abzuschätzen. Da es sich nur um Module handelt, ist eine geeignete Veranstaltung zu finden, in die diese eingebettet werden können. Auch für diese Materialien ist eine Aufbereitung zur Weitergabe vorzunehmen.

Wir sehen jedoch die wesentlichen Arbeiten an den MuSofT-Materialien im Bereich der Dokumentation und Integration, also der inhaltlichen und technischen Aufbereitung zur Weitergabe an andere Lehrende. Hier sind noch eine Reihe von Details zu verbessern, die im Einzelnen nicht besonders spannend wirken mögen, deren Vorhandensein jedoch wesentlich über den Erfolg einer solchen Einheit entschieden kann. Beispiele sind hier etwa die grafische Gestaltung oder das Vorhandensein von themenbezogenen Klausuraufgaben. Die gesamte Einordnung in die LOM-Schemata (und die damit einhergehende Modularisierung) wird in diesem Bereich einen Schwerpunkt unserer Arbeiten bilden.

6 Zusammenfassung

In diesem Bericht wurde dargestellt, welche Weiterentwicklungen das MuSofT-Material des Teilprojektes 1.1 seit dem letzten Jahresbericht genommen hat. Mit mehreren Einsätzen in der Hochschullehre und Evaluationen dieser Einsätze haben wir Informationen gewonnen, um diese Entwicklungen bedarfsgerecht vorzunehmen. Begleitende Arbeiten zur inhaltlichen, rechtlichen und technischen Abrundung bereiten die spätere Wiederverwendung der Materialien vor.

Literatur

MuSofT-Lerneinheit 1.2: Entwicklung von Informationssystemen

Dirk Jesko

Die Lerneinheit 1.2 befasst sich mit der Erstellung von Materialien für Lehrveranstaltungen des Fachgebiets Datenbanken. Der Bericht fasst die Ergebnisse des zweiten Projektjahrs zusammen, in dem die Implementierung verschiedener Werkzeuge im Vordergrund stand. Diese sollen als Ergänzung der bereits existierenden Folien dienen. Ein weiterer Schwerpunkt war die Evaluierung und Erweiterung eines Ansatzes zur Bereitstellung von Vorlesungsvideos befasste.

Inhaltsverzeichnis

1 Einleitung

Das Ziel des Teilprojekts 1.2 „Entwicklung von Informationssystemen“ ist die Erstellung von Medien und Werkzeugen für die Unterstützung einer Grundlagenvorlesung im Fachgebiet Datenbanken. Die Grundlage der Lerneinheit bildet die Magdeburger Datenbanken I Vorlesung, für die ein Foliensatz zur Verfügung steht. Aufgrund der Komplexität des Themengebiets können im Rahmen des Projektes dabei allerdings nur ausgewählte Aspekte betrachtet werden. Insbesondere im Bereich praktischer Übungen, z.B. bei der Modellierung und den Anfragesprachen stehen kaum Werkzeuge zur Verfügung. Im Rahmen des Projektes werden daher entsprechende Werkzeuge implementiert, die sowohl zur Präsentation in Vorlesungen, als auch zu Übungszwecken verwendet werden können. Weiterhin wurde im Rahmen des Projekts ein Ansatz zur Bereitstellung von Videomitschnitten der Vorlesung realisiert und evaluiert.

Der vorliegende Beitrag gibt einen Überblick des derzeitigen Projektfortschritts. Abschnitt 2 gibt zunächst eine Übersicht des Inhalts, dessen Gliederung und der möglichen multimedialen Unterstützung. Anschließend wird in Abschnitt 3 auf die technische Realisierung eingegangen. Erläutert wird der Ansatz zur Bereitstellung von Videomitschnitten der Vorlesung. Weiterhin wird auf die Entwicklung eines Werkzeugs zur ER-Modellierung und die Auswahl und Erweiterung eines SQL-Tools eingegangen. Im Wintersemester 2001/02 wurde die Datenbanken I Vorlesung aufgezeichnet und bereitgestellt, um den zu evaluieren. Abschnitt 4 geht auf die Ergebnisse ein. In Abschnitt 5 werden abschießend die Ergebnisse zusammengefasst und es erfolgt ein Ausblick auf die für das Jahr 2003 geplanten Arbeiten.

2 Vorstellung der Lerneinheit

Die Grundlage der Lerneinheit bildet der vorhandene Foliensatz. Dieser ist entsprechend [HS00] in größere Themengebiete gegliedert. Insbesondere im Zusammenhang mit der Bereitstellung der Vorlesungsvideos erfolgt derzeit eine weitere Aufteilung der Folien in kleinere thematische Blöcke. Eine umfassende Beschreibung und Strukturierung der Inhalte erfolgte bereits in [Jes02]. Für verschiedene dieser Themen bietet sich eine Unterstützung durch Medien oder Werkzeuge an, die im Rahmen des Projektes bearbeitet werden:

Motivation: In diesem Abschnitt wird der Einsatz von Datenbanken und deren Vorteile. Weiterhin werden grundlegende Begriffe eingeführt und es wird ein kurzer Einblick in die folgenden Themen gegeben. Für diesen Abschnitt ist der Einsatz eines Videos sinnvoll, das reale Vorgänge aus dem Bereich der Datenverarbeitung darstellt.

Konzeptioneller Entwurf: In diesem Abschnitt werden verschiedene Sprachen für die konzeptionelle Modellierung eingeführt, insbesondere verschiedene ER-Modelle und UML-Klassendiagramme. Für das Erlernen dieser Sprachen ist deren praktische Anwendung von großer Bedeutung. Daher wurde für diesen Bereich mit der Entwicklung eines Werkzeugs zur Modellierung begonnen.

Logischer Entwurf: Dieser Abschnitt stellt insbesondere Verfahren zur Abbildung der Modelle des konzeptionellen Entwurfs auf das Relationenmodell und zur Normalisierung vor. Zur Veranschaulichung werden diese implementiert, so dass sie mit beliebigen Daten schrittweise ausgeführt werden können.

Anfragesprachen: In diesem Modul werden Anfragesprachen und deren Grundlagen eingeführt. Insbesondere wird auf die Structured Query Language (SQL) eingegangen. Das Erlernen der Sprache erfordert praktische Übungen. Da bisher keine geeigneten Werkzeuge zur Verfügung stehen wird ein solches angepasst, so dass es sowohl in Vorlesungen, Übungen und im Selbstudium eingesetzt werden kann. Weiterhin werden beispielhaft Videos der Anwendung des Tools und zur Einführung von SQL erstellt.

Der Einsatz von Videos in anderen Bereichen, als der Motivation der Vorlesung ist nicht sinnvoll. In Seminaren und Übungen ist dies beispielsweise problematisch, da sich die verwendeten Beispiele häufig ändern, was eine Neuproduktion der Videos erforderlich machen würde.

3 Technische Realisierung

Dieser Abschnitt geht genauer auf technische Aspekte der Realisierung der Werkzeuge und die Aufbereitung der Vorlesungsvideos ein.

3.1 ER-Editor

Für praktische Übungen im Gebiet der konzeptionellen Modellierung wurde die Erstellung eines Editors für das Entity-Relationship-Modell (ER-Modell) vorgesehen. Da es diverse unterschiedliche Arten von ER-Modellen gibt und sich auch die Notationen teilweise stark unterscheiden sollte dieser Editor so gestaltet werden, dass eine Anpassung möglich ist.

Für die Erstellung des Editors wurden zunächst zwei Ansätze verfolgt. Zum einen die Erweiterung eines existierenden Modellierungstools, zum anderen eine eigene Implementierung basierend auf einem geeigneten Framework. Für den ersten Ansatz wurde u.a. die Erweiterung eines UML-Modellierungstools wie Together (`http://www.togethersoft.com/`) oder ArgoUML (`http://argouml.tigris.org/`) in Betracht gezogen, da auch die Modellierung mit UML-Klassendiagrammen Teil der Vorlesung ist. Together bietet bereits eine eigene ER-Syntax und die Möglichkeit sich mit Datenbanken über JDBC zu verbinden. Bei der Realisierung hat sich jedoch gezeigt, dass dieser Ansatz verschiedene Probleme aufwirft. Insbesondere die Komplexität der GUI dürfte bei Präsentationen zu Problemen führen, da für die eigentlich interessanten Schemata zu wenig Platz bleibt und von den wesentlichen Informationen ablenkt. Die notwendigen Änderungen erwiesen sich als zu umfangreich.

Daher wurde eine eigene Implementierung angestrebt. Zu diesem Zweck wurden zunächst verschiedene Java-Frameworks für die Erstellung von Grapheneditoren untersucht, da ein ER-Schema einen Graphen mit Knoten unterschiedlicher Typen zusammensetzt (Entität, Relation, Attribut, etc.). Diese werden durch Kanten verbunden, wobei bestimmte Regeln eingehalten werden müssen, z.B. dürfen zwei Entitäten nicht direkt miteinander verbunden werden. Analysiert wurden u.a. GenGEd [Bar00] und DiaGen [MK99]. Ersteres basiert auf Graph Transformation und bietet eine GUI, um Symbole, Transformationsregeln, Editor etc. zu spezifizieren. DiaGen basiert auf Hypergraphen und Hypergraph-Grammatiken, die in Textform definiert werden. Abschließend wird daraus automatisch Java-Code für einen Editor generiert, der manuell erweitert werden kann. Im Unterschied zu GenGEd sind die erstellten Editoren ohne die Entwicklungsumgebung lauffähig. Die Definition der Editoren erwies sich als schwierig, insbesondere im Falle von DiaGen (rein textuelle Definition). Weitere Probleme waren die Plattformabhängigkeit (GenGED) und Fehler beim automatischen Layout (DiaGen).

Schließlich wurde ein Prototyp eines Editors basierend auf Jazz [BMG00] erstellt. Dieses Framework wurde für die Erstellung von „Zoomable Userinterfaces“ (ZUI) erstellt. Jazz ist in Java implementiert und steht im Sourcecode zur Verfügung. Es bietet insbesondere Funktionalitäten für Visualisierungen, z.B. Zoom. Funktionen für das Editieren (Markieren, Verschieben, Kopieren, Löschen etc.) stehen ebenfalls zur Verfügung oder können relativ leicht realisiert werden. Schließlich stehen die notwendigen graphischen Konstrukte für die ER-Modellierung zur Verfügung. Bei der Implementierung ergaben sich allerdings auch verschiedene Probleme, z.B. Layoutfehler. Darauf wird im folgenden Abschnitt im Zusammenhang mit der Visualisierung der Tabellenstrukturen eingegangen.

3.2 SQL-Tool

Für den späteren praktischen Einsatz, insbesondere in der Entwicklung von Datenbanken, ist die Kenntnis von Anfragesprachen von Bedeutung. Wie bei der Modellierung können auch diese nur durch praktische Übungen erlernt werden. Derzeit stehen dafür keine geeigneten Werkzeuge zur Verfügung. Im Rahmen des Projekts wird ein solches angepasst. Bei der Auswahl waren verschiedene Aspekte von Bedeutung, u.a. Plattform- und Datenbankunabhängigkeit, Erweiterbarkeit und Laden und Speichern von Skripten. Insbesondere soll das Werkzeug auch die Visualisierung von Tabellen und deren Beziehungen ermöglichen. Schließlich muss die Oberfläche an unterschiedliche Einsatzszenarien anpassbar sein, z.B. Präsentation und Übungen am Rechner.

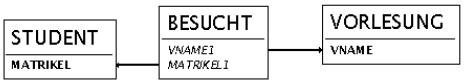
Die Anforderung der Plattformunabhängigkeit läßt sich durch den Einsatz eines auf Java und JDBC basierenden Tools realisieren. Derzeit existiert eine Reihe derartiger Werkzeuge, z.B. DBVisualizer (`http://www.minq.se/products/dbvis/`), Squirrel SQL Client (`http://squirrel-sql.sourceforge.net/`) und Independent SQL Tool (`http://isql.sourceforge.net/`). Unter diesen ist DBVisualizer das einzige, welches eine graphische Darstellung der Tabellen und deren Beziehungen bietet. Andererseits ist der Quellcode nicht frei verfügbar und es gibt keine Möglichkeit für Erweiterungen. Für die anderen Werkzeuge gilt ähnliches. Die beste Alternative bietet der Squirrel SQL Client. Eine graphische Darstellung ist nicht verfügbar. Über eine Plugin-Architektur besteht aber die Möglichkeit für Erweiterungen.

Derzeit steht bereits eine Reihe von Erweiterungen zur Verfügung, die für unser Anwendungsfeld von Interesse sind. Das **SQLScript Plugin** gestattet das Laden und Speichern von einzelnen Anfragen und kompletten Skripten. Dies bildet den Ausgangspunkt für die Bereitstellung von Aufgabenstellungen und die Übermittlung von Lösungen zu Übungsaufgaben. Das **JEdit Plugin** erlaubt das Syntax-Highlighting von SQL-Anfragen. So werden beispielsweise Schlüsselworte, Tabellen- und Attributnamen hervorgehoben, was hilfreich bei der Erstellung von Anfragen ist und auch die Präsentation in Vorlesungen unterstützen kann. Die Anpassung der Oberfläche (Fonts, Farben, etc.) ist mittels des **Look and Feel Plugin** möglich. Damit kann der Client leicht an unterschiedliche Umgebungen anpassen, z.B. Präsentation in Vorlesungen (erfordern größere Fonts) oder Übungen. Schließlich bietet das **SQL Validator Plugin** Zugang zu einem Web Service, der die syntaktische Validierung einer SQL-Anfrage ermöglicht. Dies ist insbesondere für Übungen und das Selbststudium von Interesse.

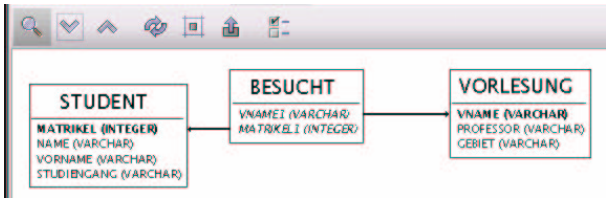
Eine Erweiterung für ein besseres Verständnis von SQL, ist die graphische Visualisierung der Strukturen in der Datenbank. Visualisiert werden sollen insbesondere Tabellen und deren, durch Schlüssel und Fremdschlüssel definierten, Beziehungen, sowie Attribute mit bestimmten Eigenschaften, z.B. Schlüssel. Insbesondere bei der Formulierung von Anfragen mit Joins wird die Bestimmung der Attribute erleichtert.

Da der Squirrel SQL Client derzeit keine Visualisierung ermöglicht, wurde ein erster Prototyp eines entsprechenden Plugins implementiert. Als Basis für die graphische Repräsentation wurde wiederum Jazz gewählt. Die realisierte Visualisierungskomponente, gestattet u.a. auch ein gewisses semantisches Zooming, d.h. je nach Zoomfaktor werden mehr oder weniger Informationen angezeigt. Alternativ kann die Umschaltung auch manuell erfolgen. Insbesondere bei Präsentationen ist es damit möglich, nicht erforderliche Informationen auszublenden. In Abbildung 1 ist dies beispielhaft dargestellt.

(a) Tabellennamen



(b) Tabellennamen und Schlüssel



(c) Alle Informationen

Abbildung 1: Detailierung der Darstellung

Bei der Erstellung des Prototypen haben sich einige Probleme ergeben, z.B. bei der graphischen Darstellung im Zusammenhang mit dem semantischen Zooming. Daher ist eine Neuimplementierung erforderlich, wobei an Stelle von Jazz dessen Nachfolger Piccolo eingesetzt wird. Dieses Framework zielt auf das selbe Anwendungsfeld, ist aber (aufgrund der vereinfachten Klassenstruktur) einfacher zu handhaben. Außerdem zeigte ein erster Test, dass die graphische Darstellung stabiler ist. Eine Funktion, die keines der untersuchten Graphikpakete hinreichend unterstützt ist das automatische Layout der Graphen, sodass dies vorerst nicht weiter betrachtet wird.

Eine weitere Funktion, die bei der Visualisierung geplant war, ist die Beschränkung auf Tabellen, die in einer Anfrage verwendet werden. Damit soll die schrittweise Erstellung von Anfragen und das Verständnis von join-Anfragen unterstützt werden. Theoretisch werden über das JDBC-Interface alle dafür notwendigen Informationen bereitgestellt, d.h. Namen von Tabellen und Attributen, die im Ergebnis der Anfrage erscheinen. Daraus läßt sich ermitteln, welche Tabellen an der Anfrage beteiligt sind und welche Beziehungen bestehen. In der Praxis hat sich allerdings gezeigt, dass viele JDBC-Treiber die notwendigen Methoden nicht implementiert. Der Treiber für Oracle liefert beispielsweise keinen Tabellennamen. Daher konnte diese Funktion nicht implementiert werden.

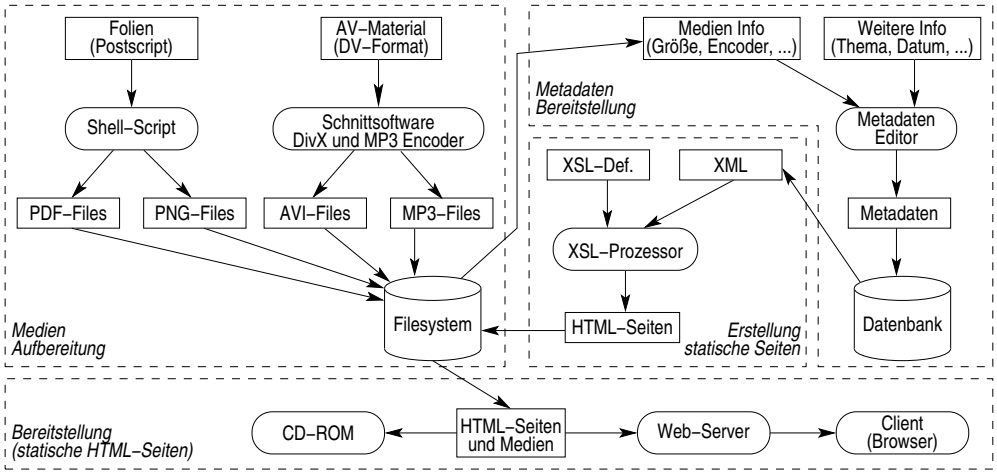


Abbildung 2: Erstellung von Vorlesungsvideos

3.3 Vorlesungsvideos

Im Rahmen des Projektes wurde ein Ansatz entwickelt und getestet, mit dem die Bereitstellung von Vorlesungsvideos mit einer thematischen Strukturierung ermöglicht werden soll. Im folgenden wird die Vorgehensweise (vgl. Abb. 2) zu deren Aufbereitung kurz vorgestellt. Eine genauere Beschreibung erfolgte in [JHS02]. Ziel des Ansatzes ist es, die Videos der einzelnen Vorlesungen in kleine Abschnitten (Szenen) aufgeteilt bereitzustellen. Da die Vorlesung weiterhin basierend auf Folien durchgeführt wird, wurde eine Aufteilung entsprechend dieser Folien vorgenommen. Dadurch wird u.a. die Suche erleichtert und es bieten sich mehr Möglichkeiten der Präsentation. Um den Aufwand für die Erstellung möglichst gering zu halten, war es zunächst erforderlich, geeignete Werkzeuge und Vorgehensweisen zu entwickeln.

Die Aufbereitung gliedert sich in drei wesentliche Schritte. Zunächst wird das aufgezeichnete Video mittels einer Schnittsoftware in die gewünschten Segmente aufgeteilt. Diese werden dann in das DivX- und MP3-Format kodiert. Diese haben sich als besonders geeignet erwiesen, da sie bei den erforderlichen, geringen Datenraten noch eine ausreichende Qualität liefern. Für diese beiden Arbeitsschritte erwiesen sich VirtualDub (`http://www.virtualdub.org/`) unter Windows und transcode (`http://www.theorie.physik.uni-goettingen.de/~ostreich/transcode/`) unter Linux als die geeignetste Lösung. Insbesondere, da diese eine Batchbearbeitung bzw. den Einsatz in Shell-Scripten ermöglichen. Damit lies sich der Anteil manueller Arbeitsschritte wesentlich reduzieren.

Ein weiterer Arbeitsschritt ist die Vorbereitung der im Postscript-Format vorliegenden Folien für die Präsentation im Internet. Diese müssen in einzelne Dateien (eine je Seite) aufgeteilt und konvertiert werden. Zu diesem Zweck wurde ebenfalls ein Shell-Script erstellt, dass diese Aufgabe automatisiert. Neben Folien und Video werden weitere Informationen präsentiert und verwaltet, etwa Verweise auf Bücher, Schlagworte etc. sowie verschiedene Metadaten, z.B. Dateiformate. Schließlich sind noch Informationen für die Generierung der HTML-Seiten erforderlich, z.B. Dateiname der Folie. Die Verwaltung erfolgt derzeit in einer

XML-Datei verwaltet. Für eine spätere Weiterentwicklung ist die Verwaltung in einer Datenbank vorgesehen. Die Eingabe der Metadaten erfolgt derzeit manuell. Teilweise ist aber eine Automatisierung möglich, insbesondere bei den technischen Informationen, wie Dateigröße, Laufzeit etc. Mittels XSL-Script werden, unter Verwendung von Xalan (`http://xml.apache.org/xalan-j/`), die HTML-Seiten automatisch generiert und strukturiert. Derzeit wird die Strukturierung nach Vorlesungsdatum erzeugt. Die Erweiterung um eine thematische Gliederung ist derzeit realisiert.

4 Evaluierung

Das als Grundlage der Vorlesung und Lerneinheit dienende inhaltliche Konzept wird bereits seit mehreren Jahren erfolgreich in den Grundlagenvorlesungen Datenbanken I und Datenmanagement eingesetzt. Dieser Abschnitt beschränkt sich daher auf die Ergebnisse der Evaluierung zum Einsatz der Videomitschnitte.

4.1 Auswertung der Videoaufzeichnung

Im Wintersemester 2001/02 wurde erstmals eine Vorlesung (Datenbanken I) in der zuvor beschriebenen Art aufgezeichnet und bereitgestellt. Die Vorlesung wurde von etwa 200 Studierenden besucht. Im praktischen Einsatz wurde untersucht, wie aufwendig die Erstellung ist und welche technische Voraussetzungen für gegeben sein müssen. Weiterhin sollte evaluiert werden, ob die Videos von den Studierenden angenommen werden.

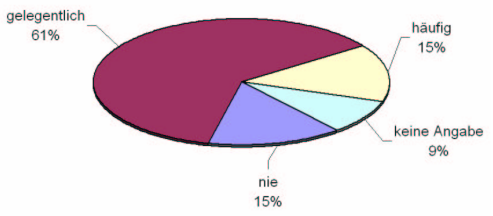
Vom technischen Standpunkt hat sich gezeigt, dass für die Aufnahmen ein normaler digitaler Camcorder hinreichend ist. Eine professionelle Ausrüstung ist nicht erforderlich, da die Qualität durch die spätere Kodierung ohnehin stark reduziert wird. Die Aufbereitung der Videos und insbesondere der Metadaten erwies sich trotz der Automatisierung durch die verschiedenen Skripte als sehr arbeitsintensiv. So wurden für die Aufzeichnung und Aufbereitung einer 90-minütigen Vorlesung etwa 8 bis 10 Stunden benötigt. Bei einer Wiederverwendung ist allerdings mit einem geringeren Aufwand zu rechnen, da insbesondere sie aufwendige Eingabe diverser Metadaten nicht erneut erfolgen muss. Weiterhin stellte sich die Frage, welche Technik für die Bereitstellung erforderlich ist. Es hat sich gezeigt, dass bei der genannten Zahl von Studierenden kein spezieller Server notwendig ist. Die Ablage der etwa 2,5 GB und der monatliche Transfer von etwa 30GB konnte durch einen „normalen“ PC bewältigt werden. Bei einer größeren Anzahl von Studierenden dürfte dieser allerdings bald an Grenzen stoßen. Tabelle 1 fasst einige der wesentlichen Ergebnisse der Evaluierung der technischen Aspekte zusammen.

Tabelle 1: Technische Informationen zur Videoaufzeichnung

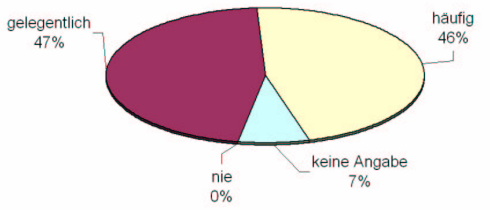
DV-Material einer Vorlesung (90 Minuten)	ca. 20 GB
Gesamtes Material einer Vorlesung (90 Minuten)	ca. 160 MB
Zeit zur Aufzeichnung und Bearbeitung einer Vorlesung	ca. 8–10 Std.
Speicherbedarf aller 14 Vorlesungen	ca. 2,3 GB
Durchschnittlicher monatlicher Download	ca. 30 GB

Eine weitere Auswertung wurde zum Ende des Semesters mittels eines kleinen Fragebogens durchgeführt. Zunächst war dabei von Interesse, ob seitens der Studierenden die technischen Möglichkeiten für den Zugriff auf die Materialien zur Verfügung stehen. Daher wurden zunächst gefragt, welche Art der Netzwerkverbindung bestand. Hier hat die Auswertung ergeben, dass etwa 90% der befragten Studierenden entweder das Uni-Netzwerk nutzten oder eine DSL-Verbindung besaßen, so dass auch die Übertragung größerer Datenmengen kein Problem darstellte. Hinzu kam, dass die Studierenden selbst CDs anfertigten und verbreiteten.

Wünschenswert wäre eine Untersuchung hinsichtlich der Videonutzung auf die Häufigkeit der Vorlesungsbesuche gewesen. Diese lässt sich allerdings nur bedingt durchführen, da keine Anwesenheitslisten geführt werden und auch die Nutzung des Videos nicht eindeutig bestimmt werden kann. Letzteres insbesondere, da die Aufnahmen auch auf CD bereitgestellt bzw. von der Studierenden zunächst vollständig heruntergeladen und abschließend offline betrachtet wurden. Daher beschränkte sich der Fragebogen auf die Aussagen der Studierenden. Für die Nutzung der Videos und die Besuche der Vorlesung wurden vier Möglichkeiten von „nie“ bis „häufig“ bzw. „immer“ angeboten. Weiterhin bestand die Möglichkeit die persönliche Meinung zu äußern. Einige Ergebnisse sind in Abbildung 3 zusammengefasst. Von den Befragten gaben jeweils etwa 50% an, dass sie die Vorlesung häufig bzw. selten besuchten. Aufgrund der zuvor erwähnten Probleme bei der objektiven Bewertung, sind diese Zahlen allerdings nur bedingt aussagekräftig.



(a) Vorlesung häufig besucht



(b) Vorlesung selten besucht

Abbildung 3: Auswertung zur Videonutzung (Wie häufig wurde das Videoangebot genutzt?)

Schließlich wurde gefragt, welche Art der Bereitstellung (kleine Szenen oder ganze Vor-

lesung) bevorzugt und ob das Angebot als hilfreich angesehen wurde. Hier zeigte sich, dass etwa 60% der befragten die thematische Gliederung bevorzugten. Die überwiegende Zahl der Befragten (etwa 70%) sah das Angebot als hilfreich an.

Zusammenfassend lässt sich aus der Befragung ableiten, dass die Ergänzung und Bereitstellung der Folien mit Video- und Audiomitschnitten in der beschriebenen Form von den Studierenden überwiegend positiv aufgenommen und auch. Erwartungsgemäß wurde das Video häufiger genutzt, wenn die Vorlesung selten besucht wurde. Inwieweit das Videoangebot und der Vorlesungstermin (Freitag 7:30) diese Ergebnisse beeinflusst lässt sich derzeit allerdings nicht feststellen. Um brauchbare Aussagen zu erhalten müsste die Vorlesung mehrfach mit dem Video angeboten werden.

Derzeit (Wintersemester 2002/03) wird eine weitere Vorlesung (Multimediadatenbanken) aufgezeichnet und bereitgestellt. Der Aufwand für die Erstellung konnte dabei weiter reduziert werden, da beispielsweise der Schnitt und die Kodierung durch den Einsatz anderer Software optimiert wurde.

4.2 Evaluierung der Werkzeuge

Durch die in den Abschnitten 3.1 und 3.2 beschriebenen Probleme bei der Implementierung der Werkzeuge konnten diese bisher nicht in der Praxis eingesetzt und evaluiert werden. Diese wird frühestens im kommenden Sommersemester 2003 möglich sein, wenn die Komponenten reimplementiert wurden. Dabei wird vorrangig das SQL-Tool bearbeitet. Für die Evaluierung soll das Werkzeug dann beispielsweise in Übungen oder einem Praktikum eingesetzt werden. Abschließend erfolgt wiederum eine Befragung der Studierenden.

5 Zusammenfassung und Planung für 2003

Im zweiten Projektjahr lag der Schwerpunkt der Arbeiten bei der Realisierung von Werkzeugen, mit denen die Vorlesung unterstützt werden soll. Insbesondere wurden die Implementierung eines graphischen Editors für das ER-Modell und ein Werkzeug für die Arbeit mit SQL bearbeitet. Während der Analyse hat sich dabei gezeigt, dass komplexe (kommerzielle) Werkzeuge für den Einsatz in Vorlesungen, z.B. im Bereich der Modellierung, nur bedingt geeignet sind. Daher wurden eigene Implementierungen vorgezogen. Für das Gebiet der Anfragesprachen bot andererseits die Erweiterung eines bestehenden Werkzeugs (Squirrel SQL Client) an, da dieses bereits gute Möglichkeiten für Präsentation bot und nur geringfügige Anpassungen notwendig waren. Eine Erweiterung, die derzeit implementiert wird, soll die Visualisierung der Tabellen in einer Datenbank und deren Beziehungen ermöglichen. Erfahrungsgemäß fördert eine derartige Darstellung das Verständnis bei der Definition von Anfragen in SQL. Durch die Möglichkeit, vorbereitete Anfragen zu laden und zu editieren bietet sich dieses Werkzeug auch für den Einsatz in Vorlesungen und Seminaren an.

Weiterhin wurde der Ansatz zur Bereitstellung von Vorlesungsvideos weiterentwickelt und beim Einsatz im Wintersemester 2001/02 evaluiert. Eine Befragung der Studierenden hat dabei gezeigt, dass die Videos vorwiegend positiv bewertet wurden. Auch der Ansatz diese als relativ kurze Szenen bereitzustellen wurde vorwiegend positiv bewertet. Es hat sich aber auch gezeigt, dass diese Art der Bereitstellung mit einem erheblichen Arbeitaufwand verbunden ist.

Für das kommende Jahr 2003 ist vorwiegend die Vervollständigung der Werkzeuge geplant. Zum einen wird das Visualisierungs-Plugin für den Squirrel SQL Client in einer ersten nutzbaren Version fertiggestellt. Weiterhin ist geplant, Videos für die Einführung in die Anwendung des Werkzeugs und die Einführung in SQL zu erstellen und Beispiele-Skripte für den Einsatz in Squirrel bereitzustellen. Diese sollen dann die Folien ergänzen.

Für die Motivation der Vorlesung ist die Mitarbeit an der Erstellung eines Videos mit realen Szenen geplant. Dieses Video soll an einem konkreten Beispiel darstellen, welche Daten beispielsweise in einem Versandhaus anfallen und verwaltet werden müssen. Das Video könnte etwa die manuelle Aufnahme von Bestellungen durch einen Mitarbeiter und deren Weiterleitung an das Lager darstellen. Auch eine Beschreibung der Datenverwaltung in einem Lager ist denkbar. Daran könnten dann die für die Motivation der Vorlesung wesentlichen Aspekte, z.B. Probleme redundanter Datenhaltung erläutert werden.

Schließlich sollen die Werkzeuge für die Erstellung der Vorlesungsvideos abschließend abgepasst werden. Dazu ist u.a. eine Anpassung der Struktur der Metadaten erforderlich, die derzeit durchgeführt wird. Diese ist erforderlich, um neben der derzeit verfolgten Bereitstellung nach Datum der Vorlesung auch eine rein thematische Gliederung zu ermöglichen. Die Umstrukturierung ist weiterhin erforderlich, um die Materialien im MuSofT-Portal zur Verfügung zu stellen. Geplant ist, die Folien, die WWW-Seiten und die Werkzeuge für die Videoerstellung im Portal bereitzustellen. Weiterhin werden auch das SQL-Tool und die erstellten Beispiele bereitgestellt, sobald die Implementierung abgeschlossen ist.

Literatur

[Bar00] BARDOHL, ROSWITHA: *GenGEd: Visual Definition of Visual Languages based on Algebraic Graph Transformation*. Verlag Dr. Kovač, 2000. (Dissertation, Technische Universität Berlin, FB Informatik).

[BMG00] BEDERSON, BENJAMIN B., JON MEYER, and LANCE GOOD: *Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java*. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Toolkit Support for UI, pages 171–180, 2000.

[HS00] HEUER, ANDREAS und GUNTER SAAKE: *Datenbanken: Konzepte und Sprachen*. Informatik Lehrbuch-Reihe. MITP-Verlag, Bonn, 2. Auflage Auflage, 2000.

[Jes02] JESKO, DIRK: *Zwischenbericht zu LE1.2 – Entwicklung von Informationssystemen*. In: DOBERKAT, ERNST-ERICH und GREGOR ENGELS (Herausgeber): *Ergebnisbericht des Jahres 2001 des Projektes MuSofT – Multimedia in der SoftwareTechnik*, Nummer MEMO Nr. 121 in *Internes Memorandum des Lehrstuhls für Software-Technologie*, Seiten 16 – 26. Universität Dortmund, Februar 2002.

[JHS02] JESKO, DIRK, THOMAS HERSTEL, and INGO SCHMITT: *Enhancing a Course by Multimedia Objects*. In POPOV, ANGEL (editor): *Proceedings of* 16^{th} *International Conference on Systems for Automation of Engineering and Research (SAER-2002)*,

Varna, Bulgaria, , volume Radi Romansky, pages 186–190. SAER Forum Group, 2002.

[MK99] MINAS, MARK and OLIVER KÖTH: *Generating diagram editors with DiaGen*. In NAGL, MANFRED, ANDY SCHÜRR, and MANFRED MÜNCH (editors): *Proceedings: Applications of Graph Transformations With Industrial Relevance: International Workshop and Symposium AGTIVE'99, Kerkrade, The Netherlands, September 1-3*, volume 1779 of *Lecture notes in computer science*, pages 433–440, Berlin, Heidelberg, 1999. Springer-Verlag Inc.

Der MuSofT-Jahresbericht 2002 Lerneinheiten 1.3 und 2.4

Olaf Scheel, Johannes Magenheim

Jahresabschlussbericht 2002 für die Lerneinheit 1.3 und die Lerneinheit 2.4.

Inhaltsverzeichnis

1 Einleitung

Dies ist der Jahresabschlussbericht 2002 für die Lerneinheit 1.3: *Softwareengineering in der Informatiklehrerausbildung* und Lerneinheit 2.4: *Dekonstruktion von Softwaresystemen*. Beide Lerneinheiten sollen innerhalb der Informatiklehrer-Ausbildung an der Universität Paderborn eingesetzt werden, Teilbereiche sind bereits eingesetzt worden. Darüber hinaus sollen die einzelnen Lernobjekte, Module und Einheiten von anderen Teilprojekten des MuSofT-Konsortiums nutzbar sein. Im Wesentlichen gelten dabei noch immer die Modulbeschreibungen aus dem Projektantrag des Jahres 2000 und die Angaben aus dem Zwischenbericht des Jahres 2001. Dieser Bericht möchte weitgehend nur die Änderungen und Fortschritte innerhalb des Jahres 2002 darlegen.

2 Vorstellung der Lerneinheit und Didaktisches Konzept

2.1 Lerneinheit 1.3

In der Lerneinheit *Softwareengineering in der Informatiklehrerausbildung* soll den Studierenden anhand der lerneinheit-übergreifenden Fallstudie *Hochregallager* das Konzept des *soziotechnischen Informatiksystems* und damit zusammenhängend der *Systemorientierten Didaktik* vermittelt werden. Die einzelnen Lernmodule beschäftigen sich also zunächst mit diesen beiden Konzepten, greifen aber darüber hinaus Methoden der professionellen Software-Technik auf und untersuchen sie in (schul-)unterrichtlichen Zusammenhängen. Dabei werden auch typische Berufsbilder in der Informatik behandelt. Die Materialien bestehen zum einen aus Texten zu den oben genannten Konzepten, aus Materialien zur Fallstudie des Hochregallager, die dekonstruktiv erarbeitet werden oder als Beispiele eingesetzt werden können und aus Arbeitsaufträgen rund um das Szenario einer automatengesteuerten Packstraße, in dem die Studierenden ihr erworbenes Wissen im Sinne des blended learning konstruktiv einsetzen können.

Konkret beschäftigt sich das Modul 1 *Soziotechnisches Informatiksystem* mit dem vorgenannten Konzept - ausgehend von Informatiksystemen und ihrer Bedeutung für die Fachwissenschaft und -didaktik. Ausgehend davon wird dann der Begriff des STIS eingeführt, dass man verkürzt als Informatiksystem mit seinem technischen und sozialem Umfeld charakterisieren kann. Die Fallstudie Hochregallager dient dabei als Beispiel für ein solches System. Das nachfolgende Modul 2 *Systemorientierte Didaktik* beschäftigt sich mit den Konsequenzen für die Fachdidaktik, wenn sie sich nach dem Konzept des STIS ausrichtet: Unterrichtsinhalte und Methoden. Beispiele für dies beiden Teilaspekte werden wieder aus der Fallstudie der Lerneinheit gewählt. Die beiden anschließenden Module beschäftigen sich nun näher mit diesen Methoden. In Modul 3 aus einer fachwissenschaftlichen Perspektive, in Modul 4 aus einer fachdidaktischen. Innerhalb dieses Modul wird ein mögliches Vorgehensmodell für den Informatikunterricht entwickelt, das mit Beispielen und Aufgaben mit der Fallstudie verknüpft wird. Das letzte Modul der Lerneinheit beschäftigt sich dann mit Rollen und Rollenklischees während eines Entwicklungsprozess. Hier sollen Videos von Tätigkeiten verwendet werden. Soweit möglich sollen auch Video-Dokumente eingesetzt werden, die während des nächsten Sommersemester aufgezeichnet werden sollen. Das Modul hat somit Schnittstellen sowohl zum Modul 3, als auch zum Modul 1 dieser Lerneinheit.

Die multimedialen Bausteine bzw. Lernobjekte der Lerneinheit werden im Wesentlichen durch die Fallstudie des Hochregallagers bestimmt. Diese werden so aufbereitet, dass sie das Wechselspiel auf Dekonstruktion und Konstruktion widerspiegeln, wie in Abbildung 1 beschrieben. Der Teilaspekt Dekonstruktion wird durch das in der Arbeitsgruppe implementierte Lego-Hochregallager realisiert, der Aspekt der Konstruktion durch eine computergesteuerte Packstraße, ebenfalls in Lego, deren reales Vorbild ebenfalls aus dem Bereich der Warenlogistik (Kommissionierung) stammt. Für diesen Teil werden von uns aber nur die grundlegende Lego-Hardware und Videos des Vorbildes bereitgestellt. Alles andere soll von den Studierenden selbst implementiert werden.

2.2 Lerneinheit 2.4

Die Lerneinheit *Dekonstruktion von Softwaresystemen* beschäftigt sich mit einzelnen Teilaspekten des didaktischen Ansatzes der Dekonstruktion von Softwaresystemen und soziotechnischen Informatiksystemen. Als Fallstudie wird hier ein *Schulkiosk/Warenwirtschaftsystem* verwendet. Die einzelnen Module enthalten Lernobjekte zu den damit verbundenen (Analyse-)Methoden, Sichtweisen auf Informatiksysteme, aber auch zum Gebrauch von Werkzeugen. Während in der Lerneinheit 1.3 parallel zur Fallstudie Arbeitsaufträge zu einem neuen Szenario von den Studierenden bearbeitet werden können, soll in den praktischen Phasen dieser Lerneinheit eine Erweiterung des bestehenden Systems zu einem Online-Shop geleistet werden und die sich daraus hinsichtlich der Lerninhalte der Einheit ergebenen Konsequenzen vermittelt werden. Somit steht hier weniger das Wechselspiel zwischen Dekonstruktion und Konstruktion im Vordergrund, wie in Lerneinheit 1.3, sondern der Gedanke des Reengineering. Da die von uns implementierte Schulkiosk-Software in Java nach dem MVC-Muster entworfen wurde, ist dies von den Studierenden nach einer Erkundungsphase mit didaktischen Fenstern in das Informatiksystem zu leisten.

2.3 Didaktisches Konzept

Das didaktische Konzept für die Lerneinheiten 1.3 und 2.4 ist geprägt durch den Begriff der *Lernwerkstatt Informatik*. Hierunter soll verstanden werden, dass die innerhalb des MuSofT-Projektes für die Präsenzlehre entwickelten Materialien nicht nur im Präsenzteil (im engeren Sinne) der Veranstaltungen genutzt werden sollen, sondern auch bei der Nachbereitung durch die Studierenden und für Übungen. Die Materialien sollen dabei für einen nach konstruktivistischen Ideen ausgerichteten Lernprozess geeignet sein und nicht ausschließlich für dozentenzentrierte Phasen der Präsenzveranstaltung. Schlüsselbegriffe sind hier: blended learning, Erkundung von didaktische Szenarien, kooperatives Lernen in Lerngruppen und didaktische Fenster. Was darunter zu verstehen ist, soll im weiteren Verlauf unter *Technischer Realisierung* näher erläutert werden. Die für die Lerneinheiten erstellten Materialien werden auf den am Institut vorhandenen Steam-Server (http://steam.upb.de/) als Lernplattform eingestellt werden und somit für die Studierenden nutzbar.

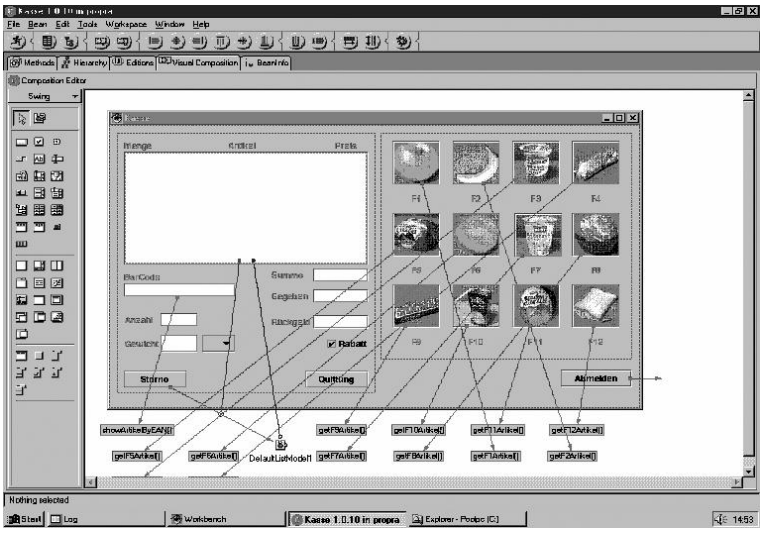


Abbildung 1: Schulkiosk

3 Technische Realisierung

3.1 Lernwerkstatt Informatik

Das MuSofT-Projekt soll Präsenzlehre durch multimediale Lernmodule unterstützen und aufwerten. Aus diesem Grund wurden innerhalb des Teilprojektes keine Materialien für reine Online-Kurse ohne Präsenzphasen entwickelt. Stattdessen wurden Lernobjekte erstellt, die für den gezielten punktuellen Einsatz in Präsenzveranstaltungen wie Vorlesungen oder auch Seminaren geeignet sind, aber auch für die Nacharbeitung der Studierenden, für Selbstlernphasen, Übungen und durch Groupware unterstützte Präsenzseminare, in denen netzgestützt kooperativ gearbeitet wird. Unterstützt werden sollen also unterschiedliche Phasen des blended learning mit unterschiedlich starker Aktivität der Studierenden und des Dozenten. - All dies zusammengefasst unter dem Schlagwort der *Lernwerkstatt*, in die zu einem späteren Zeitpunkt auch Ergebnisse anderer Projekte der AG Didaktik der Informatik Paderborn einfließen sollen. Zum jetzigen Zeitpunkt soll sie die innerhalb des MuSofT-Projektes inhaltlich abgeschlossene Lerneinheiten aus wieder verwendbaren Bausteinen enthalten.

Das didaktische Konzept der Lernwerkstatt beruht dabei aus Sicht der Studierenden auf vier Säulen:

- Unterweisung durch den Dozenten unter exemplarischer Benutzung der für die Fallstudie erstellten Materialien
- Selbstständige Erkundung der Materialien mit Hilfe einer Lernplattform
- Praktische Anwendung des Gelernten auf ein neues oder ein erweitertes System

- Kooperatives Lernen und Arbeiten in einer Community

Daraus ergeben sich folgende Bausteine der Lernwerkstatt für die Lerneinheiten 1.3 *Softwareengineering in der Informatiklehrerausbildung* und 2.4 *Dekonstruktion von Softwaresystemen*:

- Fachwissenschaft Informatik
- (CASE-)Tools
- Das CMS zur Distribution der Materialien
- Didaktik der Informatik
- Unterrichtsmaterialien
- Unterrichtsmethoden
- CSCL-Portfolio

Beispielhaft wird ein Nutzungsszenario für die bereitgestellten Materialien dann so aussehen: Die Studierenden erhalten durch den Dozenten eine kurze Einführung in das neue Thema und einen Erkundungsauftrag unter Einbeziehung der Fallstudie. Die Ergebnisse werden im Plenum besprochen, gefestigt und gesichert. Daran schließt sich eine praktische Phase an, in der das neu erworbene Wissen auf eine neue Situation angewendet - also transferiert - werden soll. Diese praktische Phase kann in Gruppen oder im Plenum stattfinden. (Oder in Einzelarbeit mit der Möglichkeit zur netzgestützten Kooperation.) Abschließend werden die Ergebnisse gesammelt und bewertet. Somit wechseln sich dekonstruktive, erkundende Phasen immer mit konstruktiven, praktischen Phasen ab. [siehe Abbildung 2] Das gerade genannte Beispiel könnte somit auch eine guided tour durch die Materialien beschreiben.

Ein zweites wichtiges Leitprinzip während der erkundenden Phasen ist das der *Sichten auf Software*. Damit ist nicht nur die unterschiedlichen Sichtweisen von am Entwicklungsprozess beteiligten Personen gemeint, sondern auch die des sozio-technischen Umfeldes. Man kann darüber hinaus noch zwischen verschiedenen Sichtweisen unterschiedlicher Abstraktionsniveaus unterscheiden.

3.2 Lerneinheit 1.3

Die praktische Umsetzung des oben genauer beschriebenen Konzeptes der Lernwerkstatt Informatik sieht in der Lerneinheit 1.3 wie folgt aus:

- Aufarbeitung theoretischer Hintergrundtexte. (Textzusammenfassungen, Folien, Arbeitsaufträge - siehe Jahresbericht 2001)
- Entwicklung eines Lego-Modells des Bertelsmann-Hochregallagers (mit angrenzenden Systemen) bestehend aus zwei Regalen mit Bediengeräten, Übergabestationen und autonomen Transportfahrzeugen.
- Entwicklung der Steuerungssoftware des Lego-Modells in Java

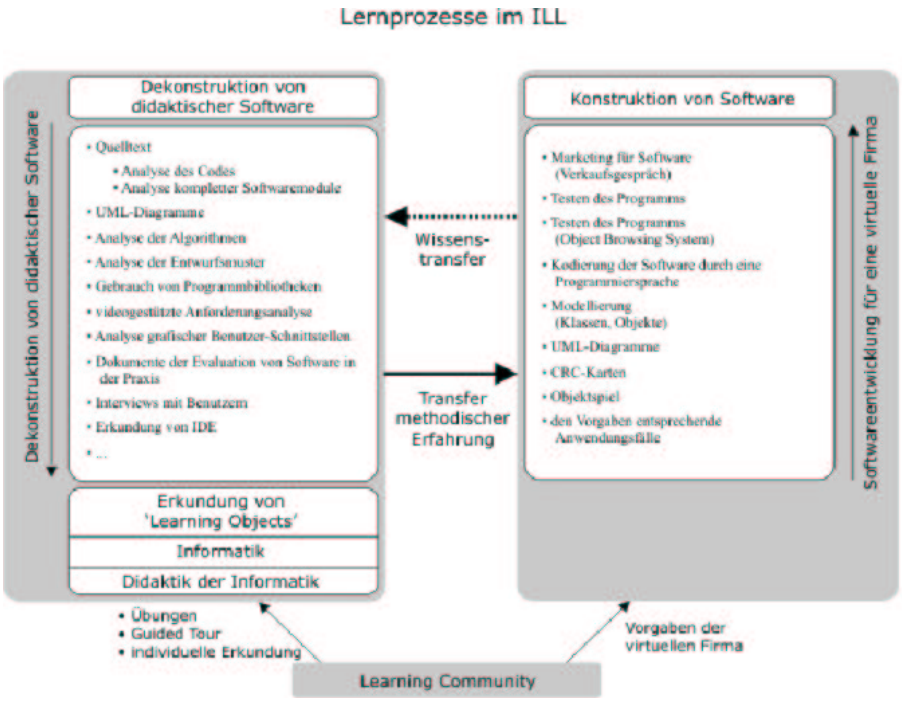


Abbildung 2: Dekonstruktion von Informatiksystemen

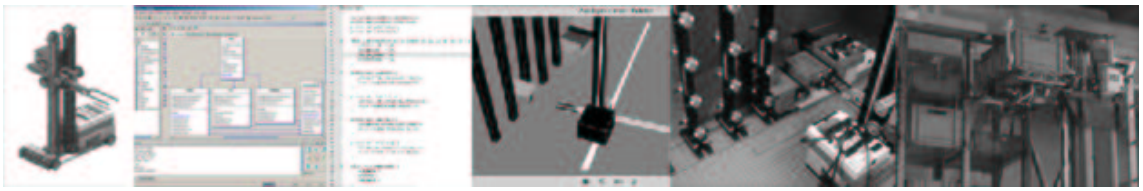


Abbildung 3: Elemente der Fallstudie Hochregallager

- Entwicklung eines PlugIns für das Together Control Center zur (einfacheren) Programmierung und Modellierung des Lego-RCX in Java und UML
- Sammlung von Dokumenten zum Entwicklungsprozess (CRC-Karten, Klassendiagramme, Use Cases)
- Sammlung von Dokumenten zum Thema embedded systems und robotics
- Beginn der Entwicklung einer Simulationsumgebung für das Lego-Hochregallager
- Begonnene Drehbuchentwicklung für Videoaufnahmen im Bertelsmann-Hochregallager
- Gestaltung verschiedener Flash-Animationen zum Thema Hochregallager (real und Lego-Modell - Kommunikationsschema, Ein-/Auslagern, Steuerung der Transportfahrzeuge...)
- Sammlung von Aufgaben für die konstruktiven Phasen (automatengesteuerte Packstraße)
- Planung erster *guided tours* durch die Materialien

3.3 Lerneinheit 2.4

Bisher lag der Schwerpunkt des Teilprojektes auf der Lerneinheit 1.3. Dies wird sich aber innerhalb des ersten Quartals 2003 ändern. Bisher wurden für diese Lerneinheit erstellt:

- Aufarbeitung von Hintergrundtexten wie oben
- Anpassung der Fallstudie Warenwirtschaftssystem/Schulkiosk
- Didaktische Fenster für die Erkundung der Software
- Sammlung erster Dokumente zum Entwicklungsprozess
- Erste Aufgaben für die konstruktiven Phasen (Erweiterung zum Online-Shop)

4 Evaluierung

Bisher sind nur Teile der Fallstudien *Hochregallager* und *Schulkiosk* in den Veranstaltungen *Didaktik der Informatik* und *Grundlagen der Informatik für Lehramtsstudierende* zum Einsatz gekommen. Auch das Konzept der Lernwerkstatt Informatik ist dort vorgestellt und diskutiert worden. Aufgrund des Auslaufens einer Landesstelle der Arbeitsgruppe und damit verbunden die (leider) notwendige Einschränkung unseres Lehrangebotes, ist es noch nicht zu einem Einsatz größerer Teile der Materialien gekommen. (Das eigentlich dafür eingeplante Seminar *Modellieren im Informatikunterricht* konnte nicht mehr stattfinden.)

Ein umfangreicherer Einsatz der Materialien ist nun für das Sommersemester 2003 geplant. Hier werden von unserer Arbeitsgruppe zwei Seminare mit Inhalten und Material des MuSofT-Projektes veranstaltet. (*Lernwerkstatt Informatik* und *eLearning- Plattformen*) Hier soll dann

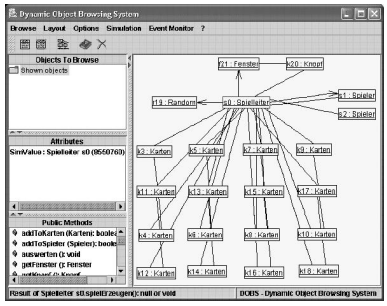


Abbildung 4: Erkundung des Schulkiosk mit DOBS

semesterbegleitend die Evaluation der Materialien stattfinden. Ein umfangreicherer Eingangs- und ein Ausgangstest soll das (Vor-)Wissen zu den Seminar- Inhalten prüfen, aber auch Einstellungen und Vorstellungen zum Einsatz multimedialer Lernbausteine und der verwendeten Unterrichtsmethodik. Darüber hinaus werden die eingesetzten Materialien einer informellen Evaluation unterliegen und semesterbegleitend verbessert werden. Ein weiterer Schwerpunkt wird auf der Evaluation der guided tours durch die Materialien liegen. *Sind die Lernpfade angemessen? Wurden die didaktischen Fenster angemessen ausgewählt?*

Die Evaluation soll dabei vor allem der Verbesserung der Materialien dienen und weniger einer umfangreichen quantitativen Analyse, da bei der zu erwartenden Gruppenstärke von ca. 10-15 Personen kaum allgemeingültige Aussagen getroffen werden können. Es wird aber versucht werden, die fehlende Allgemeingültigkeit durch eine entsprechende Tiefe bei der Abschlussbefragung (Einzelinterviews) zu kompensieren. Der Fragebogen wird sich dabei auch den geschlechtsspezifischen Aspekten und Rollenklischees widmen. Falls eine günstige Verteilung männlicher und weiblicher Studierender im Seminar vorliegen sollte, wird darüber hinaus eine geschlechtsspezifische Evaluation durchgeführt werden.

5 Planung für 2003

5.1 1. Quartal

- Planung und Durchführung der Videodrehs im Hochregallager Bertelsmann
- Vorbereitung der Lerneinheit 1.3 für das Sommersemester 2003
- Fertigstellung der Fallstudie Hochregallager
- Distribution ausgewählter Lernobjekte über die MuSofT-Plattform
- Fertigstellung der Hintergrundmaterialien für Lerneinheit 2.4 (ohne Fallstudie)
- Erste Animationen für die Fallstudie Schulkiosk

5.2 2. Quartal

- Einsatz der Lerneinheit 1.3 und Teile der Lerneinheit 2.4 in zwei Seminaren
- Semesterbegleitende Evaluation
- Verbesserung der guided tours, Ergänzung der Materialien (zum Teil auch durch Material der Studierenden
- Materialerstellung zur Fallstudie Schulkiosk/Warenwirtschaftssystem
- Fertigstellung eines CRC-Tools in Flash

5.3 3. Quartal

- Auswertung der Evaluation, Überarbeitung der Materialien
- Distribution der Lernobjekte über die MuSofT-Plattform
- Integration von Objekten anderer Teilprojekte
- Fertigstellung der Simulationsumgebung für das Lego-Hochregallager
- Fertigstellung der Lerneinheit 2.4 inklusive Fallstudie (bis auf Videos)

5.4 4. Quartal

- Videodrehs für die Fallstudie Warenwirtschaftssystem
- Fertigstellung eines CRC-Karten-Tools für Together
- Einsatz beider Lerneinheit im Wintersemester 2003/2004

6 Zusammenfassung

Die Aufbereitung der Inhalte der einzelnen Lerneinheiten ist zum jetzigen Zeitpunkt weitgehend abgeschlossen. Etwas hinterher hinkt aber die Arbeit an den Fallstudien (besonders der Videoproduktion), der Einsatz der Materialien und damit verbunden der Evaluation und der Distribution über die MuSofT-Plattform. Trotzdem bietet der oben entwickelte Zeitplan für die Fertigstellung der Lerneinheiten eine realistische Perspektive für einen positiven Abschluss des Projektes. Mit der Entwicklung des Konzeptes *Lernwerkstatt Informatik* ist darüber hinaus ein Szenario für die nachhaltige Weiterentwicklung und den Einsatz der Materialien des Projektes entwickelt worden. Durch die Anknüpfungspunkte an die Fachwissenschaft Informatik ist auch bei einem weiteren Ausbau der Lernwerkstatt nicht der Sinn der MuSofT-Distributionsplattform über das Ende der Projektlaufzeit hinaus in Frage gestellt.

Lerneinheit 2.1 – Software-Architektur

Jörg Pleumann

Dieser Bericht gibt den aktuellen Stand der MuSofT-Lerneinheit 2.1 wieder, die sich mit dem Thema „Software-Architektur“ beschäftigt. Er fasst die Ergebnisse des zweiten Projektjahres sowie die weitere Vorgehensweise für das abschließende Jahr zusammen.

Inhaltsverzeichnis

1 Einleitung

Dieser Bericht gibt den aktuellen Stand der MuSofT-Lerneinheit 2.1 wieder, die sich mit dem Thema „Software-Architektur“ beschäftigt. Nachdem das erste Projektjahr 2001 [Ple01] weitgehend von einer Einarbeitung in das Thema und der Konzeption der Lerneinheit geprägt war, ging es im zweiten Projektjahr 2002 wesentlich darum, Entscheidungen bezüglich der technischen Basis zu treffen und die – stärker als andere MuSofT-Teilprojekte – implementierungslastige Lerneinheit umzusetzen.

Ausgangspunkt der Lerneinheit ist die Überlegung, dass die Architektur von Software-Systemen ein sehr praxisorientiertes Thema innerhalb der Softwaretechnik darstellt und auf eine entsprechende Weise gelehrt werden sollte. Die eher theoretische Vermittlung der Eigenschaften bestimmter Architekturen oder architektureller Stile in Vorlesungen befähigt die Studierenden nicht automatisch zum Entwurf komplexer Software-Systeme – genau wie das Wissen um *if*-, *for*- und *while*-Konstrukte sie nicht dazu befähigt, strukturiert zu programmieren. In beiden Fällen ist das praktische Einüben des vermittelten Wissens nötig.

Die Lerneinheit soll den Studierenden helfen, die gewünschten praktischen Erfahrungen mit Architekturen zu sammeln. Dazu stellt sie als Kernelement ein spezielles Werkzeug bereit, das zum Bearbeiten von Software-Architekturen mit graphischen Mitteln dient – quasi einen Architektur-Editor, der in besonderem Maße für die Lehre in der Softwaretechnik geeignet ist. Der Editor kann gleichermaßen in Übungen und Praktika Verwendung finden und bietet die nötige Infrastruktur, den Studierenden Aufgaben im Bereich des Modellierens von Software-Architekturen zu stellen, die anschließend durch einen menschlichen Tutor bewertet werden können. Zudem können die graphischen Möglichkeiten des Editors zur komfortablen und übersichtlichen Visualisierung von Architekturen genutzt werden, so dass das Werkzeug auch im Vorlesungsbetrieb einsetzbar ist und dort Tafelanschrieb oder Overhead-Folien ergänzen oder gar ersetzen kann.

2 Vorstellung der Lerneinheit

Die Lerneinheit vermittelt Grundlagen- und – soweit möglich – Erfahrungswissen zu Software-Architekturen. Im wesentlichen geschieht dies, indem die Möglichkeiten des graphischen Editors von Dozenten zur Visualisierung oder von Studierenden zur praktischen Arbeit eingesetzt werden. Zur Abrundung und um den Zugang zur und die Navigation innerhalb der Lerneinheit zu erleichtern, besitzt diese jedoch auch eine gewisse Menge an (hyper-) textuellem Lehrstoff. Dieser ist wie folgt strukturiert:

Grundlagen Enthält Motivation sowie Grundbegriffe und -ideen aus dem Bereich der Software-Architektur und soll den Studierenden in komprimierter Form als eine Art Referenz oder „Online-Hilfe“ zur Verfügung stehen. Entspricht in etwa den einleitenden Kapiteln von [BMR+96, SG95] oder [GS94, HHK+96, PW92, SW99].

Notation Führt die Grundelemente einer Architekturbeschreibung und die zugehörige graphische Notation innerhalb des Editors ein. Letztere basiert auf Ideen aus Real-Time Object-Oriented Modeling (ROOM) [SGW94] bzw. einer an die Unified Modeling Language (UML) angelehnten Variante UML for Real-Time (UML-RT) [SR03, Lyo03] und bedient sich im wesentlichen eines objektorientierten (Meta-) Modells von Architektur-Komponenten, deren Verhalten mit State Charts, einer speziellen Form von Zustandsübergangsdiagrammen, beschrieben wird.

Architekturstile Erläutert den Sinn von Architekturstilen und stellt einige der bekannteren Stile vor. Gleicht den üblichen Katalogen von Entwurfsmustern [BMR+96, GHJV96], bewegt sich aber auf einer höheren Abstraktionsebene.

Fallstudien Stellt ausgewählte Architekturen in Fallstudien vor, um den Studierenden ein Gefühl für die Vor- und Nachteile der im vorangehenden Abschnitt eingeführten Architekturstile bei konkreten Problemen zu vermitteln.

Übungen und Praktikum Enthält verschiedene Aufgaben, die von den Studierenden bearbeitet werden können. Tatsächlich wird dies kein eigener Teilbereich der Lerneinheit sein. Stattdessen werden die Aufgaben in die entsprechenden anderen Bereiche integriert.

Zwischen Editor und Hypertext existiert eine Verbindung derart, dass Aktionen des Benutzers in der einen Komponente Reaktionen der anderen Komponente zur Folge haben können. So kann etwa beim Anklicken eines Hyperlinks im Text automatisch eine neue Architekturskizze im Editor angezeigt werden, oder Elemente einer bestehenden Architektur werden geändert.

Die Lerneinheit richtet sich an Haupt- oder Nebenfachinformatiker, die – vermutlich durch eine Softwaretechnik-Vorlesung oder eine entsprechende Spezialvorlesung – ein Interesse an Software-Architekturen haben. Sie soll aber nicht auf Universitäts- oder FH-Studierende festgelegt sein, sondern zumindestens theoretisch auch innerhalb der beruflichen oder privaten Weiterbildung Verwendung finden können.

Programmierkenntnisse werden vorausgesetzt bzw. sind wünschenswert, wenngleich Studierende innerhalb der Lerneinheit nicht mit Quellcode konfrontiert werden. Die Beschreibung struktureller und dynamischer Eigenschaften von Software-Systemen findet ausschließlich mit den abstrakten Mitteln der ROOM/UML-basierten Notation statt. Es ist jedoch nicht anzunehmen, dass ein Studierender, dessen Programmierpraxis sich ausschließlich auf die in Übungsaufgaben üblichen Mehrzeiler erstreckt, ein Gespür für die Probleme besitzt, die bei der Realisierung größerer oder langlebiger Software-Systeme auftreten. Damit wird ihm die Motivation für die Beschäftigung mit Software-Architektur fehlen.

Das didaktische Konzept der Lerneinheit liegt nicht völlig fest – es kann vom Dozenten durch die Art und Weise, wie der Architektur-Editor eingesetzt wird, variiert werden. Das „Komplett-Angebot“ aus Editor, vordefinierten Architekturen, einführendem Material und Aufgaben lässt sich jedoch durch die Begriffe „fallbasiertes Lernen“ (da Architekturen an Fallbeispielen verdeutlicht werden) und „oszillierend zwischen konstruktiven und dekonstruktiven Phasen“ (da Architektur-Beschreibungen in den Übungen sowohl erstellt als auch analysiert werden sollen) charakterisieren. Durch die Verknüpfung von Editor und Hypertext-Material ist auch in Grenzen „entdeckendes Lernen“ realisierbar.

3 Technische Realisierung

Den Absprachen innerhalb des Projektes folgend geschieht die technische Umsetzung der Lernumgebung in Java. Dies garantiert eine Einsetzbarkeit auf einer maximalen Zahl von Plattformen und kommt damit der Nachhaltigkeit des Projektes zugute. Die Lernumgebung setzt sich, wie bereits im Jahresbericht 2001 beschrieben, aus mehreren Hauptkomponenten zusammen:

- Graphischer Editor
- Hypertext-System
- Simulation
- Animation/Scriptsprache

Diese Komponenten werden zu einer geschlossenen Anwendung verbunden, die dann die eigentliche Lerneinheit darstellt.

3.1 Graphischer Editor

Der Editor dient zum Visualisieren und Bearbeiten von Software-Architekturen. Je nach Betrachtungsweise handelt es sich dabei um ein einfaches, auf architekturelle Bedürfnisse zugeschnittenes CASE-Tool oder ein 2D-Vektorgrafik-Programm, bei dem komplexere Elemente (Kapseln, Konnektoren etc.) definiert und auf der Basis vorgegebener Anforderungen oder Regeln (Architektur-Stile) verbunden werden können.

Der Editor ist der Kern der Lerneinheit, und er stellt sicherlich den aufwändigsten Teil der gesamten Anwendung dar. Um die Eigenimplementierung hier in Grenzen zu halten, wurden für diesen Teil bereits frühzeitig verschiedene Komplett-Werkzeuge und Klassenbibliotheken auf ihre Eignung als technische Basis hin evaluiert:

- ArgoUML (`http://argouml.tigris.org`)
- Together (`http://www.togethersoft.com`)
- Y-Files (`http://www.yworks.com`)
- JHotDraw (`http://jhotdraw.sourceforge.net`)
- Jazz (`http://www.cs.umd.edu/projects/hcil/jazz`)

Im Rahmen der Evaluation fand unter anderem im Frühjahr 2002 ein Projekt-interner Workshop statt, an dem eine Reihe von MuSofT-Teilprojekten mit ähnlichen technischen Anforderungen teilnahmen. Hier wurden Erfahrungen und Evaluationsergebnisse ausgetauscht. Für Teilprojekt 2.1 schieden ArgoUML und Together bereits frühzeitig als zu umfangreiche Werkzeuge aus, die nur mit großem Aufwand an die speziellen Bedürfnisse von MuSofT anzupassen sind. Ein wesentlicher Teil der Arbeit hätte in diesen Fällen darin bestanden, den vorhandenen Funktionsumfang um das zu reduzieren, was in MuSofT in keinem Fall benötigt wird und nur zur Verwirrung des Nutzers führt. Dies schien nicht gerechtfertigt. Y-Files und Jazz schieden aus, weil es sich dabei im wesentlichen um Bibliotheken für Graph-Editoren – im Unterschied zu allgemeinen *graphischen* Editoren – handelt. Komplexe Algorithmen für das Layout und zum Traversieren von Graphen spielen im Teilprojekt keine Rolle. Wichtiger ist die Möglichkeit, weitreichenden Einfluss auf die Gestaltung einzelner Bildschirmfiguren zu nehmen und einen komfortabel zu bedienenden Editor zu erstellen. Gegen Y-Files sprach zudem die Tatsache, dass es sich dabei um ein kommerzielles Produkt handelt, für das jeder Benutzer der MuSofT-Materialien erneut Lizenzgebühren hätte zahlen müssen.

Die Entscheidung in Teilprojekt 2.1 fiel schließlich zugunsten von JHotDraw [Kai03], der Java-Variante eines Frameworks für graphische Editoren, das ursprunglich von Beck und Gamma in Smalltalk implementiert wurde. JHotDraw bietet neben der grundsätzlichen Funktionalität eines Vektor-orientierten Editors bereits einen reichhaltigen Fundus an vorgefertigten Figuren sowie einige Rahmenapplikationen, auf deren Basis mit geringem Aufwand eine eigene Applikation erstellt werden kann. Ebenso liegen diverse Beispielapplikationen vor, etwa ein Editor für Petri-Netze, der als Anschauungsmaterial dienlich war. Der Quellcode von JHotDraw ist frei verfügbar, und das Projekt wird auf Sourceforge gepflegt.

Auf der Basis der Funktionalität von JHotDraw wurden zunächst einige eigene graphische Figuren zur Repräsentation struktureller Elemente einer Architekturbeschreibung entwickelt

(Kapsel-Klassen, Kapsel-Instanzen, Protokolle, Ports, Konnektoren). Um eine saubere Trennung von Modell und Sicht (Editor) zu erreichen, wurden diese Figuren mit einem eigenen (Meta-) Modell unterfüttert, das unabhängig von JHotDraw ist und nur Informationen über die verwaltete Architekturbeschreibung enthält. Das Metamodell bildet, wenn es instanziiert wird, eine Baumstruktur, die sich in eine XML-Datei speichern bzw. aus einer solchen laden lässt. Damit steht prinzipiell die Möglichkeit offen, entstandene Architekturbeschreibungen in anderen Werkzeugen zu verwenden.

Die Trennung von Modell und Sicht erleichterte nicht nur die weitere Entwicklung der Anwendung, sie wurde spätestens in dem Moment unabdingbar, in dem zusätzlich zum eigentlichen Editor-Fenster drei weitere Sichten auf die Architekturbeschreibung hinzukamen: Ähnlich einem echten CASE-Tool enthält die Lernumgebung eine Baumansicht der Architekturbeschreibung, eine überblicksartige Schnellansicht des gesamten Modells sowie eine Detailansicht der Eigenschaften des aktuell ausgewählten oder bearbeiteten Elementes. Diese Ansichten waren ursprünglich nicht geplant, ließen sich aber mit geringem Aufwand realisieren und erleichtern die praktische Arbeit mit dem Werkzeug erheblich, nicht zuletzt, da sie für Anwender, die mit anderen CASE-Tools vertraut sind, eine gewohnte Umgebung bieten.

3.2 Hypertext-System

Das Hypertext-System dient zur Vermittlung des zuvor beschriebenen Lehrstoffes. Die Nutzung multimedialer Möglichkeiten beschränkt sich auf Texte, Grafiken und Hyperlinks. Zur Realisierung eines Teils der Übungen (nach der Art klassischer Aufgabenzettel) können zudem Formular-Elemente verwendet werden.

Die Realisierung des Hypertext-Systems erforderte nur geringen Implementierungsaufwand, da es im wesentlichen auf der Java/Swing-Klasse `JEditorPane` beruht, die bereits in der Lage ist, HTML-Seiten anzuzeigen und – in begrenztem Umfang – sogar Cascading Style Sheets (CSS, vermutlich Level 1) interpretiert. Die Klasse ist zudem in der Lage, auf das Anklicken von Hyperlinks zu reagieren und Zielseiten von Links gegebenenfalls automatisch nachzuladen, so dass der größte Teil des Hypertext-Systems ohne eigenes Zutun funktioniert. Einzige Besonderheit hier sind jene Hyperlinks, die eine spezielle Bedeutung innerhalb des Editors besitzen (z.B. die Architekturbeschreibung ändern). Diese Links werden durch einen eigenen Protokoll-Präfix (anstelle von „http:“) gekennzeichnet und innerhalb der Anwendung gesondert behandelt.

Für die interne Darstellung von HTML-Seiten im Editor sprach die gewünschte enge Verzahnung von Modellen und erläuternden Texten. So ist es zum Beispiel möglich, Elemente eines Modells mit Hyperlinks zu versehen, die beim Anklicken des Elements eine entsprechende Erläuterung im Hypertext-System anzeigen. Ebenso ist es umgekehrt möglich, Links im Hypertext dazu zu verwenden, Modellelemente zu fokussieren. Damit sind verschiedene interessante Lernszenarien denkbar, etwa eine Art von entdeckendem Lernen, bei welcher der Student sich Stück für Stück durch ein komplexes Modell „klickt“, erläuternde Texte zu den einzelnen Teilen aufnimmt und versucht, eine bestimmte Aufgabe zu lösen (etwa eine spezielle Eigenschaft des Modells herauszufinden). Nachteil der Verzahnung von Editor und HTML-Material ist die Tatsache, dass zumindestens Teile des Materials nur schwierig ausserhalb des Systems verwendet werden könnten. In jedem Fall würden dabei die Modelle fehlen (die ja aufgrund ihrer Komplexität ausschließlich innerhalb des graphischen Editors angezeigt

werden), und es würden einzelne „tote“ Links existieren. Das Teilprojekt sucht jedoch nach Wegen, diese Effeke zu minimieren.

3.3 Simulation

Das im Editor modellierte System ist in Grenzen ablauffähig. Zu diesem Zweck stellt die Lernumgebung einen Simulator bereit, der im wesentlichen auf der Basis der Verhaltensbeschreibungen der einzelnen Komponenten in Form von State Charts basiert. Sämtliche State Charts werden parallel interpretiert, während der Benutzer Eingaben für das System liefert. Die resultierenden Zustandsänderungen und Ausgaben können beobachtet werden und erlauben Aussagen über das Systemverhalten, etwa derart, dass ein bestimmter Zustand immer erreicht wird oder niemals erreicht werden kann. Das Ergebnis lässt sich näherungsweise als graphischer Debugger für Software-Architekturen bezeichnen.

Die Implementierung dieser Komponente basiert wesentlich auf der Implementierung einer Harel State Machine [Har87], die in ihrer ursprünglichen Version im Rahmen des UVM-Projektes „Modellieren in einer Virtuellen Welt“ zur Steuerung eben dieser virtuellen Welt verwendet wurde. Während einer späteren Überarbeitung wurde diese State Machine an die UML-Spezifikation angepasst.

Die Simulationskomponente wird in einigen Beispiel-Architekturen verwendet, um Kontroll- und Datenfluss zu verdeutlichen. Sie soll ausserdem massiv in Aufgaben verwendet werden, in welchen die Studenten zum Erstellen eigener Architekturen für bestimmte Probleme aufgefordert sind.

3.4 Animation und Scriptsprache

Zur Verdeutlichung spezieller Eigenschaften vornehmlich der bereits im System enthaltenen Architekturbeschreibungen können Teile einer Architektur animiert werden. Die Animationsmöglichkeiten umfassen simples Hervorheben oder Umrahmen von Elementen oder Verbindungen, aber auch das Ändern oder sukzessive Erstellen einer kompletten Architekturbeschreibung im Editor. Letzteres wird durch eine einfache Scriptsprache unterstützt, die den Editor-Inhalt bzw. das zugrunde liegende Architekturmodell manipulieren kann. Die Kommandos der Scriptsprache können durch spezielle Hyperlinks des Hypertext-Systems ausgelöst werden und bewegen sich etwa auf dem Niveau „Bewege Element A von x nach y“ oder „Verbinde die Elemente B und C“. Die derzeitige Planung sieht eine rein textuelle Verwendung der Scriptsprache vor. Ein spezieller „Makro-Rekorder“ zum Aufzeichnen von Kommandos wäre wünschenswert, wird aber vermutlich aus Zeit- und Aufwandsgründen nicht realisiert werden.

3.5 Die Summe der Teile

Abbildung 1 zeigt die Lernumgebung mit den verschiedenen Sichten auf das Modell. In der Mitte des Bildschirms befindet sich der eigentliche Editor, links oben davon die Baumansicht. Diese verdeckt die Schnellansicht. Im linken unteren Teil ist die Detailansicht der Eigenschaften eines Modell-Elementes zu sehen. Der Bereich am rechten Rand dient zur Anzeige von

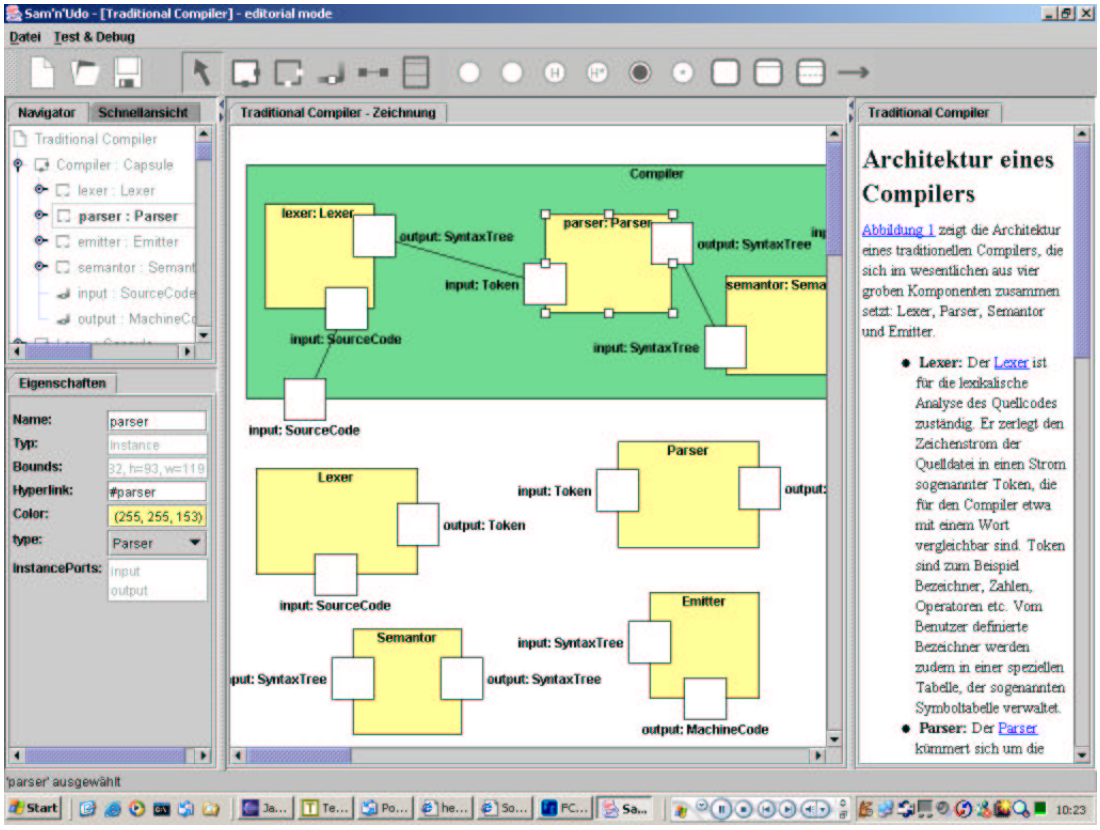


Abbildung 1: Bildschirmphoto der Lernumgebung

Hypertext. Alle Bereiche sind vom Benutzer frei positionierbar. Zusätzlich kann zwischen einigen voreingestellten Konfigurationen gewählt werden, die an bestimmte Anwendungsfälle (Präsentation in einer Vorlesung, Praktische Arbeit in einer Übung) angepasst sind und die Bereiche entsprechend positionieren bzw. nicht benötigte Bereiche komplett ausblenden.

4 Evaluierung

Die Lerneinheit ist noch nicht im Einsatz, soll aber in Teilen im Sommersemester 2003 erstmalig an der Universität Dortmund im Rahmen der Stammvorlesung „Softwaretechnik“ des Hauptstudiums Informatik eingesetzt werden. Neben dem Einsatz als Werkzeug zur Visualisierung von Architekturen besteht auch Interesse daran, den Editor bzw. die Simulation der State Charts separat zu verwenden, da dieser visuelle Formalismus sonst üblicherweise mit Tafel und Kreide bzw. Folien eingeführt wird, was gerne zu Problemen führt.

Für den Einsatz bzw. die Evaluation der Lerneinheit sind ansonsten folgende Veranstaltungen an der Universität Dortmund geeignet:

- Der Übungsbetrieb zur bereits erwähnten Vorlesung „Softwaretechnik“. Hier nutzen die Studierenden den Editor zur Lösung von Aufgaben und verwenden das hypermediale Lehrmaterial zum Nacharbeiten des Vorlesungsstoffes.
- Das Software-Praktikum. In dieser Veranstaltung werden bereits jetzt CASE-Tools projektbegleitend eingesetzt, jedoch bewegen sich diese auf der Ebene von UML-Analyse- und Entwurfsdiagrammen. Der Editor könnte hier helfen, frühzeitig eine Architektur für die zu entwickelnde Anwendung auszuwählen, die anschließend mit den bisherigen Werkzeugen verfeinert werden kann.
- Eventuelle Spezialvorlesungen des Hauptstudiums Informatik, die sich mit Software-Architektur beschäftigen.

Für den Einsatz in Vorlesungen an der Universität Dortmund ist die Zielgruppe – nämlich die der Lehrenden – so klein, dass sich eine Evaluation über formale Fragebögen nicht anbietet. Der Stichprobenumfang ist zu gering, um eine quantitative Aussage zu erlauben. Hier wird also der subjektive Eindruck des Lehrenden genügen müssen, dass die Präsentation von Architekturen mit dem Editor „besser“ oder „schlechter“ als das bisherige bisherigen System (Tafelanschrieb oder Folien) war. Zur Sammlung dieser subjektiven Eindrücke bieten sich (unstrukturierte oder strukturierte) Interviews an. Eventuell ändern sich die Evaluationsmöglichkeiten wenn das Werkzeug eine größere Verbreitung (innerhalb und ausserhalb von MuSofT) findet.

Im Übungsbetrieb und im Praktikum ist es sehr wohl möglich, die Studenten zu ihren Erfahrungen mit dem Werkzeug zu befragen. Speziell am Software-Praktikum in Dortmund nehmen so viele Studenten teil, dass es sich möglicherweise anbietet, ein kontrolliertes Experiment durchzuführen, um den Einfluss der Verwendung des Werkzeugs auf den Projektfortschritt (Einhaltung des Zeitplans, gefundene Fehler in späteren Phasen) zu ermitteln. Genauere Pläne hierzu liegen aber noch nicht vor.

5 Planung für das nächste Jahr

Die Implementierung der Lerneinheit ist – nicht zuletzt dank der Tatsache, dass im Herbst 2002 endlich mehrere dringend benötigte studentische Hilfskräfte eingestellt werden konnten – inzwischen recht weit fortgeschritten. Die Rahmenapplikation sowie der Editor für die strukturellen Anteile einer Architekturbeschreibung funktionieren. Einfache Architekturen können modelliert, gespeichert, geladen und ausgedruckt werden. An den Teilen des Editors, die sich mit der Verhaltensmodellierung durch State Charts beschäftigen, wird derzeit gearbeitet. Da dies aber im wesentlichen ein Übertragen der zuvor gemachten Erfahrungen bedeutet (neue Elemente für das Metamodell, neue Figuren für den Editor), sind hier keine technischen oder konzeptionellen Schwierigkeiten zu erwarten. Zum Ende des Winters hin soll das System in einem Stadium sein, das ein praktisches Arbeiten erlaubt. Ab diesem Zeitpunkt werden auch die benötigten Inhalte produziert. Gleichzeitig werden sich die studentischen Hilfskräfte mit der Implementierung der restlichen Komponenten des Systems beschäftigen:

- Die Simulationskomponente erfordert eine geeignete Integration der bereits vorhandenen State Machine in den Editor sowie entsprechende Elemente zur Anzeige des Zustandes einzelner Systemteile (im wesentlichen der State Machine jeder Kapsel) und zur Kontrolle der Simulation (wahrscheinlich durch ein Schalterfeld, das dem eines CD-Spielers ähnelt).

- Die Animationskomponente wird sich das „Decorator"-Entwurfsmuster zunutze machen, um im Editor vorhandene Figuren mit weiteren Figuren zu „schmücken", die nur zu Animations- oder Hervorhebungszwecken dienen.

- Die Scriptsprache, die bereits prototypisch umgesetzt wurde, wird ausgebaut. Der derzeitige Ansatz nutzt die „Reflection"-Schnittstelle von Java, um Elemente des Architekturmodells über Methodenaufrufe zu manipulieren.

Diese Arbeiten sind hinreichend unabhängig voneinander, um parallel durch jeweils eine studentische Hilfskraft durchgeführt werden zu können.

Um den multimedialen Charakter der Lernumgebung stärker zu betonen, beteiligt sich das Teilprojekt ausserdem an der von mehreren Teilprojekten betriebenen Video-gestützten Umsetzung einer Fallstudie aus dem Bereich Lagerhaltung/Logistik. Planungen hierzu sowie eine Führung im Hochregallager von Bertelsmann, Gütersloh, haben bereits stattgefunden. Im Rahmen dieser Führung entstand auch erstes Video-Rohmetarial, das als Vorlage für ein professionell umzusetzendes Drehbuch dienen soll. An der Abstimmung dieses Materials mit einer der Fallstudien der Lerneinheit wird gearbeitet.

Ein weiteres geplantes multimediales Element ist die Vertonung von erläuterndem Text zu einer oder mehreren Fallstudien. Die Idee hierzu kam im Herbst 2002 auf: Wie sich bei ersten Arbeiten mit dem System zeigte, verlangt die gleichzeitige Darstellung von Architekturmodellen und erläuterndem Text dem Studierenden einen häufigen Wechsel zwischen beiden Elementen ab, da beide seinen visuellen Fokus erfordern. Darunter leidet potentiell die Aufmerksamkeit des Studierenden. Eine Umsetzung des erläuternden Textes mit auditiven Mitteln würde es dem Studierenden erlauben, sich visuell auf das Modell zu konzentrieren und die Erläuterungen parallel dazu und ohne Ablenkung aufzunehmen, was speziell bei animierten oder inkrementell entstehenden Architekturmodellen hilfreich sein dürfte. Es ist derzeit allerdings noch nicht klar, ob diese Idee zeitlich und finanziell im letzten Projektjahr umsetzbar ist.

Zur Unterstützung der multimedialen Elemente sowie der allgemeinen Gestaltung der Lerneinheit finden zu Beginn des Jahres 2003 im Rahmen von MuSofT zwei Workshops statt, die von einer externen Multimedia-Expertin organisiert werden. Von beiden Workshops erhofft sich das Teilprojekt wertvolle Impulse.

6 Zusammenfassung

Software-Architektur ist ein eher praktisches Thema innerhalb der Softwaretechnik. Das muss bei der Lehre berücksichtigt werden. Neben der Vermittlung von reinem Theoriewissen in Vorlesungen, das selbstverständlich unverzichtbar ist, muss den Studierenden eine Möglichkeit gegeben werden, eigene Erfahrungen auf dem Gebiet der Software-Architektur zu sammeln

oder von den Erfahrungen derer zu profitieren, die sich zuvor mit dem Thema beschäftigt haben.

Die Lerneinheit, die innerhalb von Teilprojekt 2.1 entwickelt wird, versucht eine Infrastruktur zu schaffen, die diesem Wunsch Rechnung trägt: Das Sammeln eigener Erfahrungen wird durch die Editor-Komponente ermöglicht, die eine Vertiefung des Vorlesungsstoffes in Übungen ermöglicht. Das Hypertext-System erlaubt dabei jederzeit einen Rückgriff auf Teile des Vorlesungsstoffes oder weiterführende Informationen. Das Erfahrungswissen zahlreicher Software-Ingenieure wird – ähnlich Entwurfsmustern – durch Architekturstile kodifiziert, von denen die Lerneinheit eine kleine Auswahl vermittelt. Die Fallstudien erlauben zudem einen Blick darauf, wie sich Architekturstile auf die Eigenschaften verschiedener realer und imaginärer Projekte auswirken.

Insgesamt – so hofft der Autor – wird sich aus den in diesem Bericht genannten Zutaten eine Lerneinheit ergeben, die den Studierenden einen multimedialen Ein- und Überblick der Software-Architektur gibt und sowohl Vorlesungen als auch Übungs- und Praktikumsbetrieb der Softwaretechnik bereichert.

Literatur

[BMR+96] BUSCHMANN, FRANK, REGINE MEUNIER, HANS ROHNERT, PETER SOMMERLAD und MICHAEL STAL: *Pattern-Oriented Sofware Architecture*. John Wiley and Sons, 1996.

[GHJV96] GAMMA, ERICH, RICHARD HELM, RALPH JOHNSON und JOHN VLISSIDES: *Entwurfsmuster – Elemente wiederverwendbarer objektorientierter Software*. Addison-Wesley, 1996.

[GS94] GARLAN, DAVID und MARY SHAW: *An Introduction to Software Architecture*. Internal Report CMU-CS-94-166, 1994.

[Har87] HAREL, DAVID: *Statecharts: A Visual Formalism for Complex Systems*. Science of Computer Programming, 8(3):231–274, June 1987.

[HHK+96] HOFMANN, CHRISTOPH, ECKART HORN, WOLFGANG KELLER, KLAUS RENZEL und MONIKA SCHMIDT: *The Field of Software Architecture*. Technischer Bericht TUM-I9641, 1996.

[Kai03] KAISER, WOLFRAM: *Become a programming Picasso with JHotDraw*. http://www.javaworld.com/javaworld/jw-02-2001/jw-0216-jhotdraw.html, 2001 (zuletzt gesichtet am 01.03.2003).

[Lyo03] LYONS, ANDREW: *UML for Real-Time Overview*. 1998 (zuletzt gesichtet am 06.01.2003). http://www.rational.com/products/whitepapers/100463.jsp.

[Ple01] PLEUMANN, JÖRG: *Lerneinheit 2.1 – Software-Architektur*. Ergebnisbericht des Jahres 2001 des Projektes MuSofT – Multimedia in der SoftwareTechnik, Seiten 39–53, 2001.

[PW92] PERRY, DAVID E. und ALEXANDER L. WOLF: *Foundations for the Study of Software Architecture*. ACM SIGSOFT Software Engineering Notes, 17(4):40–52, 1992.

[SG95] SHAW, MARY und DAVID GARLAN: *Software Architecture - Perspectives on an Emerging Discipline*. Prentice Hall, 1995.

[SGW94] SELIC, BRAN, GARTH GULLEKSON und PAUL T. WARD: *Real-Time Object-Oriented Modeling*. John Wiley and Sons, 1994.

[SR03] SELIC, BRAN und JIM RUMBAUGH: *Using UML for Modeling Complex Real-Time Systems*. 1998 (zuletzt gesichtet am 06.01.2003). http://www.rational.com/products/whitepapers/442.jsp.

[SW99] STAFFORD, JUDITH A. und ALEXANDER L. WOLF: *Architecture-Based Software Engineering*. Internal Report CU-CS-891-99, 1999.

Lerneinheit Entwurfsmuster - Ergebnisbericht 2002 MuSofT Teilprojekt 2.2

Silke Seehusen, Nina Nissen

Es wird das Konzept und die Struktur der der Lerneinheit Entwurfsmuster dargestellt, die in Teilprojekt 2.2 der Projektes MuSofT entwickelt wird. Das Ziel der Lerneinheit besteht in der Vermittlung des allgemeinen Konzeptes von Entwurfsmustern und in der Vermittlung von ausgewählten Entwurfsmustern. Die Struktur der Lerneinheit wird technisch mit XML nach der DTD L3-XML festgelegt. Die Medienobjekte der Lerneinheit werden mit verschiedenen Werkzeugen erstellt und mit L3-XML oder mit XHTML in die Lerneinheit integriert. Aus der Beschreibung in XML werden die Lehr- und die Lernansicht mit einem Werkzeug generiert. Es werden die Erfahrungen mit den Einsatz der ersten Kapitel und das Konzept für die geplante Evaluation vorgestellt.

Inhaltsverzeichnis

1 Einleitung

Zur Unterstützung der Lehre auf dem Gebiet Entwurfsmuster wird eine Lerneinheit zur Einführung in Entwurfsmuster entwickelt. Das Ziel der Lerneinheit Entwurfsmuster besteht in der Vermittlung des allgemeinen Konzeptes von Entwurfsmustern und in der Vermittlung von ausgewählten Entwurfsmustern. Ein wichtiges Lernziel besteht darin, dass die Studierenden selbständig weitere Entwurfsmuster bei Bedarf auswählen, sich aneignen und umsetzen können.

Im Abschnitt 2 wird die Struktur der Lerneinheit skizziert. Im Abschnitt 3 wird das Gesamtkonzept der technischen Realisierung beschrieben. Aus den bisherigen Erfahrungen werden Vorschläge für die weitere Arbeit im Jahr 2003 entwickelt. Im Abschnitt 4 werden die Arbeiten zur Evaluation vorgestellt. Im Abschnitt 4.1 wird der Einsatz und die Evaluation der Lehr- und Lernmittel SmartClass und SmartBoard präsentiert. In Abschnitt 5 wird das Jahr 2003 geplant und der Bericht endet mit einer Zusammenfassung.

2 Vorstellung der Lerneinheit

Die Lerneinheit Entwurfsmuster soll eine allgemeine Einführung in Entwurfsmuster und in die Beschreibungsstruktur von Entwurfsmustern geben. Desweiteren werden wichtige Kategorien von Entwurfsmustern wie Architektur-, Erzeugungs-, Struktur- und Verhaltensmuster sowie Muster zur Arbeitsorganisation eingeführt. Zu diesen ausgewählten Kategorien werden repräsentative und häufig verwendete Entwurfsmuster präsentiert.

Die Lerneinheit ist in 5 Teile gegliedert, die jeweils aus Lernmodulen (LM, nach der MuSofT-Terminologie) bestehen. Die Teile 3 und 5 haben Referenzcharakter. Auf diese Teile wird aus den anderen Lernmodulen verwiesen.

Teil 1: Einführung in Entwurfsmuster

Anhand eines ausgewählten Entwurfsmusters, Pipes-and-Filter, wird das Konzept, die Darstellungsmethode, das Beschreibungsschema und die UML-Notation von Entwurfsmustern eingeführt.

Teil 2: Entwurfsmuster

Es wird eine Auswahl weiterer Entwurfsmuster eingeführt, wobei jeweils eins oder mehrere aus verschiedenen Kategorien wie Architektur-, Erzeugungs-, Struktur- und Verhaltensmuster sowie Muster zur Arbeitsorganisation gewählt werden.

Jedes Entwurfsmuster wird in einem eigenen Lernmodul dargestellt. Aus den Erzeugungsmustern werden die Entwurfsmuster Singleton und Fabrikmethode eingeführt. Aus den Verhaltensmustern werden die Entwurfsmuster Strategie, Beobachter (Observer), Befehl (Command), MVC, Iterator und Besucher (Visitor) vorgestellt. Aus den Strukturmustern werden die Entwurfsmuster Kompositium, Adapter und Brücke (Bridge) aufgenommen. Als Arbeitsorganisationsmuster werden Delegation, Master-Slave und Blackboard präsentiert.

Zu der Beschreibung jedes Entwurfsmusters gehört ein Einführungsbeispiel, eine Beschreibung nach dem Beschreibungsschema, Klassendiagramme, Sequenzdiagramme, evtl. Kollaborationsdiagramme, evtl. Zustandsdiagramme, Animationen (orientiert an Objekt- und Sequenzdiagrammen), ein Anwendungsbeispiel (mit konkreten Klassendiagrammen), das komplett (in Java) verfügbar ist und das möglichst ein Teil des durchgängigen Beispiels aus Teil 3 darstellt, und, wenn vorhanden, die Verwendung des Entwurfsmusters in einer Java-API, und Aufgaben.

Teil 3: Durchgängiges Beispiel

So weit wie möglich wird ein durchgängiges Beispiel zur Demonstration der Anwendung der Entwurfsmuster einschließlich Implementierung in Java eingesetzt. Als Beispiel wird die Realisierung eines Kommunikationswerkzeuges mit den Funktionen Bulletin Board, Chat und WhiteBoard entwickelt. In diesem Teil 3 wird das Beispiel zusammenhängend dargestellt. In Teil 2 wurde pro Entwurfsmuster jeweils ein Teilaspekt näher beleuchtet. Der Teil 3 dient als Überblick und als Referenz zum durchgängigen Beispiel.

Teil 4: Ausblick

Es folgt ein Überblick über weiterführende Verweise und Literatur. Ein Lernziel besteht darin, dass die Studierenden lernen, selbständig weitere Entwurfsmuster bei Bedarf auswählen, sich aneignen und umsetzen können.

Teil 5: Referenzteil

In Teil 2 werden die einzelnen Entwurfsmuster einführend beschrieben. Im Teil 5, dem Referenzteil, wird jedes Entwurfsmuster noch einmal nach einem festen Beschreibungsschema kurz beschrieben. Es ist kein Lernmodul im engeren Sinne, da es einen Referenzcharakter besitzt. Es ist jedoch ein Bestandteil der Lerneinheit und kann insbesondere auch zur Wiederholung oder zur Auswahl von Entwurfsmustern genutzt werden.

Aufgaben

Aufgaben sollen die Studierenden zum aktiven Lernen motivieren. Einige Aufgaben dienen der Selbsteinschätzung der Studierenden (Lernfortschrittskontrolle), andere Aufgaben dienen als Anreiz, sich tiefer mit dem Stoff zu beschäftigen.

Die Aufgaben können z.B. wiederholend sein, es können Analyseaufgaben sein, sie können eine Reflexion des Stoffes und/oder die Konstruktion einer eigenen Lösung eines vorgegebenen Problems (die Aufgabe im engeren Sinne) erfordern. In allen Lernmodulen aus Teil 1 und 2 werden Aufgaben verschiedener Art gestellt.

3 Technische Realisierung

Bei der Entwicklung der Lerneinheit werden eine Reihe von Techniken und Werkzeuge eingesetzt. Die Entwicklung von multimedialen Einheiten erfordert zur Zeit immer noch, dass für unterschiedliche Medien unterschiedliche Werkzeuge verwendet werden müssen. Darüberhinaus müssen für einige Medien je nach Einsatzeigenschaften verschiedene Werkzeuge eingesetzt werden. Sobald firmeneigene Formate verwendet werden, ist die Nachhaltigkeit sehr gefährdet.

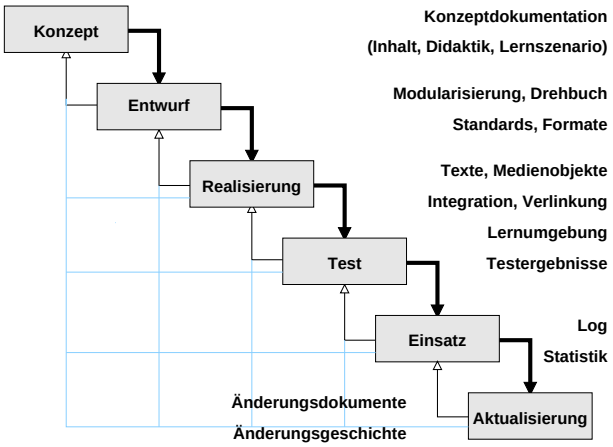


Abbildung 1: Vorgehensweise und Entwicklungsdokumente

3.1 Entwicklungsprozess

Für die Realisierung der Lerneinheit wurde ein Entwicklungsprozess weiterentwickelt, der in Abbildung 1 dargestellt wird. Die Entwicklung der Lerneinheit ist eine iterative Entwicklung, die dennoch in Phasen strukturiert werden kann [SL01]. In Abbildung 1 werden die Phasen Konzept, Entwurf, Realisierung (Implementierung), Test, Einsatz und Aktualisierung zueinander in Beziehung gesetzt.

In den einzelnen Phasen werden normalerweise Entwicklungsdokumente erstellt, wie in Abbildung 1 skizziert.

Zur Beschreibung des Konzeptes gehören das inhaltliche Konzept (Stoffplan, Methoden, Kompetenzen), die didaktischen Vorüberlegungen (Leitbild, Lernziele (des gesamten Kurses), Lernszenarien und didaktischer Ansatz) sowie die Festlegung allgemeiner Eigenschaften (z.B. Nachhaltigkeit, Plattformneutralität). Der Entwurf kann aus einem Drehbuch bestehen oder nach einer schrittweisen Verfeinerung der Lernmodule beschrieben werden. Die Realisierung umfasst die Texte und die weiteren Medienobjekte sowie deren Integration.

Es werden in der Regel verschiedene Tests durchgeführt, die entsprechend geplant, durchgeführt und dokumentiert werden.

Während des Einsatzes wird die Lerneinheit wiederholt in verschiedenen Lehrveranstaltungen genutzt. Eine begleitende Evaluation wird durch entsprechende Logs und durch statistische Auswertung unterstützt.

Nach oder sogar während jeden Durchlaufes wird der Kurs in der Regel aktualisiert. Zur Aktualisierung gehört neben der Korrektur von Fehlern, Verbessern der Darstellung auch die Aufnahme von neuen Forschungs- und Entwicklungsergebnissen aus dem vermittelten Fachgebiet. Die Aktualisierung kann im Extremfall eine fast komplette Neuentwicklung beinhalten.

3.2 Struktur und Integration mit XML

Um die Entwicklung durchgängig zu unterstützen, werden alle Entwicklungsdokumente in einem Dokument vereinigt. Ein Entwicklungsdokument stellt dann eine spezielle Sicht des Gesamtdokumentes dar.

Zur Spezifikation der Struktur der Lerneinheit wird XML [W3C00] verwendet, um insbesondere auch die Interoperabilität von verschiedenen Werkzeugen zu unterstützen. Es wurde L3-XML [SLK00], eine XML-DTD, weiterentwickelt, die die einzelnen Phasen der Entwicklung und des Einsatzes unterstützt.

Die einzelnen Medienobjekte wie Text, Bild, Animation, Ton und Interaktion werden durch Verweise bzw. Attribute in die Struktur eingeordnet. Aus einer L3-XML-Beschreibung wird mit einem Werkzeug eine Menge von vernetzten HTML-Seiten generiert.

3.3 Lehr- und Lernansicht

Aufgrund der bisherigen Erfahrungen mit den ersten Kapiteln der Lerneinheit im WS 2001/02 wurden mehrere Sichten definiert. Insbesondere wurden eine Lehr- und eine Lernansicht von den Funktionsmöglichkeiten und dann vom grafischen Entwurf festgelegt.

Die Lehransicht ist die Ansicht zur Präsentation der Lehr-/Lerninhalte in einer Vorlesung, Übung oder Praktikum. Diese Sicht ist für den Einsatz von Lehrenden konzipiert.

Die Lernansicht ist die Ansicht, die den Studierenden zum Nachschlagen, Vertiefen oder generell zum selbstorganisierten Studium bereit gestellt wird.

Die Anforderungen an diese Sichten unterscheiden sich sehr stark, obwohl die dargestellten inhaltlichen Konzepte gleich sind. Neben Schriftgrößen, Formatierung, Marginalien liegt der wesentliche Unterschied in dem Ausblenden von Details in der Lehransicht und den unterschiedlichen Navigationsmöglichkeiten und dem Zugang und der Darstellung zu der Navigation.

3.4 Navigationen und Lernpfade

In der Lerneinheit werden verschieden Arten der Navigation ermöglicht, die durch die Erstellung der Struktur und der Integration in XML und die Angabe von Metadaten ermöglicht werden. Aus diesen Informationen werden in den jeweiligen Sichten Verweisstrukturen und Verweislisten generiert.

Zu den Navigationsmöglichkeiten gehören:

- Inhaltsverzeichnis und Medienverzeichnisse: Die Verzeichnisse, die uns aus gedruckten Dokumenten bekannt sind, werden auch in einer Lerneinheit zur Verfügung gestellt und mit interaktiven Verweismöglichkeiten ergänzt. Zu den Verzeichnissen gehören das Inhaltsverzeichnis, Aufgabenverzeichnis und Medienverzeichnisse von z.B. Applets und Animationen.

- Lernpfade: Die Definition von verschiedenen Lernpfaden (Lernpfade siehe [SLK00]) wird in L3-XML explizit angegeben. Der Standardlernpfad (empfohlene Lernsequenz) wird hervorgehoben. In der Lehrversion entspricht diesem Lernpfad die normale Reihenfolge der Präsentation.

- Positionsübersicht: Zur Unterstützung der Navigation und der Übersicht der aktuellen Position in der Lerneinheit wird eine Übersichtskarte über die ganze Lerneinheit mit entsprechenden Verweisen in einer sensitiven Map dargestellt.
- Glossar: Innerhalb der Lerneinheit werden Verweise auf das Glossar bei den einzelnen Begriffen hinterlegt.
- Suchfunktion: Über eine Suchfunktion [LS02], die in Java als Applet implementiert ist, wird eine Volltextsuche auf der Lerneinheit und eine Suche nach den Metadaten zur Verfügung gestellt.

3.5 Medienerstellung

Für die einzelnen Medien werden verschiedene Werkzeuge eingesetzt (siehe [SN02]). Für Texte, die in XHTML geschrieben werden, wird im wesentlichen *emacs* eingesetzt.

Für Bilder werden werden nach wie vor *xfig, Gimp* und *Illustrator* eingesetzt, je nach Art des Bildes. Wichtig ist dabei, dass das erstellte Bild in der Ansicht ohne großen Qualitätsverlust skalierbar ist. Als Formate eignen sich dazu Vektorgrafikformate wie svg, PostScript und, eingeschränkt, JPEG.

Da in einer Lerneinheit genormte grafische Beschreibungsmittel wie z.B. Klassendiagramme oft in Bildern vorkommen, muss die Darstellung von z.B. Klassen in allen Bildern sehr ähnlich sein. Deshalb muss dazu eine Art Bildbibliothek angelegt werden.

Die Animationen wurden mit Flash erstellt. Eine Animation muss von dem Betrachtenden gesteuert werden können. Dazu wurde eine entsprechende Bedienungsleiste realisiert, die zu jeder Animation zur Verfügung gestellt wird.

Der Entwicklungsaufwand für Animationen mit Flash ist relativ hoch. Deshalb muss auch hier eine Art Bibliothek angelegt werden.

3.6 Anmerkungen in Programmtexten

Ein wichtiges Merkmal der Lerneinheit besteht auch darin, dass zu einem Entwurfsmuster auch ein lauffähiges Beispiel in Java präsentiert wird. Die Präsentation muss neben z.B. einem Klassendiagramm auch fast immer die Darstellung von Programmtext vorsehen, da nur so eine konkrete Umsetzung gezeigt werden kann. Da Programmtexte von Beispielen für die Anwendung von Entwurfsmustern relativ lang sind, wird oft auf erklärende Kommentare im dargestellten Text verzichtet.

Um jedem Studierenden die Möglichkeit zu geben, Details der Erklärung des Programmtextes leicht nachzuschauen, wurde das Werkzeug *acceNun* im Teilvorhaben 2.2 entwickelt, das aus dem Programmtext eine Ansichtsversion generiert, in denen markierte Texte maussensitiv mit erklärenden Text hinterlegt werden können. Das Werkzeug generiert aus einem Java-Programmtext ein XHTML-Seite mit JavaScript-Funktionsaufrufen. Auf diese Seite kann zur farbigen Markierung der Java-Schlüsselworte das Werkzeug *Java2Html* angewendet werden [Geb02], das als Open source unter der GNU General Public License (GPL) verfügbar ist, so dass eine visuell ansprechende Darstellung des Quelltextes entsteht. Diese Darstellung kann in HTML-Seiten eingebunden werden. Das Werkzeug *acceNun* kann eigenständig

oder aus einem anderen Programm wie bei der Generierung der venetzten HTML-Seiten aufgerufen werden.

Da der Text, der Programmteile erklärt, direkt im Programmtext als Kommentar eingefügt wird, wird die konsistente Aktualisierung von Programm und Dokumentation unterstützt, so wie prinzipiell auch die JavaDoc-Kommentare diese Konsistenz unterstützen.

3.7 Interaktive Aufgaben

Auf dem Gebiet der Entwurfsmuster gibt es diverse Aufgaben, die darin bestehen, einen vorgegebenen Software-Entwurf, der als Klassendiagramm dargestellt ist, gemäß der Aufgabenstellung zu ergänzen und/oder zu verändern. Um diese Art der Aufgaben für die Studierenden direkt am Rechner bearbeitbar und dann vom System überprüfbar zu machen, wurde im Teilvorhaben 2.2 ein Klient/Serversystem *X2ex* zur Unterstützung dieser Aufgaben entworfen und bereits teilweise implementiert. Die Realisierung wird 2003 abgeschlossen.

Das System stellt für den Autor und die Autorin einen Editor für solche einfachen Entwurfs-Aufgaben zur Verfügung, mit dem die Aufgabenstellung, das vorgegebene Klassendiagramm und die möglichen Lösungen erstellt werden können. Die Klassendiagramme werden in einem vereinfachten UML-Editor erstellt, wobei der Umgang und das Verhalten des Editors wie z.B. in Poseidon gestaltet ist. Die Aufgaben und die Lösungen werden in XML abgelegt.

Bei der Präsentation einer Aufgabe gibt die lernende Person die Lösung an, indem sie wie mit einem einfachen UML-Editor das vorgegebene Klassendiagramm verändert. Die Lösung wird dann über RMI am Server, von dem die Aufgabe geladen wurde, überprüft.

4 Evaluierung

Einige Lernmodule wurden in einem ersten Test an der Fachhochschule Lübeck im 4. Semester in der Lehrveranstaltung Programmiertechniken der Studienrichtung Informatik des Studiengangs Kommunikations-, Informations- und Medientechnik (KIM) im WS 2001/02 und SS 2002 eingesetzt. Die Lernmodule wurden in einer Art Ansicht eingesetzt, die noch nicht zwischen Lehr- und Lernansicht unterschieden hat. Der Einsatz diente dem Sammeln von Erfahrungen, die in der weiteren Entwicklung verwendet werden können.

Aus den ersten Erfahrungen kann folgendes festgehalten werden. Die vielen Möglichkeiten der Navigation erweisen sich gerade bei der interaktiven Präsentation als sehr hilfreich, weil auf direkte Fragen der Studierenden auch entsprechendes Material schnell präsentiert werden kann, ohne lange Suchvorgänge in Menüs oder Folienstapeln.

Die gemeinsame Lehr- und Lernansicht hat für die Präsentationsansicht in der Vorlesung die Nachteile, dass ein zu großer Teil des projizierten Bildes von Navigationselementen eingenommen wird. Deshalb wurde, wie in Abschnitt 3.3 dargestellt, eine Lehr- und eine Lernansicht entwickelt.

Zwei Entwurfsmuster mit Animationen wurden im Wintersemester 2002/03 eingesetzt. Die Animationen wurden von den Studierenden sehr positiv aufgenommen.

Auswertung der Ergebnisse aus anderen Projekten

Eine abschließende Evaluation ist nach Plan für das zweite Halbjahr 2003 vorgesehen. Dazu wurde zunächst eine Analyse der veröffentlichten Ergebnisse der Evaluationen [AKT02, HTH02, HB02] des Projektes Virtuelle Fachhochschule (VFH) mit dem Ziel durchgeführt, welche Ergebnisse davon für die Fragestellungen von MuSofT wichtig sind. Es wurden Aspekte aus den Bereichen Lernmaterialien, Lernraum, Betreuung und Kommunikation untersucht.

Die Evaluationen beziehen sich auf Kurse des Online-Studiums. Viele Bewertungen der Studierenden und Lehrenden sind daher im Zusammenhang mit dem Gesamt-Konzept der VFH zu sehen (wie z.B. Probleme mit der Lernraumumgebung, mit der alle Studierenden und Lehrenden arbeiteten, Studienorganisation, Rahmenbedingungen). Die Ergebnisse sind daher nur bedingt auf den Einsatz der Lerneinheit Entwurfsmuster in einer reinen Präsenzlehre übertragbar.

Themenbereiche

Aufgrund der Analyse der Ergebnisse und der globalen Fragestellung zum Einsatz multimedialer Elemente wurden für die Lerneinheit Entwurfsmuster in Teilprojekt 2.2 konzeptionell Fragestellungen zu den folgenden Themenbereichen festgelegt:

- Allgemeine Beurteilung der gesamten Lerneinheit,
- Art der Studierenden, Leistungsstand, Voraussetzungen,
- Wirkung der multimedialen Elemente, insbesondere von Animationen und interaktiven Entwurfsaufgaben, auf die Studierenden,
- Bewertung durch die Lehrenden,
- Leistungskontrolle der Studierenden,
- Wissensrepräsentation bei den Studierenden und
- Nutzen der Interaktivität allgemein.

Methodisch bietet sich ein pragmatisches Vorgehen der Evaluation mit qualitativen Methoden an, da die Evaluation im SS 2003 und im WS 2003/04 nur mit einer relativ kleinen Gruppe von Studierenden durchgeführt werden kann.

4.1 Lehr- und Lernmittel

Aus Mitteln des Projektes wurden für den Ersteinsatz als elektronische Lehr- und Lernmittel SmartClass und ein SmartBoard beschafft und erste Evaluationen durchgeführt.

SmartClass

Das SmartClass-System besteht aus einer zuverlässigen Schalttechnologie und verbindet jeden Rechner in einem Ausbildungsrechnerraum mit einer zentralen Kontrolleinheit, die vom Lehrenden bedient wird. Die Kontrolleinheit hat eine Matrixanordnung von Stationstasten, die es dem Lehrenden auf einfache Art erlaubt, den Systemstatus zu überwachen, Studierendenstationen auszuwählen und die Bildschirmausgabe und auch die Tastatur- und Mauseingabe zu übernehmen. Insbesondere ist auch eine Projektion eines Bildschirms auf alle anderen Bildschirme möglich.

Im normalen Praktikumsbetrieb hat sich das System sehr gut bewährt, weil es für die Lehrenden intuitiv und leicht erlernbar ist. Eine kurze Demonstration der Bedienung von wenigen Minuten reichte. Da das System den Praktikumsbetrieb sehr gut unterstützt, wurde es von den verschiedenen Lehrenden, die den Praktikumsraum für Lehrveranstaltungen nutzen, eingesetzt.

Aus Projektmitteln wurden nur 6 Plätze mit SmartClass ausgestattet, um die Einsatzmöglichkeiten erst einmal auszutesten. Aufgrund der positiven Erfahrungen wurde auch die übrigen Rechner im Praktikumsraum aus Hochschulmitteln 2002 um SmartClass ergänzt.

SmartBoard

Als weiteres elektronisches Lehr- und Lernmittel wurde als interaktives Whiteboard ein einfaches SmartBoard aus Projektmitteln beschafft. Es erlaubt die Präsentation und die Interaktion von Anwendungen direkt am Whiteboard, ohne dass der Dozent oder die Dozentin sich hinter einem Rechner verstecken muss.

Da wir eine mobile Version des SmartBoard gewählt haben, musste vor jedem Einsatz die Entfernung zum Beamer kalibriert werden. Eine solche Rüstzeit von ca. 5 Minuten ist für den Regelbetrieb nicht geeignet.

War das SmartBoard eingerichtet, so haben sich in einem Versuch die Studierenden sehr schnell die Funktionalitäten für eine Präsentation genutzt.

Zur besseren Ausnutzung wurde aus Hochschulmitteln ein Beamer mit deutliche höherer Auflösung beschafft und es ist geplant, die Rüstzeit durch eine andere Organisation deutlich zu verkürzen.

5 Planung für das nächste Jahr

Im Jahr 2003 werden die Projektarbeiten zum Abschluss gebracht. Wie in den vorigen Kapiteln schon jeweils erwähnt, sind folgenden Schwerpunkte geplant:

- Fertigstellung der Lerneinheit: Alle Kapitel der Lerneinheit, wie in 2 vorgestellt, werden zu Ende entwickelt. Neben Bildern sind weitere Animationen geplant. Desweiteren werden die Anmerkungen in Programmtexten ergänzt, die von dem Werkzeug *acceNun* aufbereitet werden.

  Die einzelnen Teile der Lerneinheit werden kategorisiert, um entsprechende Metadaten zu generieren, die Suche zu unterstützen und die Marginalien konsequent zu nutzen, um die Übersichtlichkeit zu erhöhen.

- Durchgängiges Beispiel: Als für die Lerneinheit durchgängiges Beispiel wird das einfache Kommunikationswerkzeug als Lehrbeispiel zu Ende entwickelt und beschrieben.

- Fertigstellung und Verfeinerung von Werkzeugen: Das Werkzeug *X2ex* wird im ersten Quartal fertiggestellt. Das Werkzeug zur Generierung der vernetzten HTML-Seiten und *acceNun* werden an die neu entwickelten Sichten für Lehrende und für Lernende angepasst. Das Konzept zur Generierung einer guten Druckversion wird fertiggestellt. Die Realisierung kann mit studentischen Hilfskräften begonnen werden. Mit einer Fertigstellung kann 2003 nicht gerechnet werden, so dass ein definierter Zwischenstand definiert wird, so dass die Arbeiten ab 2004 im Rahmen von studentischen Arbeiten fortgeführt werden können.

- Evaluation: Aus dem Konzept der Evaluation werden entsprechend den ausgewählten Instrumentarien die Fragebögen, Interview-Leitfaden etc. entwickelt. Die Daten werden im Sommersemester 2003 an der FH Lübeck erhoben, anschließend ausgewertet und die Ergebnisse aufbereitet.

- Abschlussbericht: Der Abschlussbericht für das Teilprojekt 2.2 wird erstellt.

6 Zusammenfassung

Das Ziel der Lerneinheit Entwurfsmuster besteht in der Vermittlung des allgemeinen Konzeptes von Entwurfsmustern und in der Vermittlung von ausgewählten Entwurfsmustern. Das Konzept und die Struktur der Lerneinheit wurde vorgestellt.

Die Medienobjekte der Lerneinheit werden mit verschiedenen Werkzeugen erstellt. Die Struktur der Lerneinheit wird technisch mit XML nach der DTD L3-XML festgelegt. Die Medienobjekte werden mit L3-XML oder mit XHTML in die Lerneinheit integriert.

Aus der Beschreibung in XML werden die Lehr- und die Lernansicht mit einem Werkzeug generiert. Aus den bisherigen Erfahrungen wurde die Notwendigkeit dieser beiden Sichten erkennbar.

Für einen Aufgabentyp, der typisch für eine Ausbildung in Entwurfsmustern ist, wurde ein Werkzeug entworfen und ein Prototyp realisiert.

Für des Projektjahr sind im wesentlichen die Fertigstellung der Lerneinheit, die Fertigstellung und Verfeinerung einer Werkzeuge und die Evaluation geplant.

Literatur

[AKT02] ARNOLD, PATRICIA, LARS KILIAN und ANNE THILLOSEN: *So lonely!? - Online-Betreuung als kritische Erfolgsbedingung beim telematischen Studieren*. In: *GMW Fachtagung Virtueller Campus 2002*. Waxmann Verlag, 2002.

[Geb02] GEBHARD, MARKUS: *Java2Html*. http://www.java2html.de/, 2002.

[HB02] HINZE, UDO und GEROLD BLAKOWSKI: *Anforderungen an die Betreuung im Onlinelernen - Ergebnisse einer qualitativen Inhaltsanalyse im Rahmen der VFH*. In: *GMW Fachtagung Virtueller Campus 2002*. Waxmann Verlag, 2002.

[HTH02] HARTWIG, R., C. TRIEBE und M. HERCZEG: *Software-ergonomische Evaluation im Kontext der Entwicklung multimedialer Lernmodule für die virtuelle Lehre*. In: HERCZEG, M., W. PRINZ und H. OBERQUELLE (Herausgeber): *Mensch & Computer 2002*, Seiten 313–322, 2002.

[LS02] LECON, CARSTEN und SILKE SEEHUSEN: *Combining Structure Search and Content Search for Online Courses*. In: TJOA, A.M. und R.R. WAGNER (Herausgeber): *13th International Workshop on Database and Expert Systems Applications: MIW'2002 - The 3rd International Workshop on Managment of Information on the Web - Web Based Teaching and Learning*, Seiten 366–370. IEEE, 2002. Aix-en-Provende, France, September 2-6 2002.

[SL01] SEEHUSEN, SILKE und CARSTEN LECON: *Entwicklung von Online-Kursen für den längerfristigen Einsatz*. In: *Informatik 2001 Jahrestagung GI/OCG, Workshop Virtuelle Lernräume*, Seiten 1131–1135, 2001.

[SLK00] SEEHUSEN, SILKE, CARSTEN LECON und CAY KABEN: *Specification of Learning Trails in Virtual Courses*. In: *FIE 2000, Frontiers in Education Conference, Kansas City, MO, USA*, 2000.

[SN02] SEEHUSEN, SILKE und NINA NISSEN: *Lerneinheit Entwurfsmuster - Jahresbericht 2001 MuSofT Teilprojekt 2.2*. In: *Ergebnisbericht des Jahres 2001 des Projektes MuSofT - Multimedia in der SoftwareTechnik*, Nummer MEMO Nr. 121 in *Internes Memorandum des Lehrstuhls für Software-Technologie*, Seiten 52–65. E.-E. Doberkat and G. Engels, Universität Dortmund, Fachbereich Informatik, 2002.

[W3C00] *Extensible Markup Language (XML)*. http://www.w3.org/XML/, 2000.

Der MuSofT-Jahresbericht 2002 des Teilprojekts 2.3

Peter Aschenbrenner, Andy Schürr

Inhaltsverzeichnis

1 Einleitung

Im Rahmen des MuSofT-Projektes hat das Teilprojekt 2.3 das Ziel, eine multimedial aufbereitete Lerneinheit zum Thema “Algorithmen und Datenstrukturen” zu entwickeln, die in verschiedenen Umfängen für ganz unterschiedliche Studiengänge eingesetzt werden kann. Will man dieses klassische Vorlesungsthema der Informatik multimedial unterstützen, so bietet sich insbesondere die Animation und Visualisierung der behandelten Standardalgorithmen und -datenstrukturen an. Wie bereits im letzten Jahresbericht beschrieben ([PA02]) findet man zwar heute - besonders im Internet - eine große Anzahl von fertigen Animationen und Werkzeugen, die solche Visualisierungen für einfache Datenstrukturen und Algorithmen anbieten oder deren Erstellung vereinfachen (z.B. [CCA01], [Ani01], [Sor01], [Bub01]).

Allerdings bestehen in nahezu allen Fällen Einschränkungen in der Ausnutzung des möglichen Potentials der multimedialen Lehre von Algorithmen und Datenstrukturen (wie fest vorgegebener Ablauf ohne flüssige Animationen oder völlig ohne Interaktion; fehlende Konfigurierbarkeit für neue Datenstrukturen).

Im Rahmen der Lerneinheit 2.3 wird deshalb ein neues Konzept für die Gestaltung von Lernmodulen im Bereich “Algorithmen und Datenstrukturen” entwickelt, das folgende Punkte unterstützt:

- Standard-Datenstrukturen und Algorithmen werden nach softwaretechnischen Gesichtspunkten mit graphischen Notationen (eine Teilmenge der UML) modelliert.
- Aus den erarbeiteten Modellen werden lauffähige Animationen generiert, die entweder als reine Präsentationen in Vorlesungen integriert ablaufen oder interaktives Arbeiten im Rahmen von Übungen oder dem Selbststudium erlauben.
- Das graphische Debugging selbst erstellter Programme von Studenten wird mithilfe von Visualisierungsbausteinen und automatischer Überprüfung von Invarianten von Datenstrukturen unterstützt.

Die Tätigkeiten im ersten Jahr umfassten u.a. den Test verschiedenster Animationskonzepte, fertiger Animationen und Animationsplattformen. Im Jahr darauf wurde auf der Basis der ausgewählten Plattformen ein neues Rahmenwerk aufgebaut und das AVL-Beispiel in diesem Rahmenwerk erstellt. Außerdem wurde in diesem zweiten Projektjahr die Vorlesung als modularer, auf verschiedene Arten zusammensetzbarer Foliensatz (nach internem Sprachgebrauch also in 'Level1-Version') entwickelt, eingesetzt und evaluiert, vorerst in der Programmiersprache Ada. Im dritten Jahr schließlich wird das Rahmenwerk vervollständigt und es werden weitere Animationen hinzukommen. Zusätzlich wird ein statischer Beispiel-Foliensatz der Vorlesung (intern als 'Level0-Version' bezeichnet) gebaut und entsprechende Teile auf Java umgestellt.

2 Vorstellung der Lerneinheit

Im Jahr 2002 wurde die Lerneinheit 2.3 als Vorlesung in Level1-Version (s.o.) realisiert und zwar konfigurierbar für Studenten der Elektrotechnik oder der Informatik (zu finden unter [Ein03]). Wie schon im letzten Jahresbericht ([PA02]) beschrieben werden dem Studierenden im wesentlichen folgende Lehrinhalte vermittelt:

- eine Auswahl der wichtigsten Standardalgorithmen und -datenstrukturen (insbesondere für Suchen und Sortieren auf Listen, Bäumen und Graphen)
- Prinzipien des Entwurfs von Algorithmen (wie "divide and conquer", "Greedy"-Algorithmen, dynamische Programmierung, ...)
- Programmiersprachliche Konstrukte imperativer und objektorientierter Art (z.B. Umgang mit Zeigerstrukturen, Ausnahmebehandlung zur systematischen Fehlerbehandlung etc.)
- Objekt- und Klassendiagramme der UML als graphische Hilfsmittel zum Entwurf von dynamischen Datenstrukturen
- die Idee der Datenabstraktion (Pakete, Schnittstellen etc.) und der Entwicklung generischer Programmeinheiten
- Invarianten und einfache Vor- und Nachbedingungen zur Beschreibung von Schnittstellen

- systematischer Test entwickelter Algorithmen und Datenstrukturen
- Floyd-Hoare-Kalkül zur Programmverifikation
- O-Kalkül zur Charakterisierung von Laufzeit- und Speicherplatzverhalten

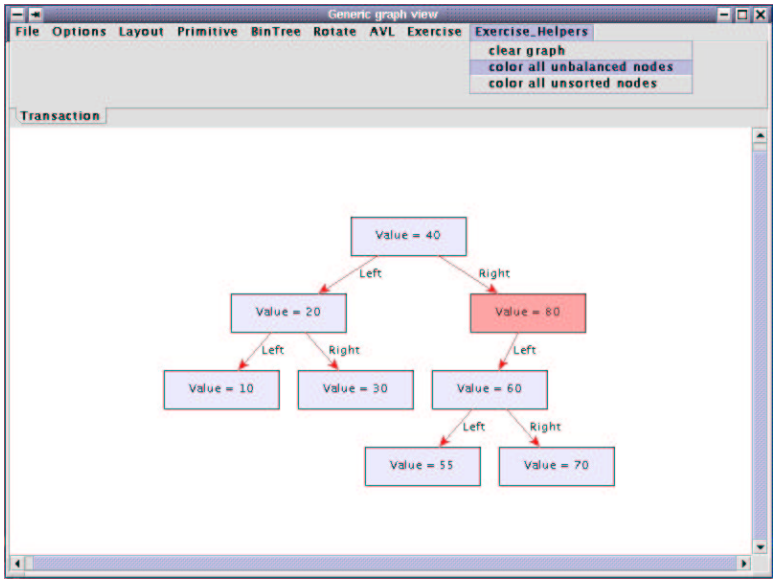


Abbildung 1: AVL-Baum, der die Balancierungs-Eigenschaft verletzt

Die Verwendung der zu unterrichtenden Datenstrukturen wird mithilfe einer fiktiven Speditionsfirma “Blitz AG” motiviert und demonstriert (siehe [PA02]).

Ergänzend dazu sollen in der zu entwickelnden Visualisierungsumgebung graphische Lernszenarien und interaktive Übungsbeispiele für AVL-Bäume, Graphenalgorithmen (die kürzeste Wegesuche und das Spannbaum-Problem) und die doppelt verkettete Liste zur Verfügung gestellt werden. Für den AVL-Baum wurde dieses Ziel - inklusive aller notwendigen Implementierungen für die Laufzeitumgebung - bereits im Jahr 2002 erreicht und der MuSofT-Projektleitung bei einem Treffen im Oktober in Darmstadt präsentiert.

So sieht sich der Lernende etwa einem bestehenden binären Baum gegenüber, der z.B. die Suchbaum-Eigenschaft oder die Balancierungs-Eigenschaft verletzt und muss diese mit Löschen und Wiedereinfügen geeigneter Knoten oder durch Rotationsoperationen wiederherstellen. Dabei unterstützen ihn Fehlermeldungen der Laufzeitumgebung und farbliche Markierung der Wurzelknoten fehlerhafter Teilbäume. Abbildung 1 zeigt eine Beispielsitzung mit einem AVL-Baum, der die Balancierungs-Eigenschaft verletzt; der Wurzelknoten des unbalancierten Teilbaums erscheint deswegen farblich markiert (im Bild etwas dunkler).

Oder er baut sich mit Hilfe mehr oder weniger mächtiger Einfügeoperationen (die weniger mächtigen erlauben es, Invarianten der Datenstruktur zu verletzen, um auch Fehlersituationen

entstehen zu lassen) selbst einen AVL-Baum zusammen und lernt dabei anschaulich dessen Verhalten kennen.

Weitere interaktive Aufgaben für die restlichen erwähnten Datenstrukturen werden im Jahr 2003 zusammen mit der dafür benötigten Funktionalität der Visualisierungsumgebung hinzukommen.

Eine besonders hervorzuhebende Eigenschaft der Lerneinheit 2.3 wird ihre Anpassbarkeit an neue Datenstrukturen (durch die Verwendung von Graphgrammatiken, vgl. Abschnitt 3.1) sein. Dadurch soll insbesondere der spätere Einsatz auch nach dem Ende von MuSofT gewährleistet werden.

Als weitere Besonderheit ist es geplant, die Laufzeitumgebung als graphischen Debugger für selbstgeschriebene Java-Programme der Lernenden einsetzen zu können. Dabei fügen diese in ihren Java-Code, der zum Beispiel eine Linksrotation in einem AVL-Baum (siehe Abbildung 2) durchführen soll, Corba-Aufrufe ein, die zu entsprechender Veränderung des visualisierten Graphen und dazu passenden (Fehler-)Meldungen führen.

3 Technische Realisierung

Wie schon im Abschnitt 2 beschrieben ist die Konfigurierbarkeit der Lerneinheit 2.3, besonders im Hinblick auf die Austauschbarkeit der jeweiligen zu unterrichtenden Datenstruktur in der interaktiven Visualisierungsumgebung, ein ganz zentrales Anliegen. Das bedeutet aber, dass eine technische Repräsentation der Datenstruktur, ihrer Schnittstellenoperationen und ggf. des Algorithmus existieren muss, die möglichst entkoppelt ist vom Rest des Systems und die nach Möglichkeit nahe an den sowieso genutzten Repräsentationen für Datenstrukturen liegt.

In einer Evaluation verschiedener Paradigmen und Tools für die Erzeugung von passenden Visualisierungen in der ersten Projektphase zeigte sich der Einsatz einer Graphtransformationsumgebung diesen Anforderungen besonders gewachsen. Die Gründe werden im nächsten Abschnitt beschrieben.

3.1 Graphtransformationen

In der herkömmlichen Lehre von Datenstrukturen in Informatiklehrbüchern werden Schnittstellenoperationen wie z.B. die Linksrotation in einem AVL-Baum (siehe Abbildung 2) oft als Aufeinanderfolge zweier Zustände dieser Datenstruktur dargestellt. Wenn man nun sowohl den Zustand der Datenstruktur vor der Schnittstellenoperation als auch den Zustand danach jeweils als Graph auffasst, kann man die Schnittstellenoperation auf ganz natürliche Weise als Graphtransformation interpretieren. Deswegen bieten sich Graphen und Graphtransformationen als Beschreibungsmittel für Datenstrukturen, Schnittstellenoperationen und Algorithmen und damit als Spezifikationsmittel für unsere Visualisierungsumgebung innerhalb der Lerneinheit 2.3 an (bezüglich des Stichwortes der Spezifikation sei noch gesagt, dass man im Sinne der UML jede Graphtransformation als Paar von Objektdiagrammen auffassen kann).

Um eine (halb-)automatische Generierung der Visualisierungsumgebung aus diesen Spezifikationen zu ermöglichen, muss es eine Laufzeitumgebung geben, die aus einer solchen Spezifikation mithilfe von Namenskonventionen, weitergehender Konfigurierung (z.B. die Ein-

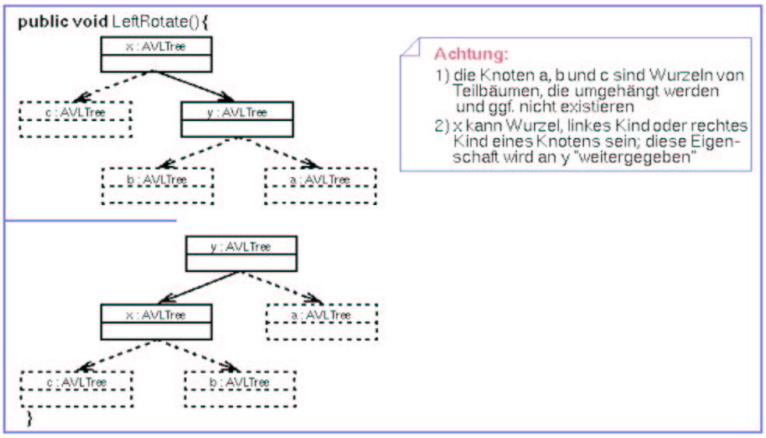


Abbildung 2: Linksrotation in einem AVL-Baum

stellung eines zu verwendenden Layoutalgorithmus) und vorgegebener Übungsbeispiele eine Art interaktiven Graphbrowser realisiert, in dem die Lernenden dann den jeweiligen Zustand der zu unterrichtenden Datenstruktur sehen und manuell oder programmatisch verändern und manipulieren können.

Das Java-Rahmenwerk UPGRADE2 unterstützt die Erstellung einer solchen Laufzeitumgebung basierend auf dem Graphtransformationssystem PROGRES. Beide werden im folgenden Abschnitt kurz vorgestellt.

3.2 Rahmenwerk

Im gewählten Graphtransformationswerkzeug PROGRES ([PRO01]) können die Datenstrukturen, ihre Schnittstellenoperationen und ggf. die Algorithmen auf graphischem (oder auch textuellem) Wege wie im Abschnitt 3.1 gefordert spezifiziert werden. Eine Beispielspezifikation der Linksrotation ist in Abbildung 3 zu sehen.

Wie bereits beschrieben ist einer der Hauptgründe für die Verwendung eines Graphtransformationswerkzeugs die Erweiterbarkeit und Wiederverwendbarkeit unserer Lerneinheit dank der naheliegenden visuellen Notation bei der Definition neuer Datenstrukturen und Algorithmen. Selbstverständlich muss auch hierbei mit einem gewissen Einarbeitungsaufwand eines hierin evtl. noch unerfahrenen Dozenten in die gewählte Graphtransformationsumgebung, hier also PROGRES, gerechnet werden, der aber unserer Meinung nach wesentlich geringer ist als z.B. bei einer Erweiterung eines vergleichbaren reinen Java-Visualisierungssystems für neue Datenstrukturen; zudem erleichtert unser Ansatz eine Erweiterung oder Anpassung durch die saubere Trennung zwischen Spezifikation der Datenstruktur und Implementierung der Visualisierung. Für den Einsatz bereits bestehender Instanzen unserer Visualisierungsumgebung sind dagegen keinerlei Kenntnisse von Graphtransformationen oder eines entsprechenden Werkzeugs notwendig.

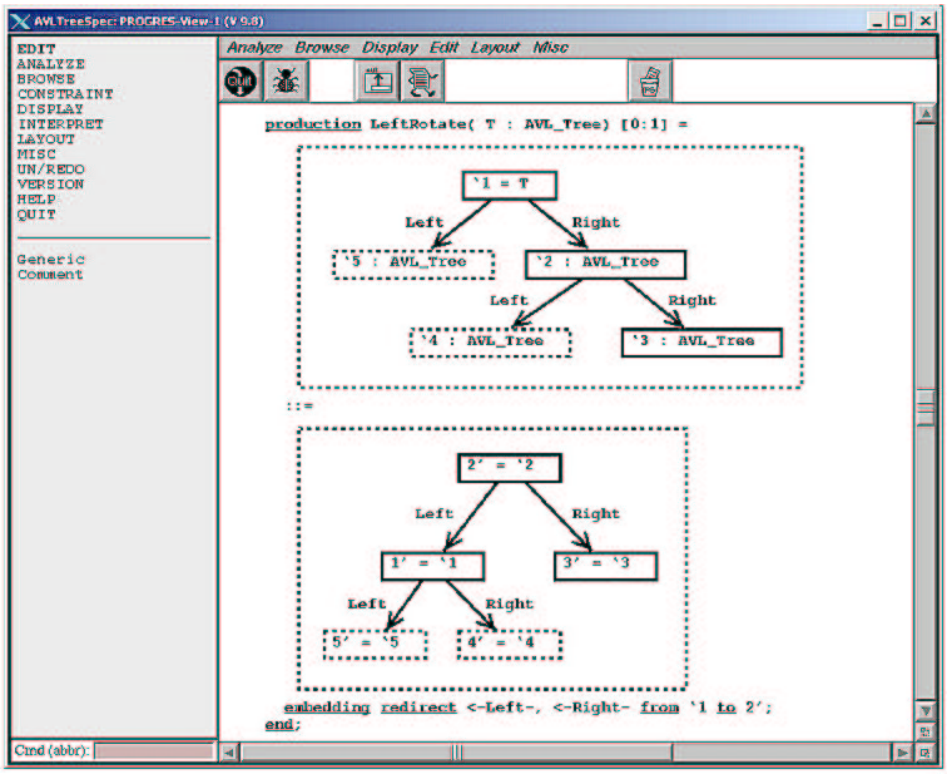


Abbildung 3: Spezifikation der Linksrotation in PROGRES

Als Java-Rahmenwerk für den Bau der Laufzeitumgebung wurde UPGRADE2 gewählt ([UPG02]), weil hier schon ein automatischer Prozess der Generierung von Java-Datenstrukturen aus Graphspezifikationen implementiert ist. Außerdem stehen einige hilfreiche Basisoperationen wie die einfach konfigurierbare Ausführung definierter Graphtransformationen aus dem Menü und die automatische Anzeige von Parameterfenstern für Graphtransformationen zur Verfügung.

Für die letztendliche Darstellung der Graphen wird innerhalb von UPGRADE2 noch eine Java-basierte Graphvisualisierungsbibliothek benötigt. Hier gibt es eine bereits ziemlich aufwendig ausprogrammierte Lösung, die auf ILog's JViews ([JVi01]) basiert. Aus verschiedenen Gründen (u.a. Lizenzgründe und die Notwendigkeit, kontinuierliche Animationen im Zusammenhang mit Layoutalgorithmen darstellen zu können) haben wir uns im Teilprojekt 2.3 entschieden, anstatt der JViews-Lösung eine eigene mithilfe der kommerziellen, ursprünglich an der Uni Tübingen entwickelten Graphenbibliothek yFiles ([yfi01]) zu entwickeln.

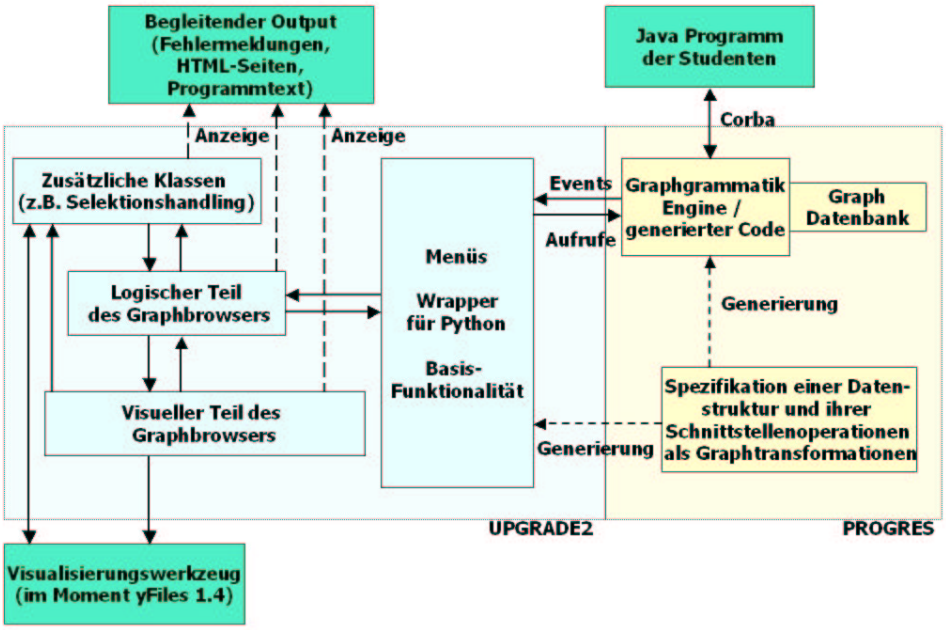


Abbildung 4: Aufbau der Visualisierungsumgebung der Lerneinheit 2.3

3.3 Aufbau

Das Zusammenspiel der bisher schon beschriebenen Komponenten wird in Abbildung 4 überblicksartig dargestellt.

Innerhalb des großen Kastens sind diejenigen Komponenten rechts dargestellt, die mit PROGRES realisiert sind, während die zu UPGRADE2 gehörigen Teile sich links befinden.

Im folgenden soll anhand Abbildung 4 kurz das Prinzip des Konfigurationsprozesses einer neuen Instanz der Visualisierungsumgebung der Lerneinheit 2.3 beschrieben werden: Als erstes spezifiziert der Dozent, der die Lerneinheit einsetzen will

- eine Datenstruktur wie etwa den AVL-Baum als PROGRES-Knotentyp,
- wichtige Schnittstellenoperationen (Einfügen eines neuen Elementes, Linksrotation, ...) als Graphtransformationen
- und für Übungsaufgaben benötigte Hilfsprozeduren auf dem AVL-Baum (wie z.B. den Test 'IsBalanced', der später verwendet werden kann, um die Lernenden die Balancierungs-Eigenschaft eines Teilbaums manuell überprüfen zu lassen) als PROGRES-Produktionen oder -Tests.

Aus dieser Spezifikation wird als nächstes in einem automatisierten Verfahren C-Code generiert, der sowohl die Basis für eine weitere Generierung von UPGRADE2-Datenstrukturen für Menüs etc. darstellt als auch die notwendige Basis für die PROGRES-Laufzeitumgebung, die später die Graphtransformationen ausführt, in der Graphdatenbank speichert, Aufrufe von UPGRADE2 entgegennimmt und Events für UPGRADE2 generiert.

Die restlichen Komponenten seien anhand eines kleinen Beispielablaufs charakterisiert: Wenn der Lernende etwa im Rahmen einer Übung zum manuellen Aufbau eines AVL-Baumes einen gerade neu erzeugten Knoten mithilfe einer Produktion 'AddLeft' als linken Sohn eines schon im Baum befindlichen Knotens einfügen will, kann er zuerst den zukünftigen Vater-Knoten mit der Maus markieren, wobei der Selektionsmanager (als eine der 'zusätzlichen Klassen' in Abbildung 4) über eine yFiles-Funktionalität den markierten Knoten identifiziert, mithilfe des logischen Teils des Graphbrowsers in eine PROGRES- / UPGRADE2-Knotennummer umrechnet und für die weitere Verwendung in einer entsprechenden UPGRADE2-Datenstruktur vormerkt. Das gleiche passiert nun mit dem neuen linken Sohn. Unter der Bedingung, dass die vorher erwähnte PROGRES-Produktion 'AddLeft' zwei Knoten als Parameter erwartet und zwar zuerst den Vater und dann den neuen linken Sohn, kann der Lernende nun im Menu 'AddLeft' aufrufen. In diesem Fall verpackt die Basisfunktionalität des generierten UPGRADE2 Prototypen diesen Aufruf in einen Aufruf der entsprechenden PROGRES-Produktion und übergibt gleich die zugehörigen Parameter. Die Graphgrammatik-Engine führt die Graphtransformation auf ihrer Datenbank aus und liefert dementsprechende Events zurück, etwa im Erfolgsfall ein 'AddEdge'-Event für die neu hinzukommende Kante. Der Layoutmanager (enthalten im visuellen Teil des Graphbrowsers) erfährt davon mithilfe der Basisfunktionalität des Prototypen und führt mit Ausnutzung von yFiles-Funktionalität einen flüssigen Animationsschritt aus, in dem der neue linke Sohn an seinen ihm nun zustehenden Platz im Graphen (links unter seinem Vater) bewegt wird.

Im Fehlerfall wäre im Rahmen des begleitenden Outputs eine Fehlermeldung ausgegeben worden und der Graph hätte sich nicht verändert.

Die bisher nicht beschriebene Komponente 'Java-Programm der Studenten' bezieht sich auf die geplante Möglichkeit, mithilfe dieser Visualisierungsumgebung ein graphisches Debugging eigener Programme der Lernenden durchzuführen.

Diese Funktionalität genauso wie die Möglichkeit, Python Skripte für den automatischen Ablauf ganzer Beispiele zu nutzen, ist für 2003 geplant.

4 Evaluierung

Die erste Fassung der Lerneinheit 2.3 kam bereits im Wintertrimester 2002 an der UniBw München zum Einsatz. Anstelle von Java (der Sprache, auf die man sich in MuSofT geeinigt hat) wurde in der Vorlesung "Einführung in die Informatik II" jedoch die Programmiersprache ADA verwendet. Das heißt, dass die Lerneinheit so konzipiert ist, dass alle programmiersprachlichen Beispiele in einfacher Weise austauschbar sind. Als kleinster gemeinsamer Nenner der ADA und Java zu Grunde liegenden Programmierparadigmen wird deshalb eine objektbasierte Softwareentwicklung propagiert, die sich in beiden Programmiersprachen in sinnvoller Weise umsetzen lässt.

In der beschriebenen Vorlesung wurden bereits Animationen eingesetzt, die allerdings noch

meistens passiv oder auf relativ einfache Interaktionsbeispiele mit Sortieralgorithmen beschränkt und im Internet frei verfügbar waren. Zusätzlich wurde dort das Konzept der Erläuterung von Datenstrukturen und Algorithmen mit UML- Klassen- und Kollaborationsdiagrammen erprobt.

Die Evaluation geschah durch die Verwendung eines Standardfragebogens, dessen Auswertung ungefähr folgendes Resultat ergab: Bei den Informatikstudenten herrschte eine positive Resonanz vor, die Elektrotechnikstudenten wurden selbst durch die einfachere Level1-Version überfordert und hätten gerne noch wesentlich mehr Animationen gehabt. Die Klausurergebnisse lassen sich mit denen der Vorjahre nicht vergleichen, da sich der Vorlesungsstoff stark geändert hatte.

Der weitere Einsatz, auch an anderen Universitäten, z.B. den Projektpartnern ist in Diskussion.

Ein zweiter Punkt betrifft die Evaluierung der in Abschnitt 3 beschriebenen Visualisierungsumgebung. Da diese jetzt einen Stand erreicht hat, der eine erste Evaluation möglich macht, ist diese so bald als möglich geplant.

Im Gespräch sind sowohl Einsatz / Evaluierung im Rahmen einer Übungsgruppe zur Vorlesung “Einführung in die Informatik II” im Wintertrimester 2003 an der Universität der Bundeswehr München als auch im Rahmen eines Programmier-Praktikums im Sommersemester 2003 an der Technischen Universität Darmstadt.

In beiden Fällen sollen einige Übungsbeispiele mit AVL-Baum, kürzester Wegesuche auf Graphen und Suche des minimalen Spannbaums eines Graphen durchgeführt werden und der Motivationsgewinn und der Lerneffekt auf geeignete Art geprüft werden.

Selbst eine zusätzliche Evaluierung kurz nach dem Projektende Anfang 2004 mit der voll funktionsfähigen Visualisierungsumgebung ist im Bereich der Möglichkeit.

Die erwähnte Evaluierung an der UniBw München wird vorraussichtlich auf folgende Art geschehen:

- Eine Teilmenge der Übungsgruppen arbeitet mit Animationen, eine Vergleichsgruppe ohne dieses Hilfsmittel.
- Eine Klausuraufgabe über eine (oder beide) der Datenstrukturen, für die eine interaktive Animation zur Verfügung stand, wird gestellt und ein spezieller, kurzer Fragebogen ausgeteilt, der die Studenten nach ihrem subjektiven Eindruck bzgl. des Nutzens der Animationen fragt.
- Dann werden die Klausurergebnisse mit dem Fragebogen verglichen.

5 Planung für das nächste Jahr

Im Jahr 2003 wird MuSofT zum Abschluss kommen. Die im Rahmen der Lerneinheit 2.3 anstehenden Aufgaben können zusammengefasst werden als

1. Hinzufügen von Funktionalität zur interaktiven Visualisierungsumgebung (u.a. Kanten als Parameter für PROGRES-Transaktionen, Anpassung der Standardfehlermeldungen für PROGRES-Transaktionen, Nutzung neuer Konfigurationsdateien für bisher hartkodierte Funktionalität)

2. Verfeinerung der Beispielspezifikationen für die Datenstrukturen doppelt verkettete Liste und AVL-Baum; Neuerstellung von Spezifikationen für Graphen und für die Algorithmen kürzeste Wegesuche und minimaler Spannbaum

3. Erstellen von Übungsaufgaben für die gerade erwähnten Algorithmen und Datenstrukturen

4. Empfehlungen für den Aufbau neuer Spezifikationen (auch für neue Datenstrukturen) und für die Struktur von Übungsaufgaben

5. Definition eines Konfigurationsprozesses für die Generierung neuer Instanzen der Lerneinheit 2.3 aus solchen neuen Spezifikationen (Konfigurationsdateien wie musoft_config.xml, neue Unix Skripte, Regeln für den Aufbau neuer Spezifikationen wie z.B. Namenskonventionen für Sichtbarkeit im Menü, hartkodierte Transaktionsnamen wie MakeNode(...), . . .)

6. Nutzung der Visualisierungsumgebung als graphischer Debugger durch die Implementierung einer Corba-Schicht in die UPGRADE2-Extension

7. Implementierung eines Java-Sliders zum passiven Anschauen und vor- und zurückspulen ganzer vordefinierter Abläufe durch die Erweiterung des bisher vorhandenen Python-Interpreters in UPGRADE2

8. Einreichung von Beiträgen für Konferenzen wie evtl. die HCC 2003, die UML 2003, die Appligraph 2003

9. Übertragung ausgewählter Algorithmen von Ada nach Java in der Level0-Fassung der Lerneinheit 2.3

6 Zusammenfassung

Bisher wurden die Lerninhalte der Lerneinheit 2.3 - auch in ihrer Struktur und ihren Abhängigkeiten - spezifiziert und nach einer ausführlichen Evaluierung einer Menge von Werkzeugen zur Animationsunterstützung zwei Kandidaten gefunden: PROGRES und yFiles. Diese wurden - zusammen mit UPGRADE2 - benutzt, um einen ersten Prototypen der Visualisierungsumgebung herzustellen, der auf die Erstellung, Manipulation und Animation von AVL-Bäumen zugeschnitten ist.

Die wichtigsten nächsten Schritte sind die Erweiterung dieses Prototypen um Funktionalität, die für weitere Klassen von Datenstrukturen wie verschiedene Graphalgorithmen und verkettete Listen gebraucht wird, der Einsatz und die Evaluation des Prototypen und die Erstellung eines Konzeptes für Konfigurierung neuer Instanzen des Prototypen für neue Datenstrukturen und Algorithmen.

Literatur

[Ani01] http://www.informatik.uni-siegen.de/db/animations.php3?lang=en, Dez. 2001.

[Bub01] http://olli.informatik.uni-oldenburg.de/fpsort/Animation.html, Dez. 2001.

[CCA01] http://www.cs.hope.edu/~alganim/ccaa/, Dez. 2001.

[Ein03] http://www2.informatik.unibw-muenchen.de/Lectures/WT2002/INF2/index.html, Jan. 2003.

[JVi01] http://www.ilog.com/products/jviews/, Dez. 2001.

[PA02] P. ASCHENBRENNER, A. SCHÜRR: *MEMO Nr. 121, MuSofT-Bericht Nr.1*. Lehrstuhl für Software-Technologie, Fachbereich Informatik, Universität Dortmund, Seiten 66–77, Feb. 2002.

[PRO01] http://www-i3.informatik.rwth-aachen.de/research/projects/progres/, Dez. 2001.

[Sor01] http://www.db.fmi.uni-passau.de/Sommercamp2001/Unterlagen/SortDemo.html, Dez. 2001.

[UPG02] http://www-i3.informatik.rwth-aachen.de/upgrade/, Dez. 2002.

[yfi01] http://www.yworks.de/, Dez. 2001.

MuSofT- Arbeiten zu LE 3.1 “V-Modell“ und LE 3.2 “Qualitätsmanagement“ in 2002

Fritz Schmidt

Kurzfassung
Die Lehreinheiten 3.1 und 3.2 des Projektes MuSofT befassen sich mit Qualitätsmanagement(QM) bei der Entwicklung großer Softwaresysteme. Schwerpunkte sind prozessorientiertes (LE 3.1) und produktorientiertes (LE 3.2) QM. Als Produkte werden Softwaresysteme aus dem Ingenieurbereich mit stark simulationsorientierten Komponenten zugrundegelegt. Die Möglichkeiten multimedialen Arbeitens werden derart genutzt, dass die Lernmodule primär einführenden Charakter haben und durch Hyperlinks mit Systemen verbunden werden, die mittels Tailoring auf aktuelle Projekte angepasst werden können. Dadurch können die Lernmodule in den verschiedensten Kontexten wiederverwendet werden. Die Lernmodule werden ergänzt durch Beispielanwendungen, die mit den Systemen erzeugt wurden.

Inhaltsverzeichnis

1 Einleitung

Die Lernmodule der Einheiten LE 3.1 “V-Modell“ und LE 3.2 “Qualitätsmanagement“ begreifen Software als technisches Produkt. Technische Produkte müssen Qualitätsforderungen genügen. Diese Forderungen müssen nicht nur aufgestellt, sondern auch erfüllt werden. Um dies zu erreichen, ist Qualitätsmanagement notwendig. Man unterscheidet prozessorientiertes

(getrieben von Vorgehensmodell und darin beschriebenem Entwicklungsprozess) und produktorientiertes (getrieben von Qualitätsmerkmalen und Teststrategien) Qualitätsmanagement.

Im Projekt werden insgesamt 14 Lehrmodule entwickelt. Zielgruppe sind Ingenieure der Fächer “Angewandte Informatik“, “Maschinenbau“ und “Simulation“. Die Lehrmodule können zum Teil einzeln verwendet und in andere Vorlesungen eingebaut werden oder aber als eigenständige Lehreinheit Verwendung finden. Sie sind zunächst für Präsenzveranstaltungen gedacht, sollen es aber auch erlauben, Kenntnisse aus vergangenen Vorlesungen aufzufrischen. Die ersten Versuche mit dem Einsatz einzelner Lehrmodule hat ergeben, dass bei Studierenden des Ingenieurwesens eine Kombination aus theoretischen Einführungen und praktischem Üben zu den besten Lehrerfolgen führt. Dem wird in den Lehrveranstaltungen im Jahr 2003 Rechnung getragen werden.

Die Inhalte der Lehreinheiten basieren primär auf unseren Erfahrungen bei der Durchführung von Qualitätsmanagement-Maßnahmen bei großen Industrie Projekten. Die Form orientiert sich dabei ebenfalls am Wunsch unserer Studierenden sich den Inhalt weniger über theoretische Präsentationen als vielmehr durch praktisches Erproben anzueignen. Die Präsentationen werden daher ergänzt um frei verfügbare Entwicklungsumgebungen, mittels derer im Rahmen von Praktika die QM Technologien erprobt und bei Studien- und Diplomarbeiten genutzt werden können. Weitere wichtige multimediale Elemente sind Hyperlinks auf weiterführendes Material, Multiple Choice Elemente zur Überprüfung des Wissensstandes und dynamisierte Wissensvermittlungen in Form von Animationen, Filmen und Simulationen technischer Vorgänge.

2 Vorstellung der Lerneinheiten “V-Modell“ und “Qualitätsmanagement“

Der Stoff wird in 2 Lerneinheiten mit insgesamt 14 Lehrmodulen präsentiert. Die Inhalte wurden im Jahresbericht 2001 beschrieben. Der aktuelle Stand der Entwicklung geht aus Tabelle 1 hervor.

Im Sinne der MuSofT Typisierung sind die Lernmodule den Leveln 0 und 1 zuzuordnen, die Lernmodule haben also vor allem Übersichtscharakter und führen in die Themengebiete ein. Dabei stehen pro Lehrmodul ca. 50 Folien zur Verfügung, die teilweise untereinander vernetzt sind und häufig auf weiterführende Informationen verweisen. Da für eine Lehreinheit von 90 min in der Regel ca. 30 Folien ausreichend sind, ist der Lehrende gehalten entsprechend den Interessen der Lernenden Schwerpunkte zu setzen und eine Auswahl zu treffen.

Im Rahmen der Lehreinheiten von MuSofT setzen wir multimediale Technologien so ein, dass die Studierenden ihre Lernfortschritte eigenständig kontrollieren und gezielt verbessern können (Ergänzung des Stoffes in Breite und Tiefe und Unterstützung des Lernens durch eigenes Experimentieren). Insbesondere verzichten wir weitgehend auf selbsterklärende Animationen. (Ziel: Unterstützung in Präsenzveranstaltungen)

Unser methodisch - didaktisches Vorgehen ist zunächst sequentiell lehrgangsmässig. Es wird ergänzt um eine hypermediale Erkundungsumgebung, die zum einen eine partielle Vertiefung und zum anderen eine problem- und prozessorientierte Annäherung an den Stoff ermöglicht.

LE	LM	Titel	Status	Freigabe
3.1	1	Fehler und ihre Kosten	T1	Ende WS 02/03
3.1	2	Prozeßqualität und Produktqualität	T1	Ende WS 02/03
3.1	3	Das Capability Maturity Modell	T1	Ende WS 02/03
3.1	4	Das V-Modell im Überblick	T1	Ende WS 02/03
3.1	5	V-Modell - Anwendungen	I	Ende WS 02/03
3.1	6	Prozessbeschreibung SSDA	T1	Ende WS 02/03
3.1	7	Der Rational Unified Process im V-Modell	T1	Ende WS 02/03
3.2	8	ISO-Normen	E	Ende SS 03
3.2	9	Prozessmodelle Übersicht	T1	Ende SS 03
3.2	10	Prüfung von SW-Komponenten - Einzeltests	E/I	Ende SS 03
3.2	11	Prüfung von komponentenbasierten Systemen - Integrationstests	E/I	Ende SS 03
3.2	12	Prüfung von Simulationsprogrammen - Funktionstests	I	Ende SS 03
3.2	13	Testumgebungen - Abnahmetests	E/I	Ende SS 03
3.2	14	Risikomanagement	I	Ende SS 03

Tabelle 1: Stand der Arbeiten bei den Lernmodulen der Lehreinheiten LE 3.1 und LE 3.2
E: Entworfen; I: Implementiert Ti: Getestet in Iteration i; F: Freigegeben

Um dies zu erreichen, benötigen wir kleine, relativ gut gekapselte Lernmodule und ein Konzept zu ihrer Integration in größere Lehreinheiten. Die Kapselung der Lernmodule erreichen wir durch die schon erwähnte Beschränkung auf die Level 0 und 1. Die Vertiefung wird dann über Hyperlinks zu Vorlesungen von Partnern aus dem Projekt MuSofT und zu Systemen, die sich auf aktuelle Probleme anwenden lassen, erreicht. Dies erfordert eine klare Trennung zwischen Lernmodulen und Anwendungsumgebungen auf der einen und Lehrinhalten und deren Präsentation auf der anderen Seite. Diese Trennungen ermöglichen es, die Lernmodule in verschiedenen Kontexten zu verwenden und durch Wahl der Beispiele oder gezielte Vertiefungen einzelner Unterthemen, an die Interessen und Möglichkeiten der Lernenden anzupassen. Zur Unterstützung der Selbstkontrolle des Lernfortschrittes der Lernenden werden multiple choice Fragebögen zur Verfügung gestellt.

3 Technische Realisierung

Bei der technischen Realisierung setzten wir bisher Standardwerkzeuge sowohl zur Erstellung des Lehrmaterials als auch zum Zugriff darauf ein. Im wesentlichen sind das MS Office Tools und Web Browser. Mit der Plattform AIDA (Active Information Development Assistant des IAS der UNI Stuttgart `http://www.ias.uni-stuttgart.de/de/start_forschung.html`) hatten wir eine Umgebung, die die Präsentation der Materialien für die Studierenden unterstützte.

AIDA wurde basierend auf der am IAS entwickelten Methode Aktive Informationspräsentation (AIP) entwickelt. Es wird im BMBF-Projekt ITO - Information Technology Online- wei-

terentwickelt und eingesetzt. AIDA ermöglicht die Erstellung von Multimedia-Vorlesungen, indem aus verschiedenen multimedialen Einzelelementen eine integrierte Vorlesungspräsentation erstellt wird. Über Internet kann der Lernende auf alle multimedialen Lehrmaterialien zugreifen, wobei er außer dem Web-Browser kein weiteres Werkzeug benötigt. Als Eingangsinformationen können u.a. mit Office-Werkzeugen erstellte Dokumente, Grafiken verschiedener Formate sowie Videos im AVI-Format verwendet werden. Unterstützende Dienste, wie etwa die Erstellung von Indizes, erleichtern dabei die Arbeit. Bei der Erstellung der Informationsapplikation werden die Eingangsinformationen ins HTML-Format (Hypertext Markup Language) transformiert und miteinander verknüpft. Leider ist dabei noch viel händische Arbeit nötig, so dass es zunehmend schwerer wurde, die Idee einer hörerangepssten Auswahl der Lehrmaterialien effektiv und zeitnah umzusetzen. Wir haben deshalb beschlossen eine alternative Lösung auf Basis des Frameworks Cocoon aufzubauen. Darüber wird in Abschnitt 3.2 weiter berichtet.

Wir haben auch gelernt Animationstechniken eher spärlich einzusetzen. Ein Zuviel an Animation ermüdet die Studierenden und lenkt ihre Konzentration statt auf die fachlichen Inhalte auf die speziellen Effekte.

Sehr stark setzen wir auf die Einbindung von Werkzeugen zur Erprobung des gelehrten Stoffes. Dies geschieht bevorzugt durch Hyperlinks auf Demoumgebungen, Vorlesungen und Stoffsammlungen anderer Kollegen und zu vertiefenden Texten. Ein Verbund wie etwa MuSofT wird da zu einer besonders sprudelnden Quelle. Wir haben Beispiele für unseren Einsatz multimedialer Techniken an Hand existierender Vorlesungen erstellt. Man findet sie im Netz auf unserer Vorlesungsseite (`http://ikeas1.ike.uni-stuttgart.de/~www_wn/lehre/`)

- Vorlesung 1: Simulation komplexer technischer Anlagen
  - Kurzbeschreibung **(Beispiel Video)**
  - Vorlesungsmaterial
- Vorlesung 2: Komponentenbasierte Softwareentwicklung
  - Kurzbeschreibung **(Beispiel: Animation)**
  - Vorlesungsmaterial
- Vorlesung 3: Numerische Methoden
  - Kurzbeschreibung **(Beispiel: Simulation)**
  - Vorlesungsmaterial

Aus diesen Beispielen geht auch hervor, dass wir mit unseren Arbeiten nicht nur in den Kontext von MuSofT eingebunden sind, sondern auch die vielfältigen Erfahrungen von Kollegen der Universität Stuttgart und die von der Universität angebotenen Dienste nutzen. Wir nehmen dazu an den Projekten “100-online“ und “selfstudy“ teil und arbeiten im Arbeitskreis “BMBF Projekte“, in dem alle im Rahmen des Projektes Neue Medien in der Bildung tätigen Institute der Universität zusammengeschlossen sind und ihre Erfahrungen austauschen, mit. Erst wenn wir in diesen Umfeldern keine oder zu wenig an unsere Bedürfnisse angepasste

Dienste zur Verfügung gestellt bekommen, gehen wir eigene Entwicklungen an, die wir aber auch wieder einstellen können, wenn sich bessere Lösungen am open source Markt finden. Drei Beispiele mögen dies erläutern.

3.1 Einbindung numerischer Experimente zur Untersuchung des Verhaltens numerischer Modelle

Technische Modelle basieren auf den Erhaltungssätzen für Masse, Impuls und Energie. Mathematisch werden diese Sätze häufig als partielle Differentialgleichungen modelliert. Aus diesen werden über eine Diskretisierung des Ortsraumes Systeme gewöhnlicher Differentialgleichungen generiert, die dann numerisch gelöst werden. Die Überführung des mathematischen Modelles in ein numerisches Modell und seine Implementierung in ein konkretes Programm auf einem konkreten Rechner erfordert Abbildungsschritte, die mit Fehlern behaftet sind

Untersuchung der Auswirkung solcher Fehler auf die Lösung eines konkreten Problems und die Bestimmung der sie beeinflussenden Größen kann Ingenieuren am leichtesten über eigene Erfahrungen vermittelt werden. Wir bieten deswegen in Experimenten typische numerische Verfahren an und erlauben den Studierenden auszuprobieren, wie sie deren Verhalten durch geschickte Wahl von kritischen Parametern beeinflussen können. Um dies auch über das Internet machen zu können, wurden die Experimente über CGI-Skripte eingebunden. Dies ist in Abb. 1 dargestellt.

3.2 Entwicklung einer Präsentationsplattform auf Basis von Cocoon

Das Vorgehen bei der Einbindung von Experimenten in eine Vorlesung genau so, wie die Präsentation von Lehrmaterial in einer vom Verlauf einer Vorlesung abhängigen Form in der Plattform AIDA erfordert sehr viel händische Arbeit, die im Laufe einer Vorlesung immer wieder von anderen Menschen geleistet wird und daher anders ausfällt. Das macht eine aktuelle Präsentation uneinheitlich oder aber über die Maßen aufwendig. Wir suchten daher nach einem alternativen Werkzeug und haben nach Durchführung einer Nutzwertanalyse Cocoon ausgewählt. (Details siehe `http://ikeas1.ike.uni-stuttgart.de/~www_wn/lehre/`) (Cocoon Projekt Homepage `http://xml.apache.org/cocoon/index.html`)

Cocoon ist ein sehr flexibles Web-Publishing-Framework und ein Teil des Apache XML-Projekts. Es ist aufgebaut aus wiederverwendbaren Komponenten mit einem einfachen Interface zwischen den Komponenten. Die Architektur von Cocoon beruht auf einem Modell, das die Entwickler von Cocoon als Pyramidenmodell (pyramid model) bezeichnen. Es wird eine klare Trennung der Arbeitsbereiche vorgenommen, die als Separation of Concerns (SoC) bezeichnet wird, aus einer strikten Trennung von Inhalt, Logik und Darstellung besteht und vom Management gesteuert wird (siehe Abb. 2).

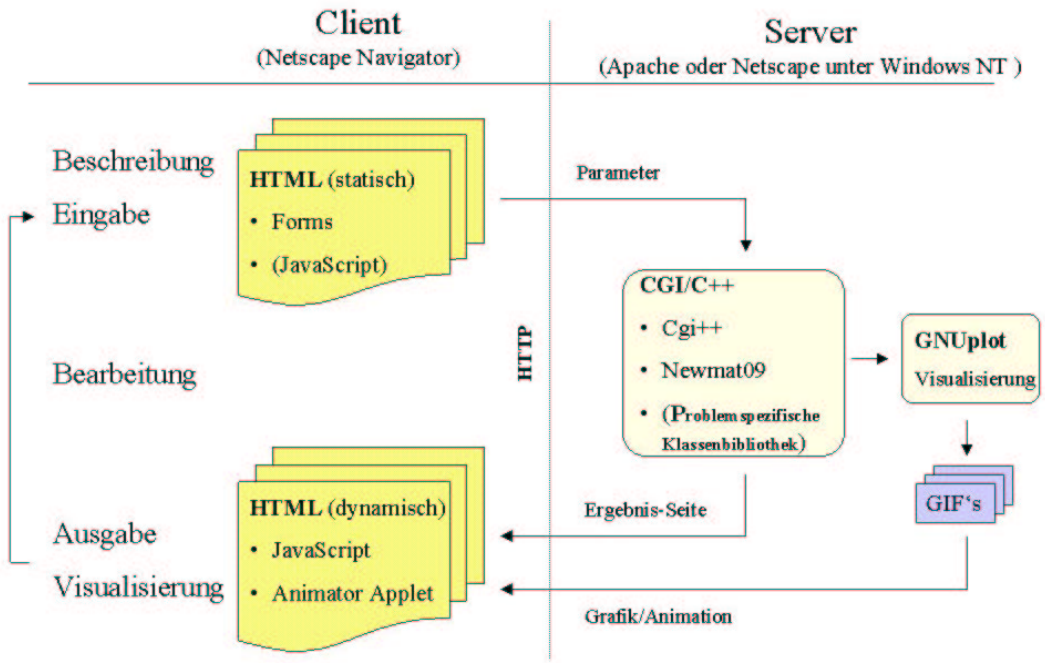


Abbildung 1: Numerische Experimente im WWW

Die Aufgaben des SoC sind in folgende vier Arbeitsbereiche gegliedert:

- Management
  - Inhalt
  - Ablauf
  - Aufbau
- Content
  - Redaktioneller Teil aller Inhalte (eigentlicher Inhalt)
  - Inhalt kann aus unterschiedlichen Datenquellen bezogen werden
- Logic
  - Integration der dynamischen Inhalte
- Style
  - Präsentation
  - Look and Feel

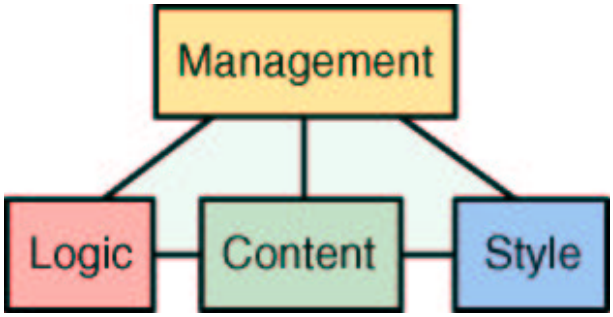


Abbildung 2: Separation of Concerns (SoC) in Cocoon

– Grafik

Die Vorteile dieser klaren Trennung der einzelnen Zuständigkeiten auf verschiedene Arbeitsbereiche und damit auch auf einzelne Komponenten liegen darin, dass erforderliche Änderungen ohne großen Aufwand innerhalb einer Komponente verwirklicht werden können. Es ist sogar ein kompletter Austausch einer Komponente nach bestimmten Regeln möglich, ohne dass andere davon betroffen sind. Der Austausch einer Komponente muss vom Management gesteuert werden.

Cocoon verarbeitet Dokumente über Pipelines. Ein XML-Dokument wird dazu in einer Datei gespeichert (file.xml), durch einen XSL-Transformator mit Hilfe eines XSL-Stylesheet-Dokumentes (stylesheet.xsl) transformiert und danach als HTML-Datei ausgegeben. Diese Cocoon-Pipeline ist in der folgenden Abbildung dargestellt.

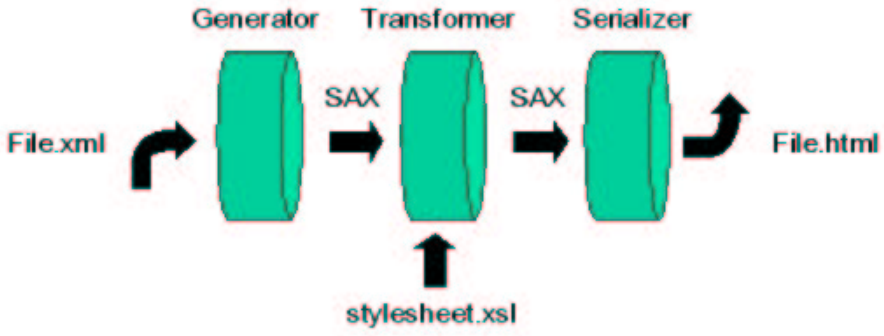


Abbildung 3: Informationsverarbeitung in Cocoon

Dazu gibt es drei grundlegende Komponenten in Cocoon (siehe Abb. 3).

1. Generatoren sind verantwortlich für das Erstellen eines SAXStreams. (SAX Project

http://www.saxproject.org/) Dieser SAX-Stream besteht aus wohlgeformtem XML-Inhalt und kann beispielsweise aus einer Datei stammen, oder aus einem Binary Large Object, einer Datenbank oder einem anderen, externen System, das über eine URI angesprochen wird, ausgehen. Die Daten können also von einer beliebigen Datenquelle kommen, sofern der Generator mit dieser umgehen kann.

2. Transformer sind für die Modifikation des XML-Streams verantwortlich. In der Praxis werden hauptsächlich Transformer wie XSL, SQL, SOAP und LDAP eingesetzt. Es ist aber auch die Verwendung von selbst erstellten Transformern möglich. Die primäre und damit wichtigste Anforderung an einen Transformer besteht darin, dass er einen XML-Stream als Eingabe akzeptiert und ebenfalls einen XML-Stream als Ausgabe wieder ausgibt.

3. Serializer sind verantwortlich für die Ausgabe und damit auch für das Beenden eines XML-Streams durch die Pipeline. Die Herausgebe des Inhalts erfolgt in einem passenden Format. Dieses Format ist meistens HTML, es kann aber auch ein nahezu beliebiges Grafikformat sein. Auch die Ablage einer Datei in einem Dateisystem oder praktisch jedes andere Format der Ausgabe ist ebenso realisierbar.

Für eine Grundinstallation von Cocoon werden folgende Softwarepakete benötigt:

1. Das Java Development Kit (JDK)

2. Ein J2EE Servlet Container, z.B. Apache Tomcat, Orion Web-Server, WebLogic, Jrun, Jboss, Resin, usw.

3. Die Cocoon Binaries. Eine genaue Anleitung zur Installation der jeweils aktuellen Version von Cocoon gibt es unter http://xml.apache.com/cocoon. Wir planen Cocoon wie in Abb. 4 gezeigt, in der multimedialen Lehre einzusetzen:

3.3 Experimentierumgebung zum Test von Software

In der Lehreinheit 3.1 (Prozessqualität) wird den Studierenden vermittelt, dass mit der Formulierung der Ziele eines Softwareprojektes und seiner Phasen schon angegeben werden muss, wann die Ziele erreicht sind und wie dies nachgewiesen werden soll. In der Lehreinheit 3.2 (Produktqualität) liegt einer der Schwerpunkte bei den Nachweisverfahren.

Zum praktischen Üben haben wir dazu auf der Basis des Test-Frameworks Junit von Erich Gamma und Kent Beck (http://www.junit.org) eine Experimentierumgebung aufgebaut (siehe Abb. 5). Das JUnit-Framework wurde für das Schreiben von Klassentests (Einzeltests) entwickelt. Das Framework verwendet das Entwurfsmuster Kompositum, das es ermöglicht, Testfälle zu Testsuits zusammenzufassen. Solche Suites können einzelne Testfälle, aber auch Folgen von Testfällen enthalten. Das macht es leicht, eine Reihe hierarchisch aufgebaute Testsuites zu entwickeln und die Tests automatisch auszuführen.

Klassentests sind hochgradig lokalisiert. Es wird angenommen, dass jede Testklasse nur in einem Paket arbeitet. Bestehen Anwendungen aus mehreren Klassen, so müssen zusätzlich zu den beteiligen Klassen die Schnittstellen zu anderen Paketen getestet werden. Dabei geht

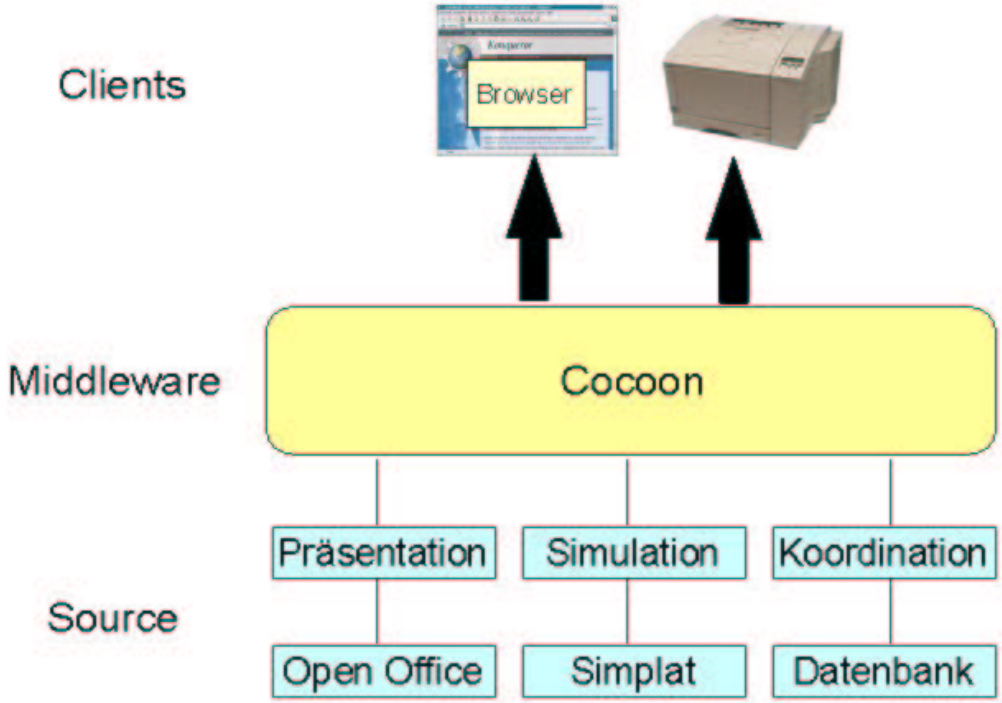


Abbildung 4: Verwendung von Cocoon in der multimedialen Lehre

man davon aus, dass die Klassen getestet sind. Funktionale Tests stellen sicher, dass das Zusammenspiel einzelner Klassen zur Erbringung einer bestimmten Funktionalität fehlerfrei ist. Sie werden in der Regel von unabhängigen Teams durchgeführt. Ein solches Team kann zu seiner Unterstützung andere Werkzeuge einsetzen. Schlägt einer der funktionalen Tests fehl, dann werden Klassentests geschrieben, die den Fehler erkennen und die Fehlerbehebung bei steigender Qualität beschleunigen.

Wie gesagt kann das Vorgehen rekursiv angewandt werden. Die Durchführung von Integrations- und Abnahmetests kann nach demselben Vorgehen erfolgen.

4 Evaluierung

Zur Evaluierung der Lernmodule wird ein mehrstufiges Verfahren eingesetzt. Nachdem die Lernmodule erstellt und vom Dozenten als fachlich richtig und dem Lernziel entsprechend bewertet werden, erfolgt im ersten Schritt eine Bewertung nach pädagogischen Gesichtspunkten. Dabei werden von einem unabhängigen Reviewer

- Formulierungen auf ihre Verständlichkeit überprüft

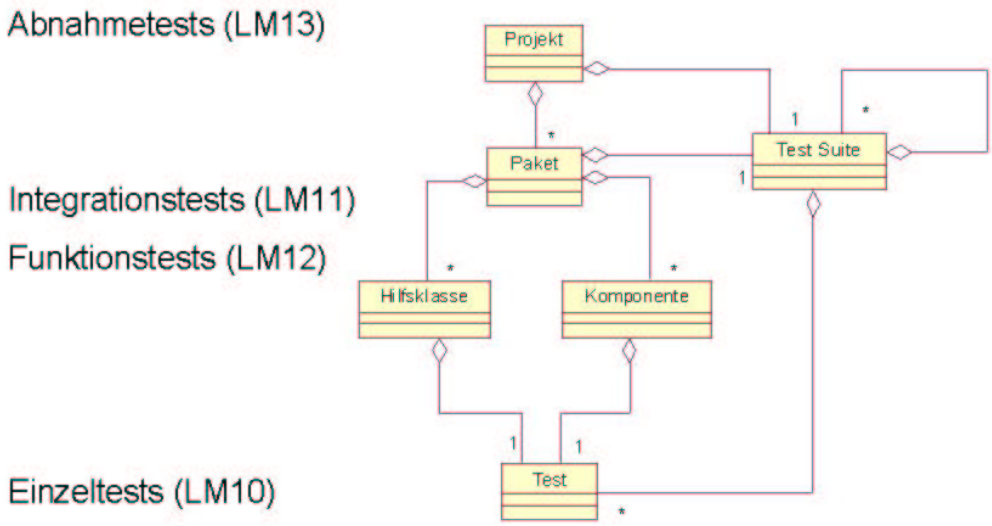


Abbildung 5: Struktur der MuSoft-Testumgebung

- Begriffe identifiziert, die gar nicht oder zu wenig erklärt werden und
- Beziehungen zu anderen Folien und Erklärungen aufgezeigt.

Anschließend werden dem aktuellen Lehrziel, mit dem der Lernmodul eingesetzt wird, entsprechende Fragen formuliert und zu jeder Frage eine html Seite generiert, über die die Studierenden ihr Verständnis an Hand unterschiedlicher Antwortalternativen überprüfen können. Beispiele dafür findet man auf der schon erwähnten Lehre homepage des IKE unter den dort veröffentlichten MuSofT Lernmodulen.

Sind die Lehrmaterialien so an die Ziele einer aktuellen Lehrveranstaltung angepasst, so werden sie dort eingesetzt und von den Studierenden bewertet. Im Maschienenbau ist die Zahl der Studierenden, die Vorlesungen aus dem Bereich der angewandten Informatik hören, zur Zeit eher gering. Wir haben daher auf statistische Erhebungen verzichtet und stützen die Bewertungen auf direkte Gespräche mit den Studierenden - vor allem während der Praktika - und auf eine Bewertung der in den Praktika erzielten Ergebnisse und des Einsatzes der Studierenden. Dabei haben sich vor allem zwei Dinge ergeben, die unsere Lehre entscheidend verändern.

Die erste Erfahrung ist die, dass die Studierenden zunächst mit einer recht allgemein gehaltenen und dafür leichter verständlichen Einführung in den Themenbereich zufrieden sind. Es ist ihnen dann aber hilfreich, wenn sie sich gezielt in Teilbereichen, die sie besonders interessieren oder die sie für Arbeiten in anderem Umfeld benötigen, vertiefend weiterbilden können. Diese Weiterbildung erfolgt ähnlich wie bei Studien- und Diplomarbeiten weitgehend auf ei-

gene Initiative. Hyperlinks zu weiterführenden Angeboten sind daher erwünscht und werden auch genutzt.

Die zweite Erfahrung ist die, dass die Studierenden die Übersichten aus den einführenden Vorlesungen durch eigenes Üben am Rechner vertiefen wollen. Es ist ihnen eher langweilig zu erfahren, wie man in der Theorie ein Vorgehensmodell auf ein abstraktes Problem zuschneidet, und sehr spannend , dies an Hand einer Aufgabe selbst durchzuführen. Praktika, die solches Üben ermöglichen, sind daher stärker als die Vorlesungen frequentiert. Dies nicht zuletzt auch deshalb, weil man sich ja den Vorlesungsstoff auf Grund der gut aufbereiteten Unterlagen im Netz selbst aufbereiten kann und es bei Bedarf möglich ist, sich weitere Informationen selbst zu beschaffen.

5 Planung für das nächste Jahr

Im Jahr 2003 werden wir unsere Arbeiten wie geplant weiterführen (siehe auch Tab. 1) und weitgehend abschließen können. Wie schon im Kap. 3 angedeutet werden wir von AIDA zu Cocoon als neuer Präsentationsplattform wechseln. Durch die dafür notwendige Trennung von Inhalt und Layout erhoffen wir uns eine größere Flexibilität beim Einsatz der Lehrmaterialien und ein einheitlicheres Vorgehen bei der Beschreibung der Lehrobjekte.

Die Lehrobjekte sollen anschließend als XML Dateien in der Musoft Lehrplattform abgelegt und zur Verwendung in anderen Kontexten freigegeben werden. Als erste Konsequenz aus den Erfahrungen bei der Evaluierung multimedialer Lernmodule haben wir die Vorlesung Simulation komplexer Anlagen und Systeme, in der wir den Studierenden Grundlagen des Softwareengineering im Hinblick auf Bau und Betrieb großer Systeme zur Simulation technischer Vorgänge vermitteln, recht radikal umgestellt. Die Vorlesung bestand in den vergangenen Jahren aus ca. 14 Doppelstunden pro Semester. Sie wurde ergänzt durch ein Praktikum, das an drei Nachmittagen stattfand.

Die neue Struktur dreht diese Verhältnisse nahezu um. Die Vorlesung wurde auf 10 Doppelstunden zurückgefahren. Sie wird jetzt ergänzt durch ein Praktikum aus 10 Vortrags und Übungs Sitzungen, für die sich die Studierenden eine ganze Woche in den Semesterferien freihalten. Der aktuelle Plan und seine Bezüge zu den MuSofT Lernmodulen der Lehreinheiten 3.1 und 3.2 stellt sich wie in Abb. 6 gezeigt dar.

Das Praktikum wurde inzwischen zur großen Zufriedenheit von Lernenden und Lehrenden durchgeführt. Sowohl der Einsatz der Studierenden, als auch die im Praktikum erzielten Ergebnisse, haben unsere Erwartungen übertroffen und zeigen damit, dass wir mit den MuSoft-Ansätzen auf dem richtigen Weg sind.

6 Zusammenfassung

Die Lehreinheiten 3.1 und 3.2 des Projektes MuSofT befassen sich mit Qualitätsmanagement(QM) bei der Entwicklung großer Softwaresysteme. Schwerpunkte sind prozessorientiertes (LE 3.1) und produktorientiertes (LE 3.2) QM. Als Produkte werden Softwaresysteme aus dem Ingenieurbereich mit stark simulationsorientierten Komponenten zugrundegelegt. Die Möglichkeiten multimedialen Arbeitens werden derart genutzt, dass die Lernmodule primär

Versuch	Aktion	Produkt	Bezug Lernmodul
1	Was wollen wir tun - das Pflichtenheft	Pflichtenheft	LM 7
2	Wie wollen wir das tun - Projekthandbuch	Projekthandbuch	LM 4
3	Mit was wollen wir das tun 1 - Modellierung	Modell in UML	LM 3
4	Mit was wollen wir das tun 2 - Entwicklungsumgebung	Eclipse Umgebung	-
5	Mit was wollen wir das tun 3 - Das Team, seine Produkte und deren Integration	CVS Umgebung	LM 7
6	Testumgebung + Testplanung	Test Umgebung	LM 10/12
7	Implementierung	Code	-
8	Integration	Programm	LM 11
9	Der Personal Softwareprozess	Review PSP	LM 6
10	Abschlussdiskussion - was würden wir das nächste mal besser machen	Testat	LM 3

Abbildung 6: Inhalt Praktikum 17. - 21.02.03

einführenden Charakter haben und durch Hyperlinks mit Systemen verbunden werden, die mittels Tailoring auf aktuelle Projekte angepasst werden können. Dadurch können die Lernmodule in den verschiedensten Kontexten wiederverwendet werden. Die Lernmodule werden ergänzt durch Beispielanwendungen, die mit den Systemen erzeugt wurden. Am Ende des Jahres 2002 haben wir eine erste Fassung aller Lernmodule realisiert. Eine erste Phase der Erprobung hat stattgefunden und zu wesentlichen Umorganisation des Lehrvorganges geführt. Die neuen Ideen werden im Jahre 2003 erprobt. Dazu nötige Werkzeuge werden erstellt oder von Partnern übernommen. Mit einer Endversion der Lernmodule rechnen wir im Herbst 2003.

Ergebnisbericht 2002 des MuSofT-Teilprojekts 3.3

Corina Kopka

Dieser Bericht gibt den aktuellen Stand der MuSofT-Lerneinheit 3.3 wieder, die als Thema die Lehre des Unified Software Process hat. Er enthält die Ergebnisse des letzten Jahres sowie die Planung für das letzte abschließende Jahr.

Inhaltsverzeichnis

1 Einleitung

Dieser Bericht gibt den aktuellen Stand der MuSofT-Lerneinheit 3.3 wieder, die als Thema die Lehre des Unified Process als Softwareentwicklungsprozess hat.

Die Lerneinheit 3.3 *Durchführung von Softwareprojekten mit dem Unified Process* wird im Rahmen des Projekts *MuSofT - Multimedia in der SoftwareTechnik* entwickelt und für die Lehre der Softwaretechnik eingesetzt. Sie soll Unterstützung bei der Durchführung von Softwarepraktika bieten. In Abgrenzung zu Programmierpraktika, bei der die Implementierung (z.B. von Algorithmen) im Vordergrund steht, handelt es sich hier um die Unterstützung umfassender Softwaretechnikpraktika, in denen Softwareentwicklungsprojekte durchgeführt werden. Die Studierenden sollen in Arbeitsgruppen neben dem Umgang mit einer Entwurfsnotation und mit den unterstützenden Werkzeugen, aufgrund einer konkreten Aufgabenstellung Software entwickeln. Der dabei zu beschreitende Entwicklungsprozess ist im Rahmen dieser Lerneinheit zu erlernen. Dazu wird der Ablauf eines Projektes auf der Grundlage des Unified Process visualisiert. Desweiteren kann ein Teil der Lerneinheit nicht nur zum Selbststudium, sondern auch durch den Lehrenden (z.B. in Vorlesungen zur Softwaretechnik oder in vorbereitende Vorlesungen innerhalb des Softwarepraktikums) eingesetzt werden.

In der Industrie hat sich die iterative Softwareentwicklung durchgesetzt, da damit Risiken frühzeitig erkannt werden können und Fehler in frühen Phasen mit weniger Aufwand als in klassischen nichtiterativen Entwicklungsprozessen behoben werden können. Für eine praxisnahe Ausbildung wurde der Unified Process als Vertreter iterativer Softwareentwicklungsprozesse gewählt.

2 Vorstellung der Lerneinheit

Während Softwaretechnikvorlesungen eher darauf ausgerichtet sind, softwaretechnisches Wissen im Sinne eines Informationsvermittlungsprozesses weiterzugeben, eröffnen Softwarepraktika u.a. die Möglichkeit, softwaretechnisches Wissen im Zusammenhang eines konkreten Projektes anzuwenden und Erfahrungen hinsichtlich der Kommunikation und Zusammenarbeit in einem Projektteam zu sammeln.

Schwerpunkt in Softwarepraktika bilden i.d.R. der Einsatz bereits bekannter Sprachen und Methoden für Analyse, Entwurf, Implementierung und Test. Darüberhinaus werden aber auch genaue Kenntnisse über Softwareentwicklungsprozesse benötigt, die oft nicht explizit gelehrt werden. Typischerweise werden Prozessmodelle vorgegeben [Sch01], so dass keine der möglichen Prozessmodellalternativen betrachtet werden. In der MuSofT-Lerneinheit *Durchführung von Softwareprojekten mit dem Unified Process* hingegen wird der Entwicklungsprozess anhand des Beispiels *Unified Process* [JBR98, Kru99] zum Lerngegenstand erhoben.

Der Unified Process ist idealerweise für große Projekte geeignet und erweist sich daher als problematisch für die Anwendung in kleineren Praktika, wie etwa typischen Programmierpraktika. Die MuSofT-Lerneinheit unterstützt daher umfassendere Softwaretechnikpraktika und Projektgruppen, in denen kleinere und mittelgroße Softwareentwicklungsprojekte durchgeführt werden.

Die MuSofT-Lerneinheit *Durchführung von Softwareprojekten mit dem Unified Process* legt einen Schwerpunkt darauf, Studierenden die Gelegenheit zu geben, Gelerntes über den Softwareentwicklungsprozess konstruktiv anzuwenden und Erfahrungen zu sammeln. Dazu wird von den Studierenden ein solcher Entwicklungsprozess geplant. Darüberhinaus wird in einem konkreten Projekt auf Basis des geplanten Entwicklungsprozesses Software entwickelt, so dass Studierende Gelegenheit bekommen, ihre Planung zu evaluieren, über ihre Entscheidungen in der Planung zu reflektieren und Konsequenzen aufgrund ihrer neuen Erfahrungen zu ziehen. Lernen erfolgt also durch reflektiertes Handeln und folgt damit konstruktivistischen didaktischen Ansätzen. Diese Art von Lernprozess erinnert nicht zufällig an den *Personal Software Process* [Hum96], bei dem die Entwickler empirische Daten über ihre eigenen Arbeiten sammeln, um sie später statistisch zu analysieren und damit nachfolgende Prozesse besser planen zu können.

Die MuSofT-Lerneinheit 3.3 unterstützt die Studierenden in der Projektplanung auf Basis des Unified Process und begleitet sie in der Projektsoftwareentwicklung. Im Vordergrund der Planung stehen nicht die Vorgabe eines verfügbaren Budgets oder eine Kosten- und Aufwandsschätzung, sondern die Identifizierung sinnvoller Aktivitäten des Entwicklungsprozesses, ihre sorgfältige zeitliche Einordnung innerhalb eines zeitlichen Rahmens, die Planung von sinnvollen Iterationen, also insbesondere auch die richtige Anwendung des Unified Process als generisches Modell für ihre spezifischen Projektzwecke. Das Lernen erfolgt im Unterschied

zu dem Ausbildungskonzept aus Siegen (Lerneinheit 3.4) nicht durch die Durchführung werkzeugunterstützter Simulationen, sondern durch Beobachtung und Analyse während der Durchführung eines Entwicklungsprozesses und durch Rückschlüsse bzgl. der ursprünglichen Planung. Somit wird das Lernen des Planens eines Softwareprojekts mit dem Lernen Software zu entwickeln integriert.

Als Konsequenz aus den bisher dargelegten Überlegungen ist unsere Lerneinheit konzipiert und wie folgt in Lernmodule strukturiert worden:

Lernmodul *Unified Process*

Dieses Lernmodul präsentiert den Projektablauf eines Musterprojekts. Der Lernende kann im Selbststudium selbstgesteuert den Ablauf eines Musterprojekts betrachten. Die Darstellung des Projektablaufs soll Tätigkeiten, Dokumente und Rollen berücksichtigen. Dieses Lernmodul kann wahlweise auch von einem Dozenten zum Lehren innerhalb einer Vorlesung eingesetzt werden. Für den Einsatz als Lehrmodul wird eine lineare feste Folge von Szenen (in Form einer *Guided Tour*) aufbereitet. Dieses Lernmodul unterscheidet jeweils eine Sicht für den Lernenden und den Lehrer.

Lernmodul *Projekte maßschneidern* (engl. Tailoring)

Dieses Lernmodul stellt ein Werkzeug zur Verfügung, mit dem die zentralen Elemente eines Entwicklungsprozesses (Aktivitäten, Dokumente, Rollen, Iterationen) auf die Erfordernisse des Anwendungsprojekts und auf die jeweilige Anwendungsdomäne anpaßbar sind. Mit dem Werkzeug soll ein konkreter Prozess für das Anwendungsprojekt modelliert werden. Die Anzahl der Iterationen in den Phasen des Unified Process wird eingeschränkt, da die Projekte eines Praktikums in der Ausbildung im Vergleich zur industriellen Entwicklung sehr kurzlebig sind. Beispiele für konkrete Projektabläufe aus Musterprojekten werden mit ihren Besonderheiten zur Hilfestellung vorgestellt.

Lernmodul *Adaptiver Projekttutor*

In diesem Lernmodul soll ein virtueller Projekttutor auf Basis eines konkreten Projektablaufs den Arbeitsfortschritt einer studentischen Arbeitsgruppe anhand der erstellten Dokumente in einem Projekt erkennen. In Abhängigkeit des erzielten Arbeitsfortschritts wird Hilfestellung für den weiteren Projektverlauf gegeben. Dies geschieht durch eine Übersicht über die verbleibenden Arbeitsschritte und durch das Präsentieren von Lernmodulen mit genauen Beschreibungen, Anleitungen und Beispielen zu diesen Schritten. Der konkrete Projektablauf ist ein von der Arbeitsgruppe geplanter Entwicklungsprozess, kann aber auch ein vorgegebener Standardprozess sein.

3 Technische Realisierung

3.1 Lernmodul *Unified Process*

Die Entwicklung des Lernmoduls *Unified Process* vollzieht sich in den groben Phasen Vorplanung, Konzeptionierung, Realisierung und Test. In die Vorplanung gehören inhaltliche und didaktische Vorüberlegungen, die Festlegung eines Vorgehensmodells und die Projektplanung. Die Konzeptionierungsphase beinhaltet im wesentlichen die Erstellung eines Grobkonzepts, eines Feinkonzepts und eines Drehbuchs, in denen die spätere Anwendung unter den Aspekten Didaktik, Inhalt, Medieneinsatz/Mediendesign und Navigationsstruktur spezifiziert wird. Eine Beispielseite aus dem Drehbuch wird in Abb. 1 dargestellt. Das Drehbuch bildet die Vorlage für die Realisierungsphase, in der die softwaretechnische Entwicklung der Lerneinheit und die Produktion von Medienobjekten stattfinden. Beim Testen wird zwischen technischem Test und Evaluation als inhaltlich-didaktischer Test der Lerneinheit unterschieden.

Die Entwicklung erfolgt iterativ. Zuerst wird eine Probesequenz (Szenenfolge) beispielhaft entwickelt und getestet, um möglichst früh eine Vorstellung vom Verhalten und Aussehen des Endprodukts zu bekommen und gegebenenfalls Änderungen vorzunehmen. Die Phasen der Drehbucherstellung, Lernprogrammentwicklung, Medienproduktion und der Tests (technischer Test, Evaluation) werden mehrfach iterativ durchlaufen. Eine Iteration wird für eine Sequenz von Szenen durchgeführt.

Obwohl die Planung des Medieneinsatzes vorangeschritten ist, wurde und wird mit einer Mediendesignerin zusammengearbeitet, um weitere Ideen für passende und aussagekräftige Medien für verschiedene Inhalte zu finden. Desweiteren soll durch die Teilnahme an dem MuSofT-Gestaltungsworkshop im Januar 2003, in dem in allgemeine Gestaltungsrichtlinien eingeführt wird, die als Grundlage für die Gestaltung von Lerneinheiten und GUIs dienen können, Anregungen für Verbesserungen des Bildschirmdesigns des Lernmoduls gefunden werden.

Die Produktion von Medien, wie bspw. Grafiken und einfache Animationen erfolgt in Eigenregie durch das Entwicklungsteam des Teilprojekts am Lehrstuhl für Software-Technologie der Universität Dortmund. Es sind aber auch Medien geplant, die schwierig, aufwändig oder nur von Medienexperten zu erstellen sind. Dies trifft auf einige Animationen mit Sprechertexten zu und auf ein Video, das die Aufgabenstellung für ein Softwareprojekt als Dialog zwischen Kunde und Softwareentwickler gestaltet. Für die Erstellung dieser Medien wird zur Zeit mit dem Medienzentrum an der Universität Dortmund zusammengearbeitet, die auf dem Gebiet der Medienproduktion kompetent sind.

Als Beispielprojekt für die Veranschaulichung des Unified Process wurde eine Aufgabenstellung aus dem Logistikbereich gewählt. Konkret wird der Ablauf eines Projekts dargestellt, in dessen Rahmen Software für die Kommissionierung im Pharmagroßhandel entwickelt wird.

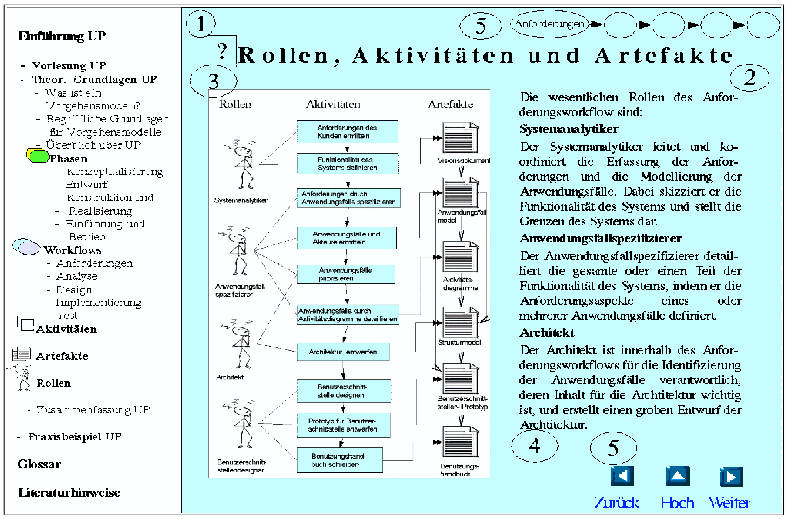
3.2 Lernmodule *Projekte maßschneidern* und *Projekttutor*

Da diese beiden Lernmodule funktionale Werkzeuge sind und nicht wie das Lernmodul *Unified Process* Wissen multimedial präsentieren, erfolgt ihre Entwicklung in Anlehnung an das Vorgehen in der traditionellen Softwareentwicklung. Insbesondere sind hier keine Drehbü-

20.Szene: Anforderungsworkflow - Rollen, Aktivitäten und Artefakte

Beschreibung:

In dieser Szene werden die am Anforderungsworkflow beteiligten Rollen mit den von ihnen auszuführenden Aktivitäten und den zu erstellenden Artefakten dargestellt.



Nr.	Typ	Beschreibung	Name
	Hintergrund	*Farbe*: Blassgrün	
1	Grafik	Wiedererkennungsmerkmal für die Rubrik „Rollen, Aktivitäten und Artefakte“ *Symbol :* ...	
2	Text	Überschrift der Szene. *Inhalt:* Rollen, Aktivitäten und Artefakte.	
3	Grafik	Die Grafik illustriert den Zusammenhang zwischen den Rollen, Aktivitäten und Artefakten.	RollenAktivitätenArtefakte.gif
4	Text	Der Text führt alle am Workflow beteiligten Rollen auf und erläutert die von ihnen auszuführenden Aktivitäten sowie die zu erstellenden Artefakte. *Inhalt:* siehe 4.2.5 Komponente: Rollen, Aktivitäten und Artefakte	4.2.5 Komponente: Überblick

Abbildung 1: Eine Beispielseite aus dem Drehbuch

cher als Vorlage für die Implementierung notwendig, der Entwurf der Werkzeuge geschieht aufgrund bekannter Methoden der Softwaretechnik.

Kern des Lernmoduls *Projekte maßschneidern* ist ein Editor zur Modellierung des eigenen Entwicklungsprozesses auf Basis des Unified Process. Mit dem Werkzeug können Model-

lierungselemente wie Phasen, Iterationen, Workflows, Aktivitäten, Artefakte und Abhängigkeiten angelegt werden. Vorhandene Prozessmodellierungswerkzeuge kamen für den Einsatz in diesem Lernmodul nicht in Frage, da sie eine gute Unterstützung in der Handhabung des Unified Process nicht gewährleisten können. Ziel ist es, ein Werkzeug anzubieten, das nicht nur die passenden Modellierungselemente anbietet, sondern auch die Eigenheiten des Unified Process berücksichtigt. Beispielsweise besteht ein Entwicklungsprozess auf Basis des Unified Process immer aus den vier Phasen *Konzeptualisierung (inception), Entwurf (elaboration), Konstruktion und Realisierung (construction), Einführung und Betrieb (transition)*. Daher ist zur Unterstützung des Lernenden beim Anlegen eines neuen Projekts das Anbieten dieser Phasen in der richtigen Reihenfolge sinnvoll, ohne dass er diese explizit anlegen muss. Ebenso sind weitere Möglichkeiten gegeben, um den Lernenden mehr Unterstützung zu geben, den Unified Process richtig anzuwenden.

Für die Realisierung des Editors wurden verschiedene Frameworks, Werkzeuge und Klassenbibliotheken auf ihre Eignung untersucht, im Sinne der Anforderungen an den Editor eingesetzt zu werden. Hierzu fand ein MuSofT-interner Workshop mit mehreren Teilnehmern aus den Teilprojekten im Januar 2002 statt, auf dem die Evaluation verschiedener Werkzeuge auf die Teilprojekte verteilt wurden. In die engere Wahl für das Teilprojekt 3.3 kamen zuerst JHotDraw und später Jazz. Die Entscheidung fiel auf JHotDraw, ein Framework in Java für grafische Editoren. Zur Zeit befindet sich die Entwicklung des Editors in der Implementierungsphase.

Die Entwicklung des Lernmoduls *Projekttutor* ist angelaufen. Zur Zeit ist die Entwicklung in der Analysephase.

4 Evaluierung

Für den Einsatz und die Evaluation der Lerneinheit sind verschiedene Veranstaltungen der Universität Dortmund geeignet.

Um einerseits die inhaltliche und didaktische Zielsetzung eines Softwarepraktikums und die Möglichkeiten des Medieneinsatzes in einer unterstützenden multimedialen Lerneinheit zu erforschen und um andererseits die Lerneinheit 3.3 nach ihrer Fertigstellung zu evaluieren, ist im Rahmen dieses Vorhabens eine Zusammenarbeit mit den Veranstaltern des Softwarepraktikums des Fachbereichs Informatik der Universität Dortmund angestrebt, die zur Zeit ein grobes Prozessmodell [Sch01] einsetzen, das an das ISP-Modell (*Irrational Separated Model*) von Hitz und Kappel [HK99] angelehnt ist. Der Prozess selbst ist aber nicht das Lernobjekt. Durch den Einsatz der Lerneinheit 3.3 des MuSofT-Projekts soll in Softwarepraktika auch der Entwicklungsprozess zum Lerngegenstand werden und beispielhaft anhand des Unified Process gelehrt werden können.

Für die Evaluation wird im Softwarepraktikum ein Projekt nach dem bisherigen groben Prozessmodell durchgeführt. Ein zweites Projekt wird mit Unterstützung der Lerneinheit 3.3 nach dem Unified Process durchgeführt. Es wird ein Fragebogen entwickelt, der nach Durchführung der beiden Softwareprojekte an die Teilnehmer des Praktikums verteilt wird.

Neben der Evaluation im Softwarepraktikum ist das Lernmodul *Unified Process* auch geeignet, in der Vorlesung “Softwaretechnik“ eingesetzt zu werden. Beim Einsatz als Lehrmodul steht die Evaluation durch den vortragenden Dozenten im Vordergrund. Beim Einsatz aller

Lernmodule im Softwarepraktikum spielt neben der Evaluation durch den Lehrenden auch die Evaluation durch Lernende eine große Rolle.

Angesichts der Komplexität des Unified Process ist bei der Gestaltung der Lernmodule auf geeignete Hilfestellung zu achten. Hier haben sich zwei mögliche Strategien herauskristallisiert, die wir in einem Softwarepraktikum evaluieren wollen:

- Unterstützung des Projektteams mit Hinweisen, wie aus einem generischen Modell des Unified Process ein Minimalprozess gewonnen werden kann;
- Vorgabe eines Minimalprozesses mit Hinweisen auf Erweiterungsmöglichkeiten für spezifische Anwendungsprojekte.

Es ist geplant, die vorgestellten Lernmodule in Kombination mit den beiden vorgestellten Strategien in der vorlesungsfreien Zeit im Sommersemester 2003 an der Universität Dortmund im Softwarepraktikum einzusetzen.

5 Planung für das nächste Jahr

Die Entwicklung des Lernmoduls *Unified Process* befindet sich zur Zeit in der Implementierungsphase. Die Entwicklung des Lernmoduls soll bis auf die externe Medienproduktion im April 2003 abgeschlossen werden. Hierzu sollen noch Anregungen des Gestaltungsworkshops im Januar mitgenommen werden und mit einer externen Mediendesigner zusammengearbeitet werden, um weitere Ideen für passende und aussagekräftige Medien für verschiedene Inhalte zu finden. Die externe Produktion einiger Medien im Medienzentrum der Universität Dortmund wird noch Zeit in Anspruch nehmen, so dass das Ende der Realisierung im Juni 2003 angestrebt wird.

Die Entwicklung des Lernmoduls *Projekte maßschneidern* befindet sich zur Zeit ebenfalls in der Implementierungsphase. Es ist angestrebt, die Entwicklung des Editors zur Planung von Softwareentwicklungsprozessen nach dem Unified Process im Juni 2003 abzuschließen.

Die Evaluation der Lernmodule findet in der vorlesungsfreien Zeit im Sommersemester 2003 an der Universität Dortmund statt.

6 Zusammenfassung

Die Lerneinheit 3.3 ist ein Lehr-und Lernsystem, das über die Gestaltung als Unified-Process-Informationssystem hinaus geht. Die Einbettung der Lerneinheit in ein didaktisches Konzept führt zu Anforderungen wie die didaktischen Aufbereitung von Lehrmaterial, der Einsatz mediendidaktischer Elemente und die Betreuung der Lernenden im Lernprozess:

- Dozenten sollen anhand von Lehrmodulen den Unified Process lehren können. Die Einbeziehung eines didaktischen Konzepts in der Entwicklung der Lerneinheit 3.3 soll die Präsentation reinen Faktenwissens in einem Informationssystem ergänzen.
- Lerner sollen unter Betreuung in einem konkreten Projekt den Entwicklungsprozess auf Basis des Unified Process planen lernen. Lernen bedeutet an dieser Stelle Anwenden des

Faktenwissens über den Unified Process, insbesondere auch das Planen von Iterationen des Iterations-Workflows in einem konkreten Projekt.

- Lerner sollen den Unified Process explizit erfahren. Lernen bedeutet an dieser Stelle das Befolgen eines vorgegebenen oder des selbst geplanten Entwicklungsprozesses innerhalb des Softwarepraktikums. Hiermit werden z.B. geplante Prozesse evaluiert. Diese Evaluation ist ebenfalls ein Lernprozess mit dem Ziel, Projekte besser planen zu können.

Dies unterscheidet die Lerneinheit 3.3 als multimediales Lehr- und Lernsystem von einem reinen Informationssystem, wie z.B. das Online-System für den Rational Unified Process [RUP]. Gewiss sind die visuelle grafische Darstellung eines Projektablaufs im Softwarepraktikum und ein grafisches Navigieren durch diese Darstellung geeignete Mittel, um ein multimediales Nachschlagewerk zu konzipieren. Dies ist auch für die Lerneinheit 3.3 vorgesehen. In Abgrenzung zum RUP-Hilfesystem sollen die Zusammenhänge zwischen Rollen, Aktivitäten und Dokumenten zur besseren Übersicht auch grafisch besser verdeutlicht werden, wie etwa die Schnittstellen zwischen Rollen oder die Eintritts- und Austrittskriterien für Aktivitäten (in Form von Ein-, Ausgabedokumenten).

Literatur

[HK99] HITZ, MARTIN und GERTI KAPPEL: *UML@Work - Von der Analyse zur Realisierung*. dpunkt-Verlag, 1999.

[Hum96] HUMPHREY, WATTS S.: *Using a Defined and Measured Personal Software Process*. IEEE Software, Seiten 77–88, Mai 1996.

[JBR98] JACOBSON, IVAR, GRADY BOOCH und JAMES RUMBAUGH: *The Unified Software Development Process*. Addison Wesley, 1998.

[Kru99] KRUCHTEN, PHILIPPE: *The Rational Unified Process: an Introduction*. Addison-Wesley, 1999.

[RUP] *Rational Unified Process*. www.rational.com/rup_info/.

[Sch01] SCHMEDDING, DORIS: *Ein Prozessmodell für das Software-Praktikum*. In: LICHTER, HORST und MARTIN GLINZ (Herausgeber): *SEUH 7. Software Engineering im Unterricht der Schulen*, Seiten 87–97, Zürich, Februar 2001. dpunkt-Verlag.

Bericht des MuSofT-Teilprojekts 3.4 (Projektmanagement) über das zweite Projektjahr

Udo Kelter

Inhaltsverzeichnis

Dieses Papier beschreibt die Materialien, die im zweiten Projektjahr (weiter-) entwickelt worden sind. Wir konzentrieren uns hier auf einen Lehrfilm, das in die Bedienung eines CVS-Frontends einführt, die beim Einsatz des Films gewonnenen Erfahrungen und die aufgetretenen technischen Probleme. Ferner wird die Installation eines Streamingservers für den Film beschrieben.

1 Übersicht

Die in Teilprojekt 3.4 zu erstellenden Materialien sollen zumindest einführend, punktuell auch vertiefend folgende Themengebiete abdecken:

1. Versions- und Konfigurationsmanagement (VKM)
2. Projektplanung
3. Fehler- und Problemmanagement

Eine detaillierte Begründung der Auswahl und Einbettung in den Kontext des Gesamtprojekts findet sich im Bericht zum ersten Projektjahr.

Die Arbeiten im Bereich VKM konzentrierten sich im zweiten Projektjahr auf die Erstellung eines Lehrfilms, der in die Bedienung eines CVS-Frontends einführt; diese Arbeiten sind in Abschnitt 2 beschrieben.

Die Arbeiten im Bereich Projektplanung konzentrierten sich auf die Fertigstellung eines Editor-Simulators für Netzpläne. Einige Erfahrungen mit diesem Simulator sind schon in [1] beschrieben worden und werden daher in diesem Bericht nicht noch einmal wiedergegeben.

Im Bereich Fehler- und Problemmanagement wurde an einer Marktstudie gearbeitet; diese Arbeiten haben noch nicht zu extern sichtbaren Produkten geführt, weswegen wir hier auf diesen Bereich ebenfalls nicht näher eingehen.

2 Der CVS-Lehrfilm

2.1 Beschreibung

Bei dem CVS-Lehrfilm handelt es sich um ein ca. 30 Minuten langes Video, das die grundlegenden Arbeitsabläufe bei der Benutzung von CVS anhand von Beispielen zeigt. Gezeigt wird primär ein Bildschirm mit den Mausbewegungen darauf sowie den jeweiligen Ausgaben des Systems; dies wird durch gesprochenen Text zusätzlich erläutert.

Der Film war ursprünglich dazu gedacht, einzelnen Gruppen eines Programmierpraktikums (ca. 5 - 10 Personen) in einem kleineren Raum per Beamer gezeigt zu werden. Der Film ist daher in 8 Abschnitte von ca. 1.5 - 6 Minuten Länge gegliedert, zwischen denen jeweils eine Diskussionspause eingelegt werden kann bzw. sogar sollte. Die Themen der Abschnitte sind:

Szene 0: Beschreibung des Szenarios
Szene 1: Ein bereits bestehender Workspace
Szene 2: Das erste Einloggen
Szene 3: Checkout vom zentralen Repository
Szene 4: Bearbeitung einer Datei
Szene 5: Dateien neu anlegen
Szene 6: Aktualisierung der lokalen Daten
Szene 7: Paralleles Arbeiten in mehreren Workspaces

Als Vorbereitung zu dem Film ist ein Lehrtext vorgegeben, in dem die grundlegenden Begriffe des Konfigurationsmanagements erklärt werden und der im Prinzip vorher gelesen werden sollte.

2.2 Versionen des Films und Einsatzerfahrungen

Eine erste, noch unvertonte Version des Films wurde im Wintersemester 2001/02 testweise im Programmierpraktikum bei 8 Gruppen eingesetzt. Ca. 1 Woche später wurden die Teilnehmer bzgl. des Nutzens des Films befragt. Es wurden diverse Möglichkeiten erkannt, den Film zu verbessern.

Eine verbesserte und vertonte Version wurde zum Sommersemester 2002 fertiggestellt und erneut im Programmierpraktikum in 10 Einzelgruppen mit jeweils ca. 6 Studenten eingesetzt und über einen Fragebogen evaluiert. Wesentliche Beobachtungen bei diesem, aber auch anderen Einsätzen waren:

- Der Lehrfilm wurde weit überwiegend als nützlich und hilfreich angesehen. In der Tat verlief die Benutzung des zentralen CVS-Repositories in dem Praktikum problemlos, obwohl es sich um eine für die Studenten völlig neue Technologie handelt.

- Die Diskussionspausen wurden bei manchen Gruppen intensiv genutzt.

Neben dem eigentlichen Lehrfilm wurden noch ergänzende Materialien erstellt, u.a. Handreichungen für Betreuer von Lehrveranstaltungen, in denen ein CVS-Server installiert werden muß.

Die im Rahmen des Projekts erstellten Materialien sowie die zugrundeliegenden public-domain-Produkte wurden zu einer CD zusammengestellt, die an ausgewählte Projektpartner des MuSofT-Projekts weitergegeben wurde. Das Material wurde ferner an zwei externe Kooperationspartner weitergegeben:

- Prof. Convent (FH Bocholt); hier wurde der Film in einem sehr ähnlichen Kontext wie in Siegen eingesetzt, nämlich in einem Programmierpraktikum.

  Prof. Convent lieferte eine umfangreiche Liste mit detaillierten Kommentaren und sehr nützlichen Verbesserungsvorschlägen. Viele dieser Kommentare betrafen stillschweigende Annahmen z.B. zu Termen und Begriffen, die an anderen Standorten das Verständnis des Films unnötig erschwerten.

- Prof. Hasselbring (U. Oldenburg und U. Innsbruck); hier wurde der Film als Begleitmaterial innerhalb einer Vorlesung eingesetzt, also in einem deutlich anderen Kontext als in Siegen.

  Die von dieser Seite geäußerten Kommentare zielten u.a. darauf, die von dem Film abgedeckten Themen zu erweitern.

Aufgrund der internen und externen Evaluierung sowie eigener Überlegungen wurde im Sommer 2002 eine neue Version des Films geplant und realisiert, die die folgenden wesentlichen Unterschiede zur vorherigen aufweist:

- zusätzlicher Vor- und Nachspann
- Behebung diverser kleinerer Mängel
- Einfügung von kurzen Flash-Animationen an den Stellen, wo gerade ein CVS-Kommando ausgeführt worden ist. Die Animationen zeigen graphisch den Effekt des Kommandos.

Die neue Version wurde im Wintersemester 2002/03 bei insgesamt 17 Gruppen im Programmierpraktikum vorgeführt und per Fragebogen evaluiert. Die Ergebnisse waren ähnlich positiv wie bei vorherigen Befragungen.

Überraschenderweise war der Anteil der Befragten, die das Tempo des Films als etwas zu langsam empfanden, signifikant höher als bei früheren Befragungen. Dies geht sehr wahrscheinlich auf die zusätzlichen Animationen zurück, die für einen Teil der Zuschauer nicht nötig gewesen sind und zu einer mentalen Unterforderung geführt haben.

Diese Version des Films wurde ferner auf einem Streamingserver online abrufbar gemacht. Abschnitt 3 beschreibt den Aufbau des Servers und hierbei gemachte Erfahrungen.

2.3 Geplante zukünftige Arbeiten

Die vorhandene Version des Film soll noch einmal erweitert werden, und zwar um Szenen zu den Themen

- Tags und Releases
- Varianten und ggf. Mischen

Ferner sollen noch einige redaktionelle Verbesserungen durchgeführt werden.

2.4 Einige Erfahrungen

Der Film wurde gegenüber seiner Ursprungsversion mehrfach verändert und erweitert. Hierbei wurden einige Erfahrungen gemacht, die für zukünftige Erweiterungen oder ähnliche Vorhaben anderer Projektpartner nützlich sein können.

2.4.1 Synchronisation von Bild und Ton

Die Synchronisation von Bild und Ton ist auch bei dieser Art von Filmmaterial sehr kritisch. Dies mag zunächst überraschend klingen, denn, da “nur” Bildschirme gefilmt werden, braucht man keinen lippensynchronen Ton. Es zeigt sich allerdings, daß die Bewegungen der Maus und eventueller Texteingaben auf dem gezeigten Bildschirm sehr genau zum gesprochenen Text passen müssen und daß bereits ein geringfügiges zeitliches Auseinanderdriften als sehr störend empfunden wird.

In diesem Zusammenhang hat sich das Vorgehen, das bei der Realisierung der ersten Version des Films gewählt wurde, als ungünstig erwiesen. Das Vorgehen war in folgende Schritte gegliedert:

1. inhaltliche Planung, Entwicklung der Beispiele, Schreiben eines Drehbuchs incl. Szenenfolge und gesprochenem Text
2. Bildaufnahme (mit einer speziellen Hardware, die die am Videoausgang des PCs gelieferten Bilder aufzeichnet), wobei die Person, die den Rechner bedient, zugleich den Text spricht, um das richtige Bewegungstempo zu haben.
3. Tonaufnahme, wobei das vorher aufgenommene Video abgespielt wird und zugleich der Text von Blatt gelesen wird.
4. Synchronisation von Bild und Ton
5. Kodierung bzw. Komprimierung des Materials

Die Synchronisation von Bild und Ton (Schritt 4) hat sich als äußerst zeitraubend erwiesen. Wenn ein gesprochener Satz und der Bewegungsablauf auf den Bildschirm nicht ausreichend synchron verlaufen, läßt sich dies nachträglich nur schwer beheben. Der Ton läßt sich praktisch nicht stauchen oder strecken, bei den Bildaufnahmen kommen allenfalls die Pausen für Streckungen infrage. Daher muß sehr viel ausprobiert werden. Der Aufwand für die Synchronisation lag letztlich bei 1 - 2 Stunden Arbeitszeit pro Minute Film. Da die mit diesem Arbeiten betraute Arbeitskraft nicht ganztags und zu beliebigen Zeiten zur Verfügung stand, kam es außerdem zu längeren Wartezeiten, die relativ störend und unökonomisch sind.

Bei zukünftigen Aufnahmen werden die Schritte 2 und 3 vertauscht werden; dies reduziert vermutlich den Aufwand, weil man das Bild leichter an den Ton anpassen kann als umgekehrt: bei der Bedienung von Maus und Tastatur kann man zugleich den schon vorhandenen Ton hören und die Bewegungen an das gesprochene Wort anpassen.

2.4.2 Länge von Pausen und Sprech- bzw. Anzeigetempo

Als ein ziemlich lästiges Problem stellte sich die richtige Wahl der Länge von Pausen (sowie von Vor- und Nachspannteilen) und allgemeiner die Wahl des Sprech- und Anzeigetempos heraus.

Zu kurze Sprechpausen zwischen den Sätzen (bzw. Szenen) waren eine der häufigsten Ursachen für nachträgliche Änderungen an den Filmen[1].

Derartige Schwachpunkte stellen sich leider i.d.R. erst dann heraus, wenn ein Film komplett synchronisiert ist. Es ist bei Kontrollen nicht ganz einfach, derartige Fehler zu notieren – wenn man den Film anhält, um Notizen zu machen, hat man den Probelauf nachhaltig gestört! – und eine genaue Korrekturvorschrift anzugeben (die Angabe, daß die Pause nach dem Satz, der 3:15 Minuten nach Beginn der Szene endet, zu kurz war, ist keine direkt umsetzbare Änderungsanweisung).

In der Entwicklergruppe fehlte die Kenntnis einer “Theorie”, nach der z.B. die Länge von Sprechpausen schon im Vorfeld definiert und die systematische Einhaltung entsprechender Regeln kontrolliert werden kann. Eine solche Theorie ist umso wichtiger, wenn, wie hier, immer wieder im Abstand von mehreren Wochen kleinere Änderungen an dem Video vorgenommen werden und das Timing je nach der Tagesform der Beteiligten variiert.

[1] Die Pausen werden übrigens bei der Synchronisation festgelegt, da im Prinzip jeder einzelne aufgenommene Satz separat angeordnet wird.

2.4.3 Anpassung der Informationsdichte

Welches Tempo bzw. welche Informationsdichte als angenehm empfunden wird, ist in einem gewissen Rahmen nur subjektiv zu beurteilen. Im Gegensatz zu einem Sprecher, der unterbrochen werden kann oder der bemerkt, daß das Publikum unaufmerksam wird, weil der Inhalt zu langsam vorankommt, und dann das Tempo erhöht, läuft ein Film immer im gleichen Tempo weiter.

Hinzu kommt, daß bei einer längeren Laufzeit wie der hier vorliegenden (rund 28 Minuten, wenn man alle Szenen ohne Unterbrechung abspielt) durchaus damit zu rechnen ist, daß bei den Zuhörern zwischendurch die Konzentration nachläßt. Ein Vortragender könnte z.B. durch Tempoverlangsamung oder Wiederholung und / oder Zusammenfassung von Stoff ad hoc eine "Durchhängephase" einbauen.

Sofern man die Szenen einzeln ansieht bzw. nach jeder Szene in der Gruppe kurz über die vorangegangene Szene spricht, tritt das Problem praktisch nicht auf. Man kann hieraus schlußfolgern, daß das Timing nur für jeweils eines der Nutzungsszenarien (Ansehen einzelner Szenen vs. ununterbrochenes Ansehen aller Szenen) optimal gestaltet werden kann.

2.4.4 Komprimierung

Einen erheblicher Aufwand verursachte die Wahl eines geeigneten Komprimierungsverfahrens (Codec). Die gängigen Verfahren sind auf Bildmaterial abgestimmt, das typisch für Fernsehfilme ist. Aufnahmen von Bildschirmen unterscheiden sich hiervon ganz erheblich. So dürfen durch die Komprimierung z.B. keine Kanten verwischt werden, und die hohen Kontraste zwischen Buchstaben (genauer gesagt Linien, die zufällig Buchstaben darstellen) und dem Hintergrund sollten erhalten bleiben.

DIVX. Die ersten Versionen des Films wurden im AVI-Format mit dem Codec DIVX 4.12 erzeugt. Dieser Codec bietet im Vergleich zu den Standard-Codecs (wie Intel Indeo, Radius Cinepak) eine höhere Kompressionsrate. Die hohe Kompression wird unter anderem dadurch erreicht, daß das Video insgesamt unschärfer wird und Details in der Abbildung reduziert werden. Dies wird beim Betrachten von dunklen Linien auf hellem Hintergrund deutlich. So wird z.B. der Buchstabe "c", der in einer Beschriftung eines Buttons vorkommt, zu einer grauen Fläche mit dunklem Rand, ähnliche Effekte treten auch bei "m" und "nn" auf. Dieser Effekt läßt sich nur bedingt reduzieren, indem eine höhere Datenrate in Verbindung mit der 2-Paß-Codiermethode gewählt wird. Ein akzeptabler Kompromiß zwischen Dateigröße, Bildqualität und Framerate ließ sich bei einer Datenrate von 1.5 Mb/s und 15 Frames pro Sekunde erzielen.

Die Version vom November 2002 enthielt wie schon erwähnt zusätzliche Animationen, die mit Macromedia Flash programmiert worden waren. Die neuen Videosequenzen wurden dann wieder mit den oben beschriebenen Erfahrungswerten codiert unter Verwendung der neuen Version 5.02 von DIVX. Hier zeigte sich, daß die Probleme, senkrechte, schwarze Linien auf weißem oder hellgrauen Hintergrund scharf darzustellen, in den Animationen verstärkt auftraten und die Flash Animation sehr störend beeinflußten. So entarteten die Linien, die die Ordner der Verzeichnisstruktur verbinden, zu grauen Balken. Die Videos wurden versuchsweise mit verschiedenen Einstellungen und Versionen von DIVX codiert. Mit DIVX 5.02 konnten keine brauchbaren Ergebnisse erzielt werden, nur mit der älteren Version 4.12 gelang dies.

Realmedia bzw. SureStream. Unterdessen war parallel eine Aktivität angelaufen, den Film auch über einen Streamingserver (s. Abschnitt 3) anzubieten. Vorüberlegungen hinsichtlich der Wahl der Streamingtechnologie ergaben, daß die Realmedia-Plattform für die Situation in der Fachgruppe PI am besten geeignet ist.

Das Realmedia-Format hat die Besonderheit, mit einer variablen, dynamisch angepaßten Framerate zu arbeiten. Wenn viel Bewegung im Bild vorhanden ist, wird die Framerate bis auf einen vorgegebenen Maximalwert erhöht, um die Bewegung möglichst fließend darzustellen; ändert sich wenig im Bild, so wird die Framerate bis auf 1 fps zurückgesetzt. Dieses Verfahren erwies sich für den CVS-Film als sehr günstig, denn die Bewegungen in dem Film sind nur relativ langsame Dateneingaben in Eingabefeldern, Mausbewegungen und einige Überblendeffekte. Trotz hoher Auflösung und sehr guter Bildqualität, bei der auch scharfe Kanten mit starkem Kontrast ohne Unschärfe und Grauschleier darstellt werden, sind die entstandenen Dateien kleiner als bei DIVX oder vergleichbar groß.

Ein weiterer großer Vorteil des Verfahrens liegt darin, daß man sich bei der Komprimierung praktisch nicht um die Framerate kümmern muß, während bei anderen Verfahren zeitaufwendige Experimente mit verschiedenen Frameraten erforderlich waren.

2.5 Copyright-Probleme

Die in dem CVS-Film benutzte Software (WinCVS) unterliegt der GNU General Public License (GPL) [2], so daß es zunächst problemlos erschien, das System für den Film zu nutzen. Dies erwies sich überraschenderweise als unzutreffend.

Das Problem liegt darin, daß die GPL primär das Ausführen, Kopieren, Verteilen und Verändern einer Software behandelt und nur diese Nutzungen allgemein erlaubt. Wesentlich unklarer wird entschieden, was mit den Ausgaben eines Programms gemacht werden darf. Die Rechte an den Ausgaben gehören nur dann dem Produzenten, wenn die Ausgaben ein “eigenes”, also nicht allein durch das Programm erzeugtes Werk darstellen. Auch durch Rückfragen bei Rechtsexperten konnte nicht geklärt werden, wo die exakten Grenzen dieses Begriffs liegen.

2.6 Einsatzbewertung des Films

Angesichts der sehr hohen Kosten des Films stellt sich natürlich die Frage nach dem Nutzen, speziell also dem Mehrwert gegenüber einer ebenfalls möglichen live-Vorführung des CVS-Systems. Dies soll i.f. aus der Sicht von Lehrenden und Lernenden bewertet werden. Aus der Sicht von Lehrenden ergeben sich teilweise abhängig vom Nutzungsszenario folgende Vorteile:

- Zunächst erspart der Film natürlich den nicht unerheblichen Vorbereitungsaufwand, sich Beispiele und Abläufe auszudenken – insofern unterscheidet er sich nicht von einem Lehrbuch oder Folien.
- Für eine live-Vorführung muß ein passendes Repository installiert werden und ggf. entfernt zugreifbar sein. Auch dies verursacht Aufwand und kann zu Pannen und Verzögerungen führen, die besonders beim Einsatz in Vorlesungen sehr stören.

- Mit Hilfe des Films kann die Arbeit mit einem Versionsmanagementsystem auch in solchen Lehrveranstaltungen illustiert werden, in denen nur Konzepte eingeführt werden und gar keine praktische Arbeit mit CVS geplant ist. In diesem Fall braucht auch der Vortragende selbst keine Erfahrung mit der Installation von CVS zu haben.

- Für den gleichen Inhalt würde man bei einer live-Vorführung wahrscheinlich deutlich mehr Zeit benötigen. Dies liegt zum einen bestimmten vor- und nachbereitenden Tätigkeiten, die für den Lehrstoff nicht relevant sind und im Film weggelassen werden können, ferner am Zeitbedarf für den Wechsel zwischen Vorführung und Arbeit an Projektor oder Tafel (als Ersatz für die Animationen).

- Eine wiederholte Vorführung (z.B. in 10 - 20 Gruppen in einem Programmierpraktikum) ist zeitraubend und durchaus ermüdend; individuelle Einzelvorführungen, wie sie durch den Streamingserver möglich sind, sind nicht praktikabel.

Aus der Sicht der Lernenden stellt vor allem die jederzeitige Verfügbarkeit des Films einen Vorteil dar. Ferner dürfte der Film i.a. ein höheres Maß an Fehlerfreiheit aufweisen als eine live-Vorführung; die Qualität der Vorführung hängt auch nicht von der Tagesform des Vorführenden ab.

3 Der Streamingserver

3.1 Motivation

Der CVS-Lehrfilm wurde ursprünglich nur einzelnen Gruppen des Programmierpraktikums vorgeführt. Da die Dateien rund 200 MB groß waren, wurde er nicht zum Herunterladen angeboten, um nicht zuviel Netzlast zu erzeugen. Dennoch war es natürlich wünschenswert, daß die Studenten den Film jederzeit erneut ansehen können, im Idealfall direkt an dem Rechner, wo sie die Bedienung von CVS lernen. Daher wurde Ende 2002 in einem separaten Vorhaben untersucht, ob und wie der Film online zugreifbar gemacht werden könnte.

Zu entscheiden war in diesem Zusammenhang über die Server- bzw. Client-seitige Software und damit verbunden über die Formate, in denen das Filmmaterial vorliegen muß. Im folgenden betrachten wir einige Lösungen und begründen die Auswahl, die für die Zwecke des Projekts unter Berücksichtigung der Rechner- und Netzwerkstrukturen an der Fachgruppe Praktische Informatik am sinnvollsten erschien.

3.2 Vergleichskriterien

Für das Videostreaming werden i.w. zwei Systemkomponenten benötigt: der (zentrale) Streamingserver und Abspielsoftware auf den Arbeitsplatzrechnern. Demzufolge können die verschiedenen Streaminglösungen anhand folgender Kriterien verglichen werden:

- Anforderungen an den Rechner, der als Streamingserver fungiert

- Anforderungen an den Arbeitsplatzrechner

- unterstützte Dateiformate, in denen die Videomaterialien vorliegen
- unterstützte Bandbreiten

In den folgenden Beschreibungen bzw. Vergleichen gehen wir nicht auf die Protokolle und Verfahren ein, die den eigentlichen Transport der Videodaten besorgen, sondern nur auf eher äußerliche Merkmale wie die Verfügbarkeit für verschiedene Betriebssysteme, Hardware-Anforderungen und Portabilität.

3.3 Verglichene Systeme

Neben den drei bekanntesten Streaminglösungen, die i.f. näher untersucht werden, gibt es noch andere Lösungen [5], die interessante Ansätze haben, aber noch nicht ausgereift sind oder sehr hohe Anforderungen an die zu Verfügung stehenden Bandbreiten stellen und damit nur für schnelle LANs geeignet sind. Aus diesen Gründen werden sie hier nicht näher betrachtet.

Das sogenannten HTTP-Streaming wird hier nur kurz erwähnt, da es an sich kein Streaming ist, aber für kurze Videosequenzen durchaus eine Überlegung wert ist.

3.3.1 Realmedia

Von der RealNetworks, Inc., wird die Realmedia-Plattform für die gängigsten Betriebssysteme und Hardware-Plattformen, u.a. Sun Solaris, IBM AIX, Windows NT/2000/XP, HP UX, FreeBSD und Linux, angeboten. Teil der Realmedia-Plattform ist der Helix Universal Server.

Für einen optimalen Betrieb des Servers ist ein Rechner mit mindestens 256 MB Arbeitsspeicher nötig. Der Prozessor sollte je nach Architektur mit mindestens 400 MHz getaktet sein. Ein Motorola PowerPC wäre schon ab 375 MHz ausreichend, ein X86 sollte mit mindestens 500 MHz Prozessortakt laufen. Nähere Angaben findet man bei RealNetworks [3]. Eine eingeschränkte Version des Server ist frei verfügbar; die Summe aller Ströme ist hier auf 1 Mb/s beschränkt.

Der zugehörige Client, RealPlayer genannt, ist für die verschiedensten UNIX-, Windows- und Apple-Systeme kostenlos verfügbar. Die Windows- und Linux-Version des RealPlayers läuft bereits auf Computer mit einem Intel Celeron 300MHz und mit 64MB RAM, genaue Angaben zu der erforderlichen Leistungsfähigkeit der Client-Hardware und unterstütze Betriebssysteme sind in [4] zu finden.

Damit die Videos auch streambar sind, müssen sie im geeigneten Format vorliegen. Der Helix Server kann viele verschiedene Audio- und Videoformate streamen, dazu gehören

- MP3
- Macromedia Flash
- Windows Media
- MPEG 4
- Quicktime

- Realmedia

Realmedia-Dateien lassen sich in verschiedenen Bandbreiten erzeugen, die für langsame Modem bis hin zu schnellen LAN Verbindungen geeignet sind, sie unterscheiden sich dann in der Qualität von Bild und Ton. In einem speziellen Format, dem sogenannten SureStream Format, ist es möglich, mehrere dieser unterschiedlichen Qualitäten in *einer* Datei zu speichern. Fordert der RealPlayer nun ein Video vom Streamingserver an, so wird dem Server die maximal zu Verfügung stehende Bandbreite gemeldet. Der Helix Server sucht daraufhin den optimalen Stream aus der SureStream-Datei und sendet ihn dann an den Client.

3.3.2 Windows Media Server

Der Windows Media Server ist Bestandteil von Windows 2000 Server/Advanced Server, diese benötigen laut Microsofts Produktbeschreibung einen Pentium kompatiblen Prozessor mit mindestens 200 MHz und 128 MB Arbeitsspeicher. Diese Angaben sind nicht als realistisch zu betrachten, da Erfahrungen mit Windows NT/2000 zeigen, daß die Mindestanforderung an die Server-Plattform stark nach oben korrigiert werden muß. Ein geeigneter Server muß mindestens 256 MB Arbeitsspeicher und einen X86 kompatiblen Prozessor mit mindestens 500 MHz haben.

Auf der Clientseite muß, um das gestreamte Video betrachten zu können, der Windows Media Player installiert sein. Dieser ist für Mac OS 8/9/X, MS Windows 95/98/98SE/2000/XP und Sun Solaris verfügbar.

Videos, die gestreamt werden sollen, müssen im ASF-Format vorliegen, diese können aber mit verschiedenen Codecs komprimiert werden. Das ASF-Format beinhaltet analog zum SureStream ein Video in mehreren Qualitätsversionen und ermöglicht es, unterschiedliche Bandbreiten mit einer Datei abzudecken.

3.3.3 Apple Quicktime Streaming Server

Der Quicktime Streaming Server ist Open Source und wird im Rahmen des Darwin Projektes von Apple angeboten.

Dieser Server hat ähnliche Hardware-Anforderungen wie die beiden oben beschriebenen Server und ist für Mac OS X, Linux, Sun Solaris und Windows verfügbar.

Auf dem Arbeitsplatzrechner wird der Quicktime Player von Apple benötigt; er ist für die Betriebssysteme Mac OS 8/9/X, Sun Solaris und MS Windows verfügbar.

Dieser Server unterstützt neben Quicktime Movie auch MPEG-4 und MP3 Dateien. Das Apple Quicktime Movie Format kann analog zu Realmedia unterschiedliche Bandbreitenversionen eines Videos in einer Datei vereinigen.

3.3.4 HTTP-Streaming

Mit dieser Methode ist es relativ einfach, ein Video online verfügbar zu machen: eine Videodatei wird auf einem WWW-Server per HTTP-Zugriff abrufbar gemacht.

Fordert nun ein Client dieses Video an, so wird zunächst die Datei heruntergeladen, und der Client beginnt, das Video darzustellen. Damit nicht auf die komplette Übertragung der Datei gewartet werden muß, muß das Video in einem Format vorliegen, das schon angezeigt

werden kann, ohne die komplette Datei heruntergeladen zu haben. Dies trifft auf die Formate Realmedia und Quicktime Movie zu. Zum betrachten des Streams muß der entsprechende Media Player installiert sein. Die Nachteile des HTTP-Streamings liegen im HTTP-Protokoll: So ist es nicht möglich,

- das Video vor- oder zurückzuspulen oder
- die Übertragung an die vorhandene Bandbreite anzupassen. Harmoniert die tatsächlich vorhandene Bandbreite nicht mit der des Videos, so versucht der Player, dies zu kompensieren, indem er Daten puffert. Dies kann durchaus dazu führen, daß es sich nicht mehr um ein Streaming handelt, sondern um einen einfachen Download.

3.3.5 Auswahl für den CVS-Film

Bei dieser Auswahl waren die an der Fachgruppe Praktische Informatik vorhandene Server-Hardware und die in den Rechnerpools vorhandenen Arbeitsplatzrechner entscheidend.

Als weniger problematisch stellte sich die Frage der Bandbreite dar, da das Netzwerk in der Fachgruppe bzw. im Fachbereich gut ausgebaut ist (i.d.R. 100 Mb-Anbindung der Arbeitsplatzrechner, zentraler 100 Mb-Switch).

Für den Server standen drei Rechner zur Wahl:

1. Linux (RedHat) auf X86-PC-System
2. Sun Solaris auf X86-PC-System
3. Sun Solaris auf UltraSparc-System

Die Rechnerpools bestehen aus X86-PC-Systemen mit Linux. Diese Rahmenbedingungen legen es nahe, sich für die Kombination aus RealServer und RealPlayerzu entscheiden.

Installiert wurde der Server letztlich auf dem Linux-System (2*500 MHz P3, ca. 800 MB Hauptspeicher, U2-SCSI, 100 MBit Ethernet).

Tests mit Arbeitsplatzrechnern. Um einen breiten Einsatz des Lehrfilms zu ermöglichen, sollte er möglichst auch auf 3 - 4 Jahre alten Rechnern abgespielt werden können und nicht nur auf brandneuen, sehr leistungsfähigen Systemen. Um die Nutzbarkeit älterer Arbeitsplatzrechnern zu erkunden, wurden mehrere Benchmarktests durchgeführt.

Als Testvideo wurde die 7. Szene des Films ausgewählt. Diese Szene wurde als Realvideo im SureStream Format mit 15fps und einer Auflösung von 800x576 kodiert. Im RealPlayer wurde jeweils einmal

- 450 kb/s und
- 80 kb/s

als maximal verfügbare Bandbreite ausgewählt. Die Tests wurden auf 3 verschiedenen PCs und einem Mac G3 durchgeführt. Die Tests zeigten, daß bereits Arbeitsplatzrechner mit Intel Celeron 300 MHz und einer “einfachen” Grafikkarte (z.B. ATI Mach64, Nvidia TNT, S3) und

64 MB RAM unter Linux und Windows schnell genug sind, um den Lehrfilm einwandfrei abspielen zu können. Die CPU-Last lag kurzfristig bei ca. 80% und im Durchschnitt bei ca. 50%. Auch bei dem Mac G3 wurden keine Probleme festgestellt, der Lehrfilm konnte problemlos abgespielt werden.

Ergänzend wurden mit einigen externen Kooperationspartnern von anderen Standorten Tests gemacht, wobei wegen der größeren Entfernung ein zusätzliches Problempotential in geringer oder schwankender Bandbreite lag. In den (wenigen) Tests traten aber keine Probleme auf.

Um herauszufinden, ob die Leistung des Servers bei mehreren parallelen Strömen an Engpässe stoßen würde, wurde ein Test mit 4 parallel Streams durchgeführt. Derartige Engpässe traten nicht auf, was bei der vorhandenen Hardware- und Netzwerkkonfiguration aber auch nicht zu erwarten war.

3.4 Zusammenfassung der Erfahrungen

Die Erfahrungen mit dem Realmedia-Server / RealPlayer waren positiv.

- Der Streamingserver ist seit Ende November 2002 im Einsatz, ohne daß Probleme erkennbar wurden.
- Ein ca. 3 Jahre alter, nicht mehr ganz taufrischer Serverrechner reichte als Hardware-Basis völlig aus für die geringe Last (zwei oder mehr parallele Abspielvorgänge treten in der Praxis nicht auf).
- Aufgrund der geringen Parallelität reicht eine kostenfreie Serverlizenz aus.
- Das zugehörige Kompressionsverfahren erwies sich bei dem hier vorliegenden Bildmaterial als das beste.
- Vorteilhaft ist die Möglichkeit, Komprimierungen für verschiedene Bandbreiten erzeugen zu können.

  Als Untergrenze für die Bandbreite erwiesen sich 128 kb/s als notwendig. Bei 64 kb/s (= 1 ISDN-Kanal) läßt die Qualität sehr deutlich nach, insb. der Ton wird dumpf und schlecht verständlich. Bandbreiten oberhalb von 256 kb/s bringen kaum noch Qualitätsverbesserungen ein.

Literatur

[1] Kelter, U.: Gestaltungsrichtlinien für Editor-Simulatoren für graphartige Dokumente; p.393-400 in: Schubert, S.; Reusch, B.; Jesse, N. (ed.s): Proc. GI Jahrestagung, 30.9.-3.10.2002, Dortmund; Lecture Notes in Informatics; Gesellschaft für Informatik; 2002/10

[2] GNU General Public License (http://www.opensource.org/licenses/gpl-license.html)

[3] RealNetworks http://www.realnetworks.com

[4] Realmedia http://www.real.com/realone/index.html

[5] weitere Streamingserver:
MPEG4IP http://mpeg4ip.sourceforge.net/
FFMPEG http://ffmpeg.sourceforge.net/
VideoLAN http://www.videolan.org
MEC4 http://utenti.lycos.it/jaggomiken/index2.html
Palantir http://www.fastpath.it/products/palantir/

Der MuSofT-Jahresbericht 2002 - KT 1

Olaf Scheel, Johannes Magenheim

Dies ist der Jahresabschlussbericht 2002 des Koordinationsteams 1 - *Gestaltung von Lernmodulen*.

Inhaltsverzeichnis

1 Einleitung

Das Koordinationsteam *Gestaltung von Lernmodulen* hatte zu Beginn des MuSofT-Projektes vornehmlich die Aufgabe, den Projektpartnern Hilfen für die nach didaktisch-methodischen Gesichtpunkten ausgerichteten Lerneinheiten, Lernmodule und Gruppenobjekte anzubieten. Das Jahr 2002 ist bei allen Partnern in erster Linie durch die konkrete Arbeit an den einzelnen Lernobjekten geprägt, in denen die Vorschläge des KTs integriert werden sollten. Aus diesem Grund kann das Koordinationsteam wenig aus dem vergangenen Jahr berichten. Dieser Bericht soll deswegen vor allem zeigen, dass das KT seine Arbeit nicht als beendet betrachtet, und einen Ausblick auf mögliches Tätigkeitsfelder im Jahr 2003 sein.

2 Aufgabe des Koordinationsteams

Die Aufgabe des Koordinationsteams *Gestaltung von Lernmodulen* ist die Überwachung und Steuerung der Gleichförmigkeit, des Aufbaus und der didaktischen Eignung der Lernobjekte auf allen Hierarchiestufen (Einheiten, Module und Gruppenobjekte). Zu diesem Zweck

sollen den Projektpartnern Hilfen bereitgestellt werden, damit diese ihre Lernobjekte nach didaktisch-methodischen Kriterien ausrichten und produzieren können. Diese Hilfen sind vornehmlich auf die Startphase des Projektes ausgerichtet, sollen sich aber auch konkreten Fragestellungen während der Arbeit an den Lerneinheiten widmen. Darüber hinaus aber soll das KT auch ein Evaluations- und Bewertungsschema für bereits fertig gestellte Lernobjekte anbieten, die für die Projektevaluation aber auch für die nachhaltige Weiterentwicklung der Objekte/Module/Einheiten notwendig ist.

3 Bisherige Aktivitäten und Ergebnisse

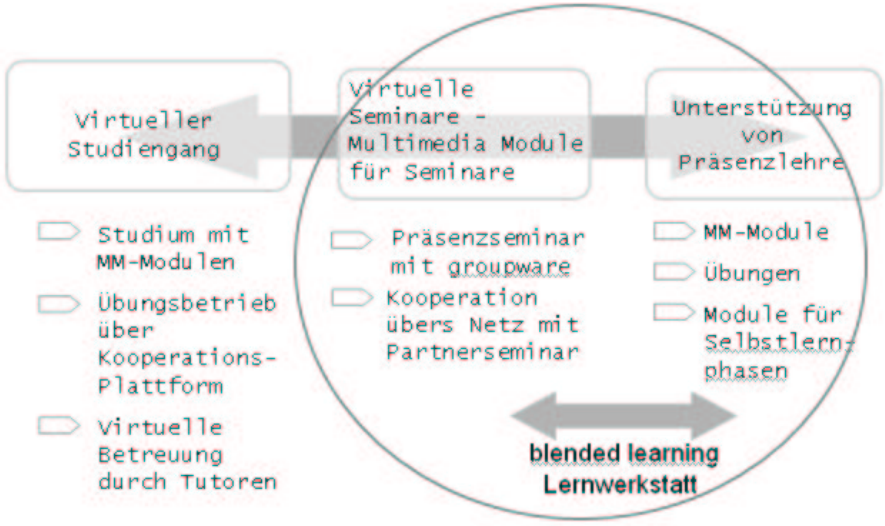


Abbildung 1: Zielbereich des MuSofT-Projektes

Das didaktische Konzept von Lernobjekten für das MuSofT-Portal spiegelt zunächst einmal die Sicht der Lehrenden auf die Objekte wieder. Teilaspekte sind hier:

- Leitbild
- Lernziele
- Lernszenario
- Methodisch-didaktisches Arrangement

3.1 Leitbild

Das Leitbild über den Lernenden beinhaltet den individuellen, den institutionellen und den technischen Kontext des Lernens. Im individuellen Kontext befinden sich kognitive und methodische Vorkenntnisse des Lernenden oder der Lerngruppe, angestrebte berufliche Qualifikationen und Einsatzfelder. Der institutioneller Kontext wird durch die Ausbildungsinstitution (Universität, Fachhochschule, Weiterbildung, Unternehmen...), den Studiengang, das Studienfach und letztlich auch den Studienabschnitt gebildet. Der technische Kontext durch die individuellen und institutionellen technischen Bedingungen. Der Einsatz, die Funktion und der Nutzen aller Lernobjekte ist von den jeweiligen Einsatzkontexten abhängig und determiniert.

3.2 Lernziele

Alle Lernobjekte sollten auf bestimmte Lernziele abhängig vom jeweiligen Leitbild ausgerichtet sein. Mit Lernzielen sind hier aber nicht nur die oft überrepräsentierten auf die Informatik bezogenen und fächerübergreifenden kognitiven Lernziele gemeint, sondern auch fachmethodische (informatikbezogene, fachübergreifende), sozial- kommunikative und normativbewertende.

3.3 Lernszenario

Die Lernszenarios beinhalten Veranstaltungstypen (Präsenz-Vorlesung, Übung, Praktikum, Seminar, webasiertes Studienmaterial, Projekte...), die Rolle der Lehrenden (Präsentieren, Organisieren von Lernprozessen, Beraten, Prüfen, Entwurfsentscheidungen), die Rolle der Lernenden (Rezipient, Selbststudium, Erkundender, Selbstorganisation von Lern- und Arbeitsprozessen), den Typ der Lernobjekte (Hypertext, sequentieller Text, Bilder, Grafiken, Animation, interaktive Elemente, Aufgaben, Tests, webbasierte Dokumente) und auch die Funktion dieser Objekte (Inhaltspräsentation, inhaltliche Differenzierung, Vertiefung, Erweiterung, Repetition, Übung, Prüfung...)

3.4 Methodisch-didaktisches Arrangement

Das Arrangement von Lehrmaterialien bzw. Lernobjekten schließlich beinhaltet nicht nur bestimmte didaktische Vorstellungen, sondern auch die Methoden, mit denen die Lerninhalte vermittelt werden sollen:

- sequentiell lehrgangsmäßig - hypermedial erkundend
- problemorientiert - prozessorientiert
- fallbasiert - exemplarisch
- instruktional - konstruktivistisch
- konstruktiv induktiv - dekonstruktiv analytisch
- projektartig

- virtual company - real situation
- virtuelle Erkundungsumgebung
- produktorientiert

Im Idealfall sollte die Struktur der Metadaten so mächtig sein, dass die Plattform abhängig von Inhalt, Leitbild, didaktischem Konzept und Methode eine Menge geeigneter Lernobjekte für das aktuell zu gestaltende Modul anbietet. Für Nutzer von größeren Einheiten (Module, Lerneinheiten) sollten die zugeordneten Metadaten eine Vorstellung über den jeweiligen Einsatzkontext vermitteln.

Dem entgegen steht die Tatsache, dass einzelne Lernobjekte je nach Kontext sehr unterschiedlich eingesetzt werden können. Letztlich kann also die Plattform dem Lehrenden die didaktisch-methodische Analyse nicht abnehmen. Somit können die oben genannten Teilaspekte nicht unmittelbar in didaktische Metadaten umgewandelt werden. Stattdessen sollen sie einen Gestaltungsraum aufspannen, in dem Lernobjekte angesiedelt werden können. Somit sind sie eine Hilfe für die *Entwicklung* geeigneter Objekte.

Die Idee der Lernszenarien und der didaktischen Metadaten ist auf nicht-informatische Bereiche übertragbar und damit auf andere Projekte.

3.5 Teilnahme an Workshops

Das didaktisch-methodische Konzept und die Inhalte des MuSofT-Projektes wurden von Johannes Magenheim auf zwei BMBF-Workshops des Jahres 2002 vorgetragen: Auf dem kevih-Workshop (Konzepte und Elemente virtueller Hochschule) in Tübingen vom November 2002 und auf dem Workshop *Didaktik und Evaluation von e-Learning* in Erlangen.

4 Planung für das nächste Jahr

Der Arbeitsbereich des Koordinationsteams *Gestaltung von Lernmodulen* wird sich im neuen Jahr in Richtung Evaluation und Bewertung verschieben.

Die Bewertung umfasst dabei die exemplarische Bewertung von bereits produzierten Lernobjekten hinsichtlich des methodisch-didaktischen Arrangements, aber auch der anderen oben genannten Aspekte, die gegebenenfalls zu einer Anpassung oder Überarbeitung führen können. Der Schwerpunkt kann aber hier nicht auf inhaltlichen Aspekten liegen, für die jedes Teilprojekt eigenverantwortlich bleiben soll, gemeint ist hingegen eher eine didaktisch/methodische Hilfestellung am konkreten Beispiel.

Hinsichtlich der Metadaten ist die Entwicklung einer didaktischen Ontologie notwendig, die zur Verwendung einheitlicher Begriffe bei der Eingabe von Metadaten führen soll.

Eine weitere Aufgabe des KTs kann die Entwicklung eines pragmatischen Evaluationskonzeptes sein. Ansatz für die Entwicklung von Evaluationskriterien könnte hier die Integration von methodischen Hinweisen und Kommentaren von Benutzern in das MuSofT-Portal bieten, wie es beispielsweise im Life3-Projekt der AG-Magenheim (http://life.upb.de/) konzipiert ist.

KT2 - Abstimmung von Lehrmoduln

Andy Schürr

Das MuSofT-Koordinationsteam KT2 befasst sich mit der inhaltlichen und stilistischen Abstimmung aller entwickelten Lernobjekte, um so die Kombinierbarkeit von Lernmaterialien zu gewährleisten, die von verschiedenen Projektpartnern für ganz unterschiedliche Zielgruppen entwickelt werden. Bei der im Vordergrund stehenden inhaltlichen Abstimmung von Lernobjekten spielt deren einheitliche Beschreibung durch sogenannte Metadaten eine herausragende Rolle. Hier ein Metadatenkonzept zu finden, das zu internationalen Standards kompatibel und dennoch in der Praxis des Projektalltags einsetzbar ist, erwies sich als anspruchsvolle Aufgabe. Ihr waren die meisten Aktivitäten von KT2 im vergangenen Jahr gewidmet, die im vorliegenden Tätigkeitsbericht überblicksmäßig dargestellt werden.

Inhaltsverzeichnis

1 Einleitung

Um die in MuSofT an unterschiedlichen Standorten entwickelten Lernobjekte zur Realisierung einer Vorlesung einsetzen zu können, sind die an den verschiedenen Projektstandorten entstehenden Lerneinheiten *aufeinander abzustimmen*. Diese Abstimmung erfordert übergreifende Aktivitäten, die maßgeblich zur Qualitätsförderung in MuSofT beitragen. Deshalb wurden zu Beginn des Projekts vier Koordinationsteams eingerichtet, welche über die einzelnen Standorte hinweg tätig sind.

Wie wir im folgenden noch sehen werden, lassen sich die Aufgaben der einzelnen KTs nicht ganz unabhängig voneinander bearbeiten. Insbesondere das Koordinationsteam KT2 besitzt mit seiner zentralen Aufgabe der *Abstimmung von Lernobjekten* Berührungspunkte zu allen anderen KTs. Auf diese Berührungspunkte können wir jedoch erst nach einer detailierteren Darstellung der Aufgaben von KT2 genauer eingehen.

Der Bericht über die Aktivitäten des Koordinationsteams KT2 im zweiten Projektjahr ist deshalb wie folgt aufgebaut: Zunächst werden im folgenden Abschnitt die Aufgaben und Ziele von KT2 genauer beleuchtet sowie die sich daraus ergebenden Berührungspunkte zu den anderen KTs. Darauf aufbauend beschreibt der Hauptabschnitt dieses Berichts zunächst kurz unsere Aktivitäten zur Festlegung einheitlicher Vorgaben für Lernobjekte. Schwerpunkt dieses Abschnittes bildet allerdings die Entwicklung eines Metadatenkonzeptes für die einheitliche Beschreibung von Lernobjekten. Die beiden letzten Abschnitte des Berichtes befassen sich schließlich mit den künftigen Aktivitäten von KT2 innerhalb der verbleibenden Projektlaufzeit und skizzieren, wie auf Basis der dabei gemachten Erfahrungen weiterführende Aktivitäten in künftigen Projekten aussehen könnten.

2 Aufgabe des Koordinationsteams

Die im vorhergehenden Abschnitt geforderte inhaltliche und stilistische Abstimmung von Lernobjekten durch KT2 ist kein Selbstzweck. Vielmehr sollen hierdurch die wesentlichen Grundlagen dafür bereitgestellt werden, dass (1) sich Lernobjekte für neue Zielgruppen durch *Kombination und Adaption bereits vorhandener Lernobjekte* erzeugen lassen und (2) die Anforderungen an ein gemeinsames MuSofT-Portal genauer festgelegt werden können. Um diesem Anspruch gerecht zu werden, wurden folgende drei Primärziele von KT2 formuliert:

1. die Nutzung von Lernobjekten in anderen Lernobjekten verschiedener Teilprojekte und die gleichzeitige Vermeidung von Redundanzen, die durch Doppelentwicklungen in verschiedenen Teilprojekten entstehen können,

2. die Identifikation fehlender Lernobjekte zur Vermittlung von Basiswissen, das von den erstellten Lernobjekten vorausgesetzt wird, und die Vergabe entsprechender Aufträge an Teilprojekte,

3. die Erstellung neuer Lernobjekte durch die Wiederverwendung vorhandener Ojekte bzw. die Anpassung bereits erstellter Lernobjekte an neue Zielgruppen.

Aus diesen Primärzielen lässt sich nunmehr ableiten, dass die inhaltliche und stilistische Abstimmung von Lernobjekten bzw. ihrer Bestandteile notwendig ist und welche Punkte da-

bei zu berücksichtigen sind. Auf *inhaltlicher* Ebene spielen die Dokumentation verschiedener Arten von Beziehungen zwischen einzelnen Lernobjekten, die Auswahl durchgängiger Fallstudien, die Festlegung der verwendeten Modellierungs- und Programmiersprachen sowie die einheitliche Beschreibung aller erstellten Lernobjekte durch sogenannte "Metadaten" eine herausragende Rolle. Auf der *stilistischen* Ebene geht es hingegen darum, Modellierungs- und Programmierkonventionen festzulegen sowie die Menge der alternativ eingesetzten Modellierungs- und Programmierwerkzeuge einzuschränken. Der stilistischen Ebene lässt sich schließlich auch der Punkt "einheitliches Layout" von Lehrmaterialien zuordnen.

Auf Basis dieser Festlegungen von KT2 und ähnlicher Überlegungen der anderen KTs wurden die folgenden Bereiche identifiziert, in denen sich die Aufgaben von KT2 mit denen der anderen Koordinationsteams überschneiden:

- KT1 ist mit seiner Aufgabe der Festlegung didaktischer Richtlinien ebenfalls (wenn nicht sogar in erster Linie) für den einheitlichen Aufbau und die einheitliche Gestaltung der erstellten Lernobjekte zuständig. Des weiteren sind die Ergebnisse von KT1 für die Beschreibung der didaktischen Eigenschaften (Attribute) von Lernobjekten von entscheidender Bedeutung.
- KT3 spielt bei der Festlegung von Lehrinhalten sowie bei der Auswahl kleinerer Beispiel und größerer Fallstudien eine wichtige Rolle; hier wird deren Eignung für ganz unterschiedliche Zielgruppen untersucht und sichergestellt. Ebenso beeinflussen die Ergebnisse von KT3 die Art und Weise, wie Lernobjekte durch Metadaten beschrieben werden sollten.
- KT4 schließlich hat die Aufgabe übernommen, auf der Grundlage der von KT2 vorgegebenen Festlegungen zu Metadaten, Werkzeugen, etc. das MuSofT-Portal zu realisieren, also eine entsprechende Integrationsplattform für alle MuSofT-Ergebnisse bereitzustellen.

3 Bisherige Aktivitäten und Ergebnisse

Im vorhergehenden Abschnitt haben wir gesehen, dass die Beschreibung der Metadaten von Lernobjekten ein, wenn nicht der zentrale Punkt ist, in dem sich die Aufgaben der verschiedenen Koordinationsteams überschneiden und selbst wieder einer Koordination bedürfen. Deshalb haben wir in KT2 diesem Aspekt im vergangenen Jahr 2002 die höchste Priorität eingeräumt. Auf der Basis der Ergebnisse von KT1 und KT3 wurde ein Vorschlag für den Aufbau der Metadaten zur Beschreibung von Lernobjekten in MuSofT erarbeitet, der die Grundlage für die Arbeiten von KT4 zur Realisierung des MuSofT-Portals bildete.

Im Folgenden werden wir aus diesem Grund nur kurz auf die anderen Aktivitäten von KT2 im vergangenen Jahr zur Auswahl von Programmiersprachen, Werkzeugen, Fallbeispiele etc. eingehen und den Hauptaugenmerk auf das Thema *Metadaten für Lernobjekte* legen.

3.1 Notationen, Werkzeuge und Fallstudien

Trotz der Vielfalt der behandelten Softwaretechnikthemen, der mannigfaltigen Unterschiede der Curricula der beteiligten Universitäten und der großen Menge konkurrierender Methoden

und Techniken in der Softwaretechnik ist es uns gelungen, für alle in MuSofT entwickelten Lehrmaterialien die folgenden Vereinbarungen zu treffen:

- Als Programmiersprache wird - wo immer sinnvoll - Java eingesetzt und damit der objektorientierten Softwareentwicklung der Vorzug gegeben. Eine begründete Ausnahme von dieser Regel bildet der Datenbankbereich; dort haben sich objektorientierte Technologien gegenüber relationalen Datenbanken mit SQL als "Programmiersprache" bislang nicht durchgesetzt.

- Als Modellierungssprache wird die UML verwendet, als Vorgehensmodell dem dazugehörigen "Unified Process" der Vorzug gegeben. Auch von dieser Regel wird nur in einigen wenigen Fällen abgewichen; im Datenbankbereich werden beispielsweise auch "Entity-Relationship"-Diagramme verwendet, während in Lehreinheiten, die das Thema Vorgehensmodelle selbst ansprechen, natürlich auch andere Standards wie das V-Modell zur Sprache kommen.

- Auf der Ebene CASE-Tools fiel unsere Wahl auf das Together Control Center als das Standardwerkzeug, da es eine enge Integration von UML-Modellierungshilfsmitteln mit Java-Entwicklungswerkzeugen anbietet und zudem über einige Anpassungs- und Erweiterungsmöglichkeiten verfügt.

- Auf der Ebene der Metawerkzeuge zur Erstellung spezialisierter CASE-Tools für bestimmte Unterrichtseinheiten wurde nach einer Diskussion verschiedener Alternativen (Dome, MetaEdit, ArgoUML-Kern, jViews, yFiles, etc.) folgendes Resümee gezogen: die Ansprüche der einzelnen Projektpartner bei der Erstellung spezieller Projektplaneditoren, Architektureditoren, interaktiver Algorithmenanimationen usw. sind zu verschieden, um diese durch ein Meta-CASE-Tool abdecken zu können. Die Festlegung auf ein Werkzeug war deshalb an dieser Stelle nicht sinnvoll; umso mehr, als die "Nutzer" unserer Lehrmaterialien ein solches Meta-Werkzeug niemals direkt zu Gesicht bekommen.

- Ebenfalls als schwierig hat sich das Thema "einheitliche Fallstudien" erwiesen. Auch hier sind die Erfordernisse verschiedener Lerneinheiten so divergierend, dass die Festlegung *einer* zentralen Fallstudie als durchgängiges Beispiel für alle entwickelten Lehrmaterialien sich als nicht sinnvoll erwies. Immerhin ist es uns jedoch gelungen, dem Thema Logistik mit den Unterthemen Lagerverwaltung, Kommissionierung und Speditionswesens eine zentrale Rolle einzuräumen.

3.2 Vorüberlegungen zur Festlegung eines Metadatenformates

Die Beschreibung von Lernobjekten durch umfangreiche Sätze von Metadaten ist eine Funktion, die heutzutage von nahezu allen Lernplattformen, -portalen, etc. angeboten wird. Um die Interoperabilität all dieser Werkzeuge in Zukunft zu gewährleisten, gibt es eine ganze Reihe von Standardisierungsbemühungen für Metadatenformate von Lernobjekten. Aus diesem Grunde war es nicht notwendig, in MuSofT ein eigenes Metadatenformat zu entwickeln. Vielmehr hat es sich als sinnvoll erwiesen, auf den IEEE LOM-Standard (*Learning Objects Metadata*) [IEE02] zurückzugreifen, der ein konzeptionelles Datenschema vorgibt, welches die

Struktur von Metadaten für Lernobjekte beschreibt. Die Verwendung dieses Metadatenstandards erlaubt nicht nur eine einheitliche Beschreibung der Lernobjekte innerhalb von MuSofT, sondern ermöglicht auch eine Interpretation der verwendeten Metadaten über die Projektgrenzen hinaus. Unser Hauptgrund für die Auswahl des LOM-Standards war zum einen seine weite Verbreitung. Zum anderen wird der LOM-Standard nahezu unverändert in verschiedene Standardisierungsbemühungen, wie z.B. dem IMS Global Learning Consortium [IMS02], SCORM [Adv02] oder ARIADNE [DFC+01] integriert.

Obwohl LOM aus den oben genannten Gründen der für unsere Zwecke am besten geeignete Standard für die Beschreibung von Lernobjekten ist, war sein Einsatz im MuSoft-Projekt mit umfangreichen Vorarbeiten verbunden. So erschien es uns vor allem unrealistisch, *alle* von LOM vorgeschlagenen über hundert Metaattribute für die Dokumentation unserer Lernobjekte einzusetzen und zudem die tatsächliche Verwendung der LOM-Attribute ohne weitere Richtlinien den einzelnen Teilprojekten zu überlassen. Ein solcher ungeregelter Einsatz von LOM würde möglicherweise zu weit voneinander abweichenden Metabeschreibungen einzelner Lernobjekte führen und damit die oben aufgeführten Zielsetzungen konterkarieren.

3.3 Die Anpassung des LOM-Standards - Objekthierarchien

Der LOM-Standard schreibt eine genau vierstufige Hierarchie von *Lernobjekten (LOs, Learning Objects)* vor, ohne allerdings im Detail festzulegen, wie eine solche Hierarchie für die Gliederung von Lernmaterial eingesetzt werden soll. Nach einer längeren Diskussion - insbesondere darüber, ob in der Praxis eine Beschränkung auf vier Hierarchiestufen akzeptabel ist - haben wir in KT2 beschlossen, die LOM-Hierarchie wie folgt auf unsere Bedürfnisse zugeschnitten auszulegen:

1. Auf der obersten Ebene gibt es *Lerneinheiten (LEs, Learning Entities)*: eine LE ist eine sich abgeschlossene Sammlung von Lernmaterialien zu einem Themenkomplex. Typischerweise handelt es sich dabei um einen Bereich, der im Rahmen *einer* Lehrveranstaltung behandelt wird.

2. Lerneinheiten setzen sich aus *Lernmoduln (LMs, Learning Modules)* zusammen: ein LM ist eine Sammlung von Lernmaterialien, die in verschiedenen Lerneinheiten (wieder-)verwendet werden kann. Sie behandelt also ein in sich abgeschlossenes Teilgebiet, das Bestandteil verschiedener Lehrveranstaltungen sein könnte.

3. Für die weitere Unterteilung von Lernmoduln gibt es die *Gruppenobjekte (GOs, Group Objects)*: ein GO ist die kleinste Ansammlung von Lernmaterialien, die nicht als atomare Einheit betrachtet wird. In aller Regel wird ein Gruppenobjekt Bestandteil genau eines Lernmoduls sein, da es weder wohldefinierte Schnittstellen zu anderen Lernobjekten besitzt, noch einen in sich abgeschlossenen Themenkomplex behandelt.

4. Auf der untersten Ebene unserer Enthaltenseinshierarchie gibt es sogenannte *Medienobjekte (MEs, Media Objects)*: ein ME ist die kleinste Untereinheit einer Lerneinheit, deren interne Struktur auf der Ebene der Metadaten nicht weiter betrachtet wird. Es kann sich dabei um einen vollständigen Foliensatz, eine html-Seite, eine Animation, ... handeln. Üblicherweise wird ein Medienobjekt einer Datei entsprechen, es kann

aber auch ein Verweis auf einen Ausschnitt einer Datei (z.B. Folie 10 bis 20 einer Powerpoint-Präsentation) sein. Oft werden solche Medienobjekte Bestandteil mehrerer Gruppenobjekte sein.

Die vier Stufen von LOM bilden damit also eine azyklische, aber i.a. nicht baumartige Hierarchie, deren Objekte im Folgenden durch Metaattribute und -beziehungen genauer beschrieben werden. Vergleicht man die hier getroffenen Festlegungen mit den unverbindlichen Vorschlägen von LOM zum Einsatz der vier Hierarchiestufen, so ist folgendes festzustellen: auf der niedrigsten Hierarchiestufe 1 (ME) sieht LOM nur einzelne Fragmente von Dokumenten wie etwa html-Seiten vor, während wir für größere Einheiten plädieren, die bereits ganzen Dateien entsprechen. Eine zu detailierte Modellierung von Lernobjekten erschien uns zu aufwändig und für die Ablage und Wiederverwendung von LOs auch wenig zielführend. Auf der obersten Hierarchiestufe 4 (LE) sieht LOM hingegen wesentlich größere Einheiten vor, nämlich ganze Curricula, also Mengen von Lerneinheiten in unserem Sinne. Auch diesem Vorschlag sind wir nicht gefolgt, da uns damit a) nur noch zwei Stufen für die Beschreibung von Medienobjekten, Lernobjekten, Lernmoduln und Lerneinheiten zur Verfügung gestanden hätten und b) uns die Ablage ganzer Curricula als in sich abgeschlossene Lernobjekte im MuSofT-Portal wenig sinnvoll erschien.

3.4 Die Anpassung des LOM-Standards - Metaattribute

Der LOM-Standard umfasst in seiner aktuellen Version über hundert Metaattribute, die in neun Kategorien unterteilt sind. Von diesen neun Kategorien haben wir nach langen Diskussionen fünf mit bislang 22 Metaattributen für die Verwendung in MuSofT ausgewählt. Dabei haben wir darauf geachtet, für jedes ausgewählte Attribut dessen Bedeutung so präzise wie möglich festzulegen und bei der Festlegung seiner Bedeutung konform zu den Erläuterungen des Standards zu bleiben. Auf die Hinzunahme neuer Metaattribute oder die Uminterpretation existierender Metaattribute haben wir bislang also gänzlich verzichtet, wenn auch gerade im Bereich der sogenannten “didaktischen Eigenschaften” der Status Quo von LOM für unsere Bedürfnisse bei weitem nicht ausreichend ist.

Im einzelnen verwendet die MuSofT-Adaption von LOM folgende Kategorien von Metaattributen:

1. *General* mit 8 Attributen für allgemeine Informationen wie eindeutige LO-Bezeichner, Titel, Schlüsselworte, Anmerkungen, etc.

2. *Technical* mit 3 Attributen für technische Eigenschaften wie Formate und “Adresse” von Dateien

3. *Educational* mit 6 Attributen für didaktische Eigenschaften wie Arbeitsformen, Einsatzkontext, Anspruch etc.

4. *Relation* mit 3 Attributen für die Beschreibung von Beziehungen zwischen Lernobjekten (unterstützt wird Versionierung, Hierarchiebildung, sowie drei verschiedene Arten von Querbeziehungen zwischen LOs)

5. *Classification* mit 2 Attributen für die Anbindung von LOs an verschiedene Klassifikationshierarchien (wie etwa das ACM-Klassifikationssystem)

Weitere Informationen zum Einsatz von Metadaten für die Beschreibung und Ermittlung von Lernobjekten im MuSofT-Portal sowie zur Ausgestaltung der Benutzeroberfläche des MuSofT-Portals findet man im Bericht des Koordinationsteams KT 4. Für eine vollständige Auflistung aller ausgewählten Metaattribute mit der Beschreibung der zulässigen Werte sei ebenfalls auf das MuSofT-Portal und die zugehörigen WWW-Seiten verwiesen.

3.5 Offene Punkte bei der Adaption des LOM-Standards

Als besonders schwierig bei der Anpassung des LOM-Standards für MuSofT erwies sich zum einen die Festlegung einer festen Hierarchie von Lernobjekten sowie zum anderen die Auswahl und Beschreibung der Metaattribute der Kategorie *Educational*. So ist es fraglich, ob eine vierstufige Hierarchie mit genau festgelegten Rollen der Lernobjekte auf den einzelnen Ebenen immer ausreichend ist oder ob man nicht lieber die Erstellung beliebig tiefer Hierarchien unterstützen sollte. Mit mindestens sechsstufigen Hierarchien könnte man beispielsweise die von uns benötigte Unterscheidung von Lernmoduln, Gruppenobjekten und Medienobjekten mit der von LOM empfohlenen Verwendung von allumfassenden “Curricula”-Objekten auf der einen Seite und sehr kleinen Dokumentfragmenten auf der anderen Seite zusammenführen; zudem würde eine beliebig tiefe Hierarchisierung beispielsweise auf der Ebene unserer Lernobjekte es uns erlauben, die Unterteilung Lehrmateralien (insbesondere von Büchern) in Kapitel, Abschnitte, Unterabschnitte, etc. mitzumodellieren.

Unbehagen löst auch die Tatsache aus, dass alle Metaattribute für die Beschreibung von Lernobjekten auf allen Hierarchiestufen zugelassen sind. Technische Attribute, wie etwa das Format eines Lernobjektes, scheinen auf den oberen Ebenen wenig Sinn zu machen, während hingegen didaktische Attribute, die z.B. den eingesetzten Studiengang eines Lernobjektes beschreiben, auf den unteren Ebenen fraglich erscheinen. Hier sind weitergehende Festlegungen dringend notwendig, die den Einsatz einzelner Metaattribute entweder für bestimmte Hierarchieebenen ganz verbieten oder aber ein entsprechendes Vererbungskonzept anbieten. Die genaue Ausgestaltung eines solchen Vererbungskonzeptes bedarf jedoch noch umfangreicher Überlegungen und Diskussionen. Beispielsweise könnte man sich bei einem Attribut wie den “Sprachen”, in denen die bereitgestellten Lernobjekte formuliert sind, folgende Vererbungsszenarien entweder alternativ oder sogar miteinander kombiniert vorstellen:

- “Sprache” ist ein *ererbtes* mengenwertiges Attribut, das auf oberster Ebene festgelegt werden kann und in die unteren Ebenen entweder immer unverändert propagiert wird oder dort ggf. eingeschränkt bzw. vielleicht sogar erweitert werden kann.
- “Sprache” ist ein *synthetisches* mengenwertiges Attribut; die Sprachen eines primitiven Lernobjektes werden explizit angegeben, die Sprachen zusammengesetzter Lernobjekte ergeben sich immer implizit aus der Vereinigung der Sprachen ihrer Bestandteile.

Abgesehen von diesen Überlegungen zur Einschränkung der Verwendung von Metaattributen und zum Einsatzung von Vererbungskonzepten für die Reduktion des Erstellungsaufwandes von Metadaten hat sich gezeigt, dass insbesondere die didaktischen Attribute eine unklare

Semantik besitzen und in der vorliegenden Form kaum nutzbringend sind. An dieser Stelle ist das Koordinationsteam KT1 gefragt, konkrete Vorschläge für eine sinnvollere Definition didaktischer Attribute zu erarbeiten und in den Rahmen von LOM einzubetten.

Wir erwarten dennoch mit der skizzierten Anpassung des LOM-Standards einen gangbaren Weg gefunden zu haben, der den Aufwand für die Erstellung von Metadaten nicht in unrealistische Höhen treibt und trotzdem die für die (Wieder-)Verwendung von Lernobjekten benötigten Informationen bereitstellt. Letztendlich sind jedoch unsere Erfahrungen mit dem flächendeckenden Einsatz des MuSofT-Portal abzuwarten, das die Verwaltung entsprechender Metadaten bereits unterstützt.

4 Planung für das Jahr 2003

Mit der Definition eines LOM-basierten Metadatenformates für Lernobjekte ist eine der Hauptaufgaben von KT2 - wenn nicht die Hauptaufgabe von KT2 - zunächst abgeschlossen. Nunmehr ist es die Aufgabe aller MuSofT-Teilprojekte unter Verwendung des von KT4 realisierten Portals, ihre erstellten Lernobjekte dort mitsamt der entsprechenden Metadaten einzustellen. Hierfür wird von KT2 ein stufenweises Konzept vorgeschlagen, das zunächst die Erfassung von Lerneinheiten und Lernmoduln mit einem minimalen Satz von Attributen erfordert und darauf aufbauend Ergänzungen und Überarbeitungen einplant, so dass bis Projektablauf eine vollständige, detailierte und konsistente Beschreibung aller MuSofT-Lernobjekte vorliegt.

Auf Basis der dabei gemachten Erfahrungen und der mit und mit vorliegenden Materialien ist es dann wiederum die Aufgabe von KT2, im nunmehr letzten Projektjahr

- noch zu schließende Lücken im Lernmaterial schnellstmöglich zu identifizieren
- die Auswahl und Ausdeutung von LOM-Bestandteilen ggf. zu überarbeiten
- und den Erstellern des MuSoft-Portals bei der Entwicklung eines Konzepts zur Versionierung und Variantenbildung von Lernmaterialien zur Seite zu stehen.

Weitere Überlegungen zur Entwicklung des oben angesprochenen Attributvererbungskonzeptes oder eines “vernünftigen” Modularisierungskonzeptes, das die Definition von Lernmoduln mit wohldefinierten Import- und Exportschnittstellen als Startpunkt und Ende von Querverweisen erlauben würde, werden innerhalb der Projektlaufzeit wohl noch in Angriff genommen, aber nicht mehr in die Praxis umgesetzt werden können.

5 Zusammenfassung

Im ersten Jahr der Laufzeit von MuSofT war es die Hauptaufgabe von KT2 gewesen, anhand größerer Fragebogenaktionen einen Überblick über die geplanten Entwicklungen von Lernmaterialien, die dabei zum Einsatz kommenden Sprachen, Methoden, Werkzeuge, Fallbeispiele etc. zu gewinnen. Auf dieser Basis wurden im zweiten Projektjahr - dem Berichtszeitraum - eine Reihe von Festlegungen getroffen, die die Grundlage für die Kombinierbarkeit aller entwickelten Lernmaterialien bilden. Hierzu gehört der einheitliche Einsatz von Java als

Programmiersprache, UML als Modellierungssprache, die Verwendung von Together Control Center als CASE-Tool sowie die schwerpunktmäßige Verwendung von Fallstudien aus dem Logistikbereich.

Dreh- und Angelpunkt für alle Koordinationsaktivitäten der einzelnen Teilprojekte sowie auch der teilprojektübergreifenden vier Koordinationsteams untereinander war aber im Berichtszeitraum die Entwicklung eines LOM-basierten Metadatenkonzeptes. Wie auf den vorgehenden Seiten ausgeführt wurde, beruht dieses Metadatenkonzept nicht nur auf den eigenen Vorarbeiten von KT2, sondern auch auf den Aktivitäten der anderen Teams KT1 und KT3. Zudem bildete es die Basis für die Entwicklung des MuSofT-Portals durch KT4. Es ist nunmehr unsere Aufgabe im letzten Projektjahr, mit Hilfe des MuSofT-Portals alle entwickelten Lernmaterialien mit Metadaten anzureichern und der Allgemeinheit zur Verfügung zu stellen. Die dabei gemachten Erfahrungen werden uns nützliche Hinweise für die Überarbeitung des vorgestellten Metadatenkonzeptes und ggf. auch für die Entwicklung von Verbesserungsvorschlägen am LOM-Standard liefern. Wie weit solche Verbesserungsvorschläge im Rahmen der Projektlaufzeit noch in die Praxis umgesetzt werden können, wird ganz entscheidend davon abhängen, wie einschneidend die dafür notwendigen Änderungen des MuSofT-Portals und wie aufwändig die daraus resultierenden Überarbeitungen hunderter von LO-Beschreibungen sein werden.

So gehen wir etwa nicht davon aus, dass die oben angesprochenen Arbeiten an einem Vererbungskonzept für Metaattribute sowie weitergehende Modularisierungskonzepte noch während der Projektlaufzeit zum Abschluss kommen werden.

Literatur

[Adv02] ADVANCED DISTRIBUTED LEARNING: *SCORM – Sharable Content Object Reference Model*. `http://www.adlnet.org`, 2002.

[DFC+01] DUVAL, ERIL, EDDY FORTE, KRIS CARDINAELS, BART VERHOEVEN, RAFAEL VAN DURM, KOEN HENDRIKX, MARIA WENTLAND FORTE, NORBERT EBEL, MACIEJ MACOWICZ, KEN WARKENTYPE und FLORENCE HAENNI: *The ARIADNE Knowledge Pool System*. Communications of the ACM, 44(5):72–78, Mai 2001.

[IEE02] IEEE LEARNING TECHNOLOGY STANDARDS COMMITTEE, IEEE, 3 Park Avenue New York, NY 10016-5997, USA: *Final Draft of the IEEE Standard for Lerning Objects and Metadata*, Juni 2002. Online erhältlich unter `http://ltsc.ieee.org/wg12/`.

[IMS02] IMS GLOBAL LEARNING CONSORTIUM, INC.: *Product Directory*. `http://www.imsproject.org/direct/getproducts.cfm`, 2002.

Bericht KT4 2001-2002

Klaus Alfert, Ernst-Erich Doberkat, Gregor Engels, Jan Hendrik Hausmann, Corina Kopka, Marc Lohmann, Jörg Pleumann, Annika Wagner

Dieser Bericht fasst die Arbeit des Koordinationsteams 4 (KT4) im Rahmen des Projektes MuSofT (Multimedia in der Softwaretechnik) zusammen. Der Berichtszeitraum umfasst die Jahre 2001 und 2002. Die Planungen für Arbeiten im Jahr 2003 sind ebenfalls aufgeführt.

Inhaltsverzeichnis

1 Einleitung

Das Projekt Multimedia in der Softwaretechnik (MuSofT) ist ein verteiltes Hochschulprojekt zur Erstellung multimedialer Lehrmaterialien und wird im Rahmen des Programms Neue Medien in der Bildung (NMB) gefördert. An MuSofT sind 8 Hochschullehrer aus 7 verschiedenen deutschen Hochschulen beteiligt. Eine solche verteilte Entwicklung macht eine Koordination

zwischen den Projektpartnern unumgänglich, um ein konsistentes und erfolgreiches Projektergebnis zu erreichen. Zu diesem Zweck wurden Koordinationsteams gebildet, die übergreifende Aufgaben im Projekt planen und realisieren sollen. In diesem Bericht werden Aufgaben und Arbeiten des Koordinationsteams 4 in den Jahren 2001 und 2002 zusammengefasst.

2 Aufgabe des Koordinationsteams

Aufgabe des Koordinationsteams 4 (im Folgenden kurz KT4) war laut Projektbeschreibung [DDE02] die Erstellung einer gemeinsamen multimedialen Integrationsplattform. Die erste Aufgabe des KTs war es nun, diesen Begriff schärfer zu fassen. Aufgrund zweier schriftlicher Befragungen der Projektpartner und begleitenden Diskussionen kristallisierte sich heraus, dass die zentrale Aufgabe dieser Plattform die *Sicherung der Nachhaltigkeit* der erzielten Projektergebnisse sein sollte. Konkret bedeutet dies, dass die Lehr-/Lerneinheiten, die im Projekt MuSofT produziert werden, sowohl während der Projektlaufzeit insbesondere aber auch darüber hinaus einen Einsatz in der Hochschullehre erfahren sollen.

Um dies zu erreichen, hat das KT4 die Einrichtung und den Betrieb eines zentralen webbasierten Portales vorgesehen, über das alle Ergebnisse von MuSoft öffentlich zugänglich sind. Zur Verwirklichung des Ziels der Nachhaltigkeit sind dabei eine Reihe von Faktoren zu berücksichtigen: Es muss ein *Nutzungsszenario* entwickelt werden, das darstellt, wer, wie und mit welchen Anforderungen die Projektergebnisse nutzen will und mit welchen Funktionen das Portal dies unterstützt (siehe Abschnitt 3.1). Eine zentrale Fragestellung ist dabei die Definition *technischer Schnittstellen* zu Systemen mit denen die Lerneinheiten erstellt, verwaltet bzw. eingesetzt werden (siehe Abschnitte 3.4 bzw. 3.5). Eine weitere Herausforderung ist es, die Lerneinheiten so zu verwalten, dass ein potentieller Nutzer schnell und bequem *Zugriff* auf die seinen Bedürfnissen entsprechenden Einheiten bekommt. Dies berührt den Bereich der Metadaten für Lerneinheiten, die von KT2 definiert wurden und für die wir eine technische Umsetzung entwickelt haben (siehe Abschnitt 3.2). Hierbei waren vor allem sinnvolle Vorgaben für die Beschreibung der einzelnen *Themengebiete* zu entwickeln, um dem Benutzer eine gezielte Suche nach bzw. eine Navigation über Materialien im Portal zu ermöglichen; Abschnitt 3.3 stellt die Arbeiten zu diesem Thema dar. Bei der Betrachtung der langfristigen Nutzung des Portales muss ein *Betriebskonzept* entwickelt werden, dass den Bestand sichert, gleichzeitig aber auch durch Aktualisierungen und Angebotserweiterungen die Attraktivität und damit die Nutzung sichert. Konzepte hierzu finden sich in 3.6. Diese Konzepte werden realisiert in der Implementierung des Portals, das bereits projektintern eingesetzt wird. Details zur Konstruktion um dem aktuellen Stand der Entwicklung finden sich in Abschnitt 3.7.

3 Bisherige Aktivitäten und Ergebnisse

3.1 Nutzungsszenario

Das Nutzungsszenario des Portals legt die grundlegende Anforderungen an das Portal fest [Ple02]. Die Aufgabe des Portals ist es, wie oben bereits dargestellt, die MuSofT-Lehrmaterialien webbasiert zu distributieren. Nutzer des Portals sollen Lehrende der Softwaretechnik in Universiäten und Fachhochschulen sein, die für ihre Lehrveranstaltungen Materialien suchen. Ne-

ben diesen konsumierenden Nutzern gibt es Autoren, die Materialien in das Portal einstellen. Zu letzteren gehören insbesondere die MuSofT-Projektpartner, eine aktive Beteiligung eines weiteren Personenkreises ist hier jedoch ausdrücklich erwünscht. Zusätzlich fallen Verwaltungsaufgaben innerhalb des Portals an. Wir haben uns daher dafür entschieden, drei Nutzerklassen einzuführen (Konsumenten, Autoren und Administrator). Die Funktionalität des Portals kann nun gestuft durch diese drei Nutzerklassen beschrieben werden.

Konsumenten können Materialien im Portal recherchieren, dabei stehen ihnen verschiedene Recherchemöglichkeiten zur Verfügung: Geordnet nach Themengebieten, nach Autoren oder nach LOM-Hierarchiestufen (siehe Abschnitt 3.2) sowie die direkte Suche. Lerneinheiten können dann als reine Mediendateien heruntergeladen bzw. zusammen mit den Metadaten für den Einsatz in Lehr-/Lernplattformen exportiert werden (siehe Abschnitt 3.4). Fehlerkorrekturen, Verbesserungsvorschläge etc. können über eine Feedbackfunktion an die Autoren einer Lerneinheiten direkt verschickt werden.

Autoren haben Schreibrechte innerhalb des Portals und müssen sich dafür im Portal authentifizieren. Sie können zusätzlich zu den Funktionaliäten der Konsumenten Materialien in das Portal einstellen und dort modifizieren. Es wird dabei gewährleistet, dass jeder Autor nur seine eigenen Materialien modifizieren darf.

Der Administrator hat zusätzliche Verwaltungstätigkeiten zu erledigen. Hierzu zählen insbesondere die Einrichtung und Verwaltung von Nutzern sowie die Pflege der Lernobjekte inklusive des Löschens/Berichtigens von Daten, die fehlerhaft im Portal gespeichert sind.

3.2 Adaption des LOM-Standards

Im Rahmen von KT4 ist nach einer XML basierten Implementation für die von KT2 vorgegebenen Metadaten zur Beschreibung von Lernobjekten gesucht worden. Die Vorgaben von KT2 basieren auf dem LOM Standard. Der IEEE LOM Standard definiert eine Menge von Metadatenelementen zur Beschreibung von Lernobjekten. Damit soll eine konsistente Verwendung von Metadaten als Basis für unterschiedliche Implementierungen festgelegt werden. Eine Beschreibung wie bzw. in welchem Datenformat diese Metadaten zwischen verschiedenen Systemen ausgetauscht werden können wurde vom IEEE jedoch nicht festgelegt. Die IMS (IMS Global Learning Consortium, Inc.) hat hierzu ein XML-Format mit Hilfe von DTDs bzw. XML-Schemas festgelegt.

Ein Grund für die Verwendung der IMS-Spezifikation ist, dass diese bereits von unterschiedlichen Lern-/Lehrumgebungen unterstützt wird. Weiterhin ist die IMS-Spezifikation auch innerhalb des SCORM-Standards, der verschiedene Standards bzw. Spezifikationen aus dem Bereich eLearning zu integrieren versucht, mit geringfügigen Erweiterungen aufgenommen worden, wodurch SCORM-konforme Metadaten erzeugt werden können.

Das vom IMS vorgegebene XML Schema kann nahezu unverändert übernommen werden [Loh02b]. Grundsätzlich stellen die Vorgaben von KT2 nur eine Teilmenge der IMS-Vorgaben dar, d.h. Musoft-Instanzen des IMS Schemas enthalten nicht alle vom IMS vorgegebenen Elemente. Zwei Einschränkungen sind jedoch zu beachten: Erstens kann das von KT2 vorgegebene Attribut „Lernziele“ nicht ohne Änderungen des IMS XML Schemas verwendet werden, da dieses Attribut nicht dem LOM-Standard entstand. Zweitens sind die von KT2 für einige Attribute festgelegten Vorgabewerte nicht Teil der IMS-Spezifikation.

3.3 Taxonomie

Um gezielt nach bestimmten Lehreinheiten zu einem Fachgebiet suchen zu können, ist es notwendig, den Inhalt der Einheit gemäß eines vorgegebenen Kataloges (einer *Taxonomie*) zu klassifizieren. So können Einheiten mit ähnlichen/zusammengehörenden Themen erkannt werden, der Suchraum wird auf sinnvolle Begriffe eingeschränkt und eine Navigation über thematische Zusammenhänge wird ermöglicht.

Als Grundlage für einen solchen Themenkatalog stellt die ACM seit 1964 eine Klassifikation zur Verfügung, die das Fachgebiet Informatik über mehrere Ebenen in einzelne Themengebiete aufteilt. Die aktuelle Version stammt aus dem Jahr 1998 und ist abrufbar unter `www.acm.org/class`. Da die ACM-Klassifikation Standardcharakter besitzt und international Anwendung findet, bietet eine Orientierung daran die größtmöglichen Chancen, eine vom Benutzer akzeptierte Einordnung in Themen festzulegen.

Auf Basis dieser Klassifikation haben wir die für MuSofT relevanten Gebiete und ihre Unterteilungen festgelegt:

- D. Software
- H. Information Systems
- K. Computing Milieux

Details dazu sowie eine erst Zuordnung von MusofT-LEs zu den einzelnen Gebieten finden sich in [Hau02a]. Der für MuSofT relevante Teil der ACM-Klassifikation ist bereits im Portal eingefügt, eine Erweiterung auf andere Felder der ACM-Taxonomie bzw. eventuelle MusofT-interne Themenbezeichnungen ist aufgrund der dynamischen Struktur des Portals jederzeit möglich.

3.4 Exportformat für Lehreinheiten

Das Musoft-Portal soll zur Verwaltung und Verbreitung von Lehreinheiten verwendet werden, die im Rahmen des Projektes Musoft erstellt werden. Um den nachhaltigen Einsatz der entwickelten Lehreinheiten aus Sicht des Portals zu ermöglichen, muss unter anderem sichergestellt werden, dass die darin abgelegten Lehreinheiten in bestehende Lehr-/Lernplattformen integriert werden können. Zu diesem Zweck muss das Musoft-Portal ein entsprechendes (standardisiertes) Exportformat anbieten. Gleichzeitig muss dieses Format den LOM-Standard, bzw. die XML-basierte Implementierung des LOM-Standards, unterstützen, damit auch die existierenden Metadaten zur Beschreibung der Lernobjekte in anderen Lehr-/Lernplattformen verwendet werden können.

Hier kommen grundsätzliche zwei standardisierte Datenaustauschformate in Frage [Loh02a]. Zum einen das IMS Content Packaging und zum anderen die entsprechenden Teile des Sharable Content Object Reference Model (SCORM) von ADL. Die IMS Content Packaging Spezifikation beschreibt Datenstrukturen, um den Austausch von Inhalten zwischen Lehr-/Lernplattformen und auch entsprechenden Autorensystemen zu standardisieren. So können Systeme Pakete mit Lehr-/Lerneinheiten erstellen, die von anderen unabhängig entwickelten Systemen aufgrund der standardisierten Struktur der Pakete eingelesen werden können. IMS

Content Packaging wird bereits von unterschiedlichen Plattformen (z.B. Blackboard, Centra, CourseKeeper) unterstützt. Jedoch ist hier kritisch anzumerken, dass die meisten Plattformen diesen Standard (noch) nicht vollständig bzw. nicht korrekt unterstützen [Boy02, Wil02].

Der SCORM-Standard beinhaltet einen Content Packaging Teil, der im wesentlichen identisch mit der IMS Spezifikation ist, und bietet darüber hinausgehend weitere Möglichkeiten, die Reihenfolge der Darstellung von Inhalten in Abhängigkeit von den Erfahrungen des Studierenden zu beeinflussen. Konkrete Aussagen, in wie weit die unterschiedlichen Plattformen den SCORM-Standard unterstützen, lassen sich noch nicht machen, da entsprechende Untersuchungen nicht verfügbar sind. Jedoch ist aufgrund der hohen Komplexität der Spezifikation davon auszugehen, dass auch dieser Standard nicht korrekt und vollständig implementiert ist.

Aufgrund der Verbreitung und der Beteiligung der wichtigsten Hersteller von eLearning-Plattformen an der Entwicklung der IMS-Spezifikationen, scheint der IMS Content Packaging Standard geeignet zu sein, um mittelfristig die Verwendung von Lernmaterialien in kommerziell erhältlichen und weit verbreiteten Lernplattformen zu ermöglichen. Das Musoft-Portal wird daher einen Export von Lernmaterialien entsprechend der IMS Spezifikationen vorsehen. Die entsprechende Funktionalität wird zur Zeit implementiert und soll Anfang 2003 fertig gestellt werden.

Ein weitere zu beachtende technische Hürde kann das Speicherformat multimedialer Medienobjekte sein. Den IMS Standard berührt dies nicht. Wir haben daher in Abstimmung mit den Projektpartnern eine verbindliche Liste verbreiteter Formate für Animationen, Filme, Dokumente etc. erstellt, die auf eine weitestgehende Verbreitung bei den Lehrenden hin optimiert ist [PA02].

3.5 Import

Um Lerneinheiten in das Portal einzustellen gibt es grundsätzlich zwei Szenarien: Der Benutzer hat eine Menge von Medienobjekten, die eine Lehreinheit bilden. Zur Veröffentlichung dieser Lehreinheit erlaubt ihm das Portal den Upload und die Annotation der Medienobjekte. Mit Hilfe des Portals können die einzelnen Medienobjekte dann zu einer Lehreinheit zusammengesetzt werden. Das andere Szenario ist, dass der Benutzer bereits eine Lernplattform einsetzt, die es ihm erlaubt, Lernobjekte gemäß LOM zu annotieren. In diesem Fall kann eine gemäß des IMS Content Packaging Standards erstellte Datei von der Lernplattform exportiert werden, die dann komplett in das Portal importiert werden kann. Dieser Import soll im Jahr 2003 entwickelt werden.

Da aber Tools zur Erstellung von IMS Content Packages zur Zeit noch nicht weit verbreitet sind, soll den Projektpartnern das Einstellen von Medienobjekten erleichtert werden, indem im Laufe des nächsten Jahres ein Offline-Editor zur Zusammenstellung von Lehreinheiten bereitgestellt wird. Mit diesem Offline-Editor können einzelne Lernobjekte gruppiert und mit Metadaten versehen werden. Nach dem Fertigstellen einer Lehreinheit kann mit Hilfe dieses Editors ein IMS Content Package erstellt werden, welches dann vom MuSofT-Portal importiert werden kann. Der Offline-Editor soll in wichtigen Teilen als Diplomarbeit realisiert werden.

3.6 Betriebskonzept

Um mit dem MuSoft-Portal nachhaltig die im Projekt erstellten Lehreinheiten verbreiten zu können ist ein Betriebskonzept erforderlich, das den erfolgreichen Betrieb auch nach Ende der Projektlaufzeit (12/2003) sicherstellt. Dies betrifft zum einen eine Sicherung des Bestandes. Hier müssen z.B. Pläne zum Betrieb und der Betreuung des Webservers festgelegt werden. Detaillierte Rahmenbedingungen, die innerhalb eines derartigen Betriebskonzeptes beachtet werden müssen, finden sich in [Hau02b]. Die Fertigstellung dieses Betriebskonzeptes soll im Jahr 2003 erfolgen.

Daneben gilt es aber auch, das Portal attraktiv zu machen und zu erhalten. Es gibt Anregungen zum Marketing für den Server, die im nächsten Jahr konkretisiert werden müssen. Auch das von KT4 entwickelte neue Layout des Portals ist ein Schritt zur Erreichung dieser Attraktivität. Die Aktualisierung, Pflege und Ausweitung der angebotenen Lehreinheiten haben eine zentrale Bedeutung für den nachhaltigen Erfolg von MuSoft. KT4 entwicklt hierzu Konzepte und richtet die Bedienung des Portals darauf aus, diese Aufgaben möglichst einfach und bequem durchführen zu können. Ein User Guide wird die notwendigen Hilfestellungen dazu geben.

3.7 Technische Realisierung des Portals

Das webbasierte MuSofT-Portal (`http://www.musoft.org`) ermöglicht die öffentliche Verbreitung der im Rahmen von MuSofT erstellten Lehreinheiten und den dazugehörigen Werkzeugen. In diesem Abschnitt beschreiben wir die technische Realisierung des Portals und den Einsatz der Metadaten.

3.7.1 Technische Realisierung

Das MuSofT-Portal kann man als eine spezielle Variante eines Content-Management-Systems (CMS) auffassen, das als Inhalte die multimedialen Lernobjekte zusammen mit Metadaten für die Recherche verwaltet. Kommerziell verfügbare CMS haben allerdings unsere Anforderungen nicht erfüllt. Als besonders problematisch haben sich dabei vier Punkte herausgestellt:

1. Klassischerweise unterscheiden CMS zwischen einer Entwicklungssicht mit und einer Präsentationssicht ohne (ausgefeilte) Recherchemöglichkeiten. Dies widerspricht aber der Arbeitsweise im MuSofT-Portal, da es dort nur eine (Präsentations-) Sicht gibt, in der insbesondere recherchiert werden soll.

2. Wir benötigen nicht nur eine fest vom System vorgegebene Menge von Metaattributen -wie dies bei CMS oft der Fall ist-, sondern zusätzlich strukturierte Metaattribute (z.B. für die Klassifikationshierarchie).

3. Da wir erwarten, dass der Satz an Metaattributen und ihrer Wertemengen sich aufgrund unserer künftigen Erfahrungen mit dem Portal ändern wird, brauchen wir eine leichte Änderbarkeit der Metaattribute und ihrer Ausprägungen im laufendem Betrieb.

4. Schlussendlich sind für eine nutzerfreundliche Bedienung spezielle Suchanfragen entlang der Hierarchie der Lernobjekte und der Hierarchie des Klassifikationsschemas nötig.

Wir haben uns daher für eine Eigenentwicklung auf der Basis einer existierenden objektorientierten Datenbanklösung entschieden, die es uns ermöglicht, ein flexibles webbasiertes Portal zu erstellen.

Serverseitig basiert das MuSofT-Portal auf dem *Infolayer-System* [HP02], das als Servlet realisiert wurde. Das Infolayer-System ist eine objektorientierte Datenbank, deren Schema mit durch OCL annotierten UML-Klassendiagrammen spezifiert wird. Als Abfragesprache wird ebenfalls OCL verwendet. Die Daten werden als XML-Dateien abgespeichert. Eine Standardweboberfläche zur Navigation durch das Schema und durch die vorhandenen Objektinstanzen wird vom Infolayer-System automatisch zur Verfügung gestellt und kann von jedem (neueren) HTML-Browser aus bedient werden. Diese Oberfläche kann sukzessive durch Schablonen an eigene Anforderungen angepasst werden, so dass Entwicklungsarbeiten sehr schnell zu bereits produktiven Prototypen führen. Diese Eigenschaften erlauben eine einfache Anpassung der Datenbank und des Portals, wenn sich die Metadaten aufgrund von Erfahrungen beim Einsatz des MuSofT-Portals ändern.

3.7.2 Verwendung von Metadaten

Das MuSofT-Portal dient zur Archivierung von Lernobjekten, die innerhalb von MuSofT erstellt werden. Die wichtigsten Aktivitäten, die mit dem Portal ausgeführt werden können, sind das Einfügen, Aktualisieren und Suchen von Lernobjekten. Beim Einfügen eines neuen Lernobjekts oder beim Modifizieren eines bestehenden Lernobjekts sind die bereits erwähnten Vorgaben für Metadaten zu berücksichtigen, damit innerhalb des MuSofT-Portals eine einheitliche Beschreibung der Lernobjekte erfolgt. Die Oberfläche des Systems unterstützt den Benutzer dabei, in dem sie Eingabefelder für die erforderlichen Attribute anbietet und für Attribute mit fester Wertemenge eine Auswahlbox verwendet, so dass dort keine ungültigen Werte eingegeben werden können. Abbildung 1 zeigt einen Ausschnitt der Benutzeroberfläche des MuSofT-Portals zur Beschreibung eines Lernobjekts.

Neben dem reinen Hochladen von Lernobjekten unterstützt das MuSofT-Portal die vom LOM-Standard vorgeschriebene Hierarchisierung von Lernobjekten. So kann z.B. eine neue Lehreinheit aus verschiedenen Lernmodulen, die zuvor in dem MuSofT-Portal erstellt wurden, zusammengesetzt werden.

Da Metadaten für Lernobjekte immer einen sehr subjektiven Charakter haben, kann es schwierig sein, passende Lernobjekte zu finden, wenn man sich ausschließlich auf Freitexteingaben für Stichworte und inhaltliche Beschreibungen verlässt. Wir verwenden daher für die inhaltliche Beschreibung zusätzlich eine festgelegte Taxonomie (siehe Abschnitt 3.3. Bei Bedarf kann dieses Klassifikationsschema sowohl um zusätzliche neue Themenbereiche als auch um verfeinerte Klassifikationen erweitert werden. Ebenso können bei Bedarf weitere unabhängige Klassifikationssysteme zur Verfügung gestellt werden. Lernobjekte können mit einem oder mehreren Einträgen aus dem Klassifikationsschema versehen werden, so wie dies bei der Klassifikation von Zeitschriftenartikeln üblich ist. Wir erwarten, dass die inhaltliche Recherche wesentlich über das Klassifikationsschema stattfinden wird.

4 Planung für das nächste Jahr

Da die Attraktivität des MusofT-Portales ein Hauptziel von KT4 ist, wird das bestehende Portals beständig aufgrund von Erfahrungen und Vorschlägen der Projektbeteiligten verbessert. Aktuelle Vorschläge sind etwa eine Gruppierung und Suche nach Medientypen, wie z.B. Flash oder Powerpoint. Eine solche Abrundung des Portalprogrammes geht einher mit einem Betriebskonzept für das Portal.

Die Realisierung eines Offline-Editors zur Erstellung von IMS Content Packages (im Rahmen einer Diplomarbeit) und die Erweiterung des MuSofT-Portals um einen entsprechenden Import soll die Verwendung des MuSofT-Portals noch weiter vereinfachen und damit die Breite des Angebot steigern. Der Export in dieses Format wird im ersten Quartal 2003 fertiggestellt.

5 Zusammenfassung

Die Aufgabe der Erstellung einer gemeinsamen multimedialen Plattform für MuSofT hat KT4 dahingehend gelöst, ein webbasiertes Portal zu entwickeln, mit dem die Lerneinheiten, die in MuSofT entwickelt werden, zentral gesammelt und der Öffentlichkeit zugänglich gemacht werden können. Ausgestattet mit komfortablen Recherchemöglichkeiten sowie (demnächst) Im- und Export auf der Basis von etablierten Metadatenstandards für Lernobjekte, bietet das MuSofT-Portal einen effektiven Weg zur nachhaltigen Distribution der MuSofT-Lerneinheiten.

Literatur

[Boy02] BOYLE, E.: *CETIS EC-SIG Content Exchange Report*, Januar 2002. Erhältlich auf `http://www.cetis.ac.uk/groups/20010809144711/FR20020301142909`.

[DDE02] DISSMANN, STEFAN, ERNST-ERICH DOBERKAT und GREGOR ENGELS: *Projektantrag MuSofT – Multimedia in der SoftwareTechnik*. Unveröffentlichter Projektantrag für das Programm *Neue Medien in der Bildung* des BMBF, Februar 2002.

[Hau02a] HAUSMANN, JAN HENDRIK: *MuSofT - Klassifikation der Lehr/Lerneinheiten - SE-Taxonomie*. Erhältlich auf `http://www.musoft.org`, erscheint als MuSofT-Bericht, 2002.

[Hau02b] HAUSMANN, JAN HENDRIK: *Nachhaltigkeit der Ergebnisse des MuSofT-Projektes*. Erhältlich auf `http://www.musoft.org`, erscheint als MuSofT-Bericht, 2002.

[HP02] HAUSTEIN, STEFAN und JÖRG PLEUMANN: *Is Participation in the Semantic Web Too Difficult?* In: HORROCKS, I. und J. HENDLER (Herausgeber): *The Semantic Web - First International Semantic Web Conference*, Band 2342 der Reihe *LNCS*, Heidelberg, 2002. Springer.

[Loh02a] LOHMANN, MARC: *MuSofT - Exportformat für Lehreinheiten*. Erhältlich auf `http://www.musoft.org`, erscheint als MuSofT-Bericht, 2002.

[Loh02b] LOHMANN, MARC: *Vergleich Metadaten KT2 - IMS Learning Resource Meta-Data XML Binding*. Erhältlich auf `http://www.musoft.org`, erscheint als MuSofT-Bericht, 2002.

[PA02] PLEUMANN, JÖRG und KLAUS ALFERT: *Unterstütze Formate für MuSofT*. Erhältlich auf `http://www.musoft.org`, September 2002.

[Ple02] PLEUMANN, JÖRG: *Anwendungsfälle für das MuSofT-Portal*. Erhältlich auf `http://www.musoft.org`, erscheint als MuSofT-Bericht, Juli 2002.

[Wil02] WILSON, S.: *Content Packaging interoperability tests reveal room for improvement*, April 2002. Erhältlich auf `http://www.cetis.ac.uk/content/20020307103412`.

Abbildung 1: Die Metadaten eines beispielhaften Lernobjektes, wie es sich im MuSofT-Portal präsentiert

Der Jahresbericht 2002 des Koordinationsteams 5: Videoproduktion

Klaus Alfert

Inhaltsverzeichnis

1 Einleitung

Ein wesentlicher Teil der finanziellen Ausstattung von MuSofT ist gedacht für die Erstellung von Medien, insbesondere für Videoproduktionen wie sie etwa für das Teilprojekt 1.1 (Videogestützte Anforderungsanalyse) vorgesehen sind. Um diese Summe nicht nur in einem einzigen Teilprojekt einzusetzen, sondern möglichst vielen anderen Teilprojekten ebenfalls die Möglichkeit zu geben, entsprechende Medien zu produzieren, ist das Koordinationsteam 5 (Medienproduktion, kurz: KT5) mit dem dritten Plenum Anfang März 2002 ins Leben gerufen worden. Die Aufgabe des KT5 ist es, Medienproduktionen, die über Teilprojektgrenzen hinweg möglich ist, zu koordinieren.

2 Aufgabe des Koordinationsteams

Aufgabe des KT5 ist es, die Planung, Konzeption und Durchführung von Medienproduktionen zu koordinieren, die teilprojektübergreifend eingesetzt werden können. Wesentlich ist es dabei, den Bedarf der einzelnen Teilprojekte herauszuarbeiten und Kristallisationspunkte zu finden, an denen die Teilprojekte gut kooperieren können. Auf der Basis dieser Überlegungen

können die Teilprojekte einzelnen oder gemeinsam ihren Mittelbedarf für die Medienproduktion bei der Projektleitung anmelden und abrechnen. Die Mittel für diese Medienproduktionen stehen hälftig in 2002 und 2003 zur Verfügung.

3 Bisherige Aktivitäten und Ergebnisse

Die bisherigen Aktivitäten des KT5 in 2002 lassen sich in die Bereiche Bedarfsermittlung, Planung und Medienproduktionen teilen.

3.1 Bedarfsermittlung

Zur Ermittlung des Bedarfes an Medienproduktionen haben zwei Treffen des KT5 stattgefunden, am 20. April 2002 in Dortmund und am 7. Juni 2002 in Paderborn.

Das erste Treffen beschäftigte sich im wesentlichen mit Fragen zu Videoproduktionen an sich. Als unterschiedliche Ausrichtungen von Videos wurden drei Kategorieren herausgearbeitet:

Melodram: zeigt Fehlverhalten und die daraus resultierenden Konsequenzen,

Instruktion: zeigt beispielhaftes Vorgehen,

Aufgabenstellung: zeigt eine Situationen, die die Basis für eine (Übungs-) Aufgabe darstellt.

Die Kategorie *Melodram* ist nur schlecht einsetzbar, weil der Aufwand an schauspielerischer Konzeption und Leistung als entschieden zu hoch angesehen wird. *Instruktionsvideos* eignen sich gut für Werkzeugschulungen und scheiden damit zwar als teilprojektübergreifende Medienproduktionen aus, sind aber für einzelnen Teilvorhaben sinnvoll. Die dritte Kategorie *Aufgabenstellung* ist daher die favorisierte Variante der Medienproduktion, die teilprojektübergreifend am sinnvollsten eingesetzt werden kann.

Als interessantes Anwendungsgebiet für die Medienproduktion hat sich das die Logistik herausgestellt, da hier viele für MuSofT interessante Probleme diskutiert werden können. Zusätzlich sind bereits einige Vorarbeiten in den Teilvorhaben verfügbar (z.B. Hochregallagersteuerung in TP 1.3, Algorithmen für Speditionen in TP 2.3). Im zweiten Treffen in Paderborn wurde daher der Themenkomplex Logistik mit dem Schwerpunkt Hochregallager intensiv diskutiert. Produktvideos von Mannesmann-Demag für Hochregallager haben einen ersten Eindruck gegeben, wie ein derartiges technisches System präsentiert werden kann. Dabei wurde deutlich, dass diese Produktvideos zwar für einzelne Aspekte etwa zur Modellierung des Systems direkt sinnvoll einsetzbar sind, aber für viele weitere Fragestellungen sich zu sehr auf die maschinenbautechnischen Fragestellungen konzentrieren und softwaretechnikrelevante Aspekte nicht thematisieren. Daher wurde beschlossen, dass eine eigene Videoproduktion angestrebt werden soll.

3.2 Planung

Die teilprojektübergreifende Medienproduktion zum Thema Logistik spaltet sich in verschiedene Teilvorhaben auf. Für die Teilprojekte 1.1 und 1.3 werden Abläufe aus dem Lagerwesen

und der Bestellabwicklung betrachtet. Für weitere Teilprojekte (z.B. 1.2, 2.1, 3.3) ist der gewünschte Inhalt noch unklar, wird sich aber auch im Umfeld des Lagerwesen und der Bestellabwicklung bewegen. Für diese beiden eng miteinanderverwandten Bereiche soll ein Video produziert werden. Für das Teilprojekt 2.3 werden dagegen explizit Problemstellungen aus dem Speditionswesen betrachtet, die damit unabhängig von den Bereichen Lagerwesen und Bestellabwicklung sind. Daher wird hier auch keine Videoproduktion realisiert.

Im Vorfeld des zweiten KT-Treffens in Paderborn wurden bereits erste Kontakte zur Logistiksparte von Bertelsmann, der avarto services GmbH in Gütersloh, geknüpft und ein Besuch des Hochregallagers in Gütersloh hat stattgefunden. Bei der weitergehenden Planung wurde der dringende Bedarf an Unterstützung und Einweisung in das Drehbuchschreiben sowie weiterer Vorarbeiten zur Vorbereitung eines Videodrehs durch ein professionelles Team festegestellt. Daher wurde ab Mitte November die Industrie-Designerin Bettina Neu von Siemens Business Services als Beratin hinzugezogen. Zur Vorbereitung der Drehbucherstellung hat dann ein weiterer Besuch unter der Leitung von Herrn Bätge (Leitung Wareneingang) bei avarto in Gütersloh stattgefunden, bei dem eine Menge an rohen Bild-, Video- und Tonmaterial entstanden ist, dass für Skizzen innerhalb eines Drehbuches und zur Festlegung von Drehorten benutzt werden kann. Desweiteren hat Herr Bätge zugesagt, dass MuSofT auf das firmeneigene Bild- und Videomaterial von avarto selbst zugreifen darf, so dass auf Weise der Bedarf an neuen Videoproduktionen gemindert werden kann.

3.3 Medienproduktionen

Neben der Vorbereitung des Videodrehs zum Lagerwesen und der Bestellabwicklung sind bereits zwei weitere Medienproduktionen im Jahre 2002 abgeschlossen bzw. angelaufen.

Als erstes ist das Instruktionsvideo zur Benutzung des Versions- und Konfigurationsmanagementwerkzeuges CVS zu nennen, dass im Rahmen des Teilprojektes 3.4 entstanden ist. Es zeigt die Benutzung eines CVS-Clients und illustriert durch Animationen die Vorgänge im CVS-Repository, die durch den Client initiiert werden.

Die zweite Produktion animiert Algorithmen und Datenstrukturen für das Teilprojekt 2.3 mit dem Anwendungsgebiet Speditionswesen. Neben der Vorführung vorprogrammierter Animationen und Algorithmen wird eine Programmierschnittstelle angeboten, so dass auch Studierenden die Animationen nutzen können, um ihre eigene Algorithmen visualisieren zu können. Dieses Animationsprojekt befindet sich zur Zeit in Arbeit und soll im ersten Quartal 2003 abgeschlossen sein.

4 Aktivitäten im nächsten Jahr

Im Jahr 2003 werden im wesentlichen die Aktivitäten zum Videodreh bei avarto forgesetzt. Als erstes finden noch im Januar zwei Workshops zu den Themen *Drehbuch schreiben* und *Gestaltung* statt. Beide Workshops dienen dazu, die mediale Konzeption der einzelnen Teilprojekte zu überarbeiten bzw. in eine Form zu bringen, die es erlaubt, mit externen Medienproduzenten effektiv zusammen zu arbeiten. Beide Workshops werden von Bettina Neu geleitet. Speziell für den Drehbuchworkshop soll das Material von avarto gesichtet werden, damit als Neben-

effekt zugleich der Bedarf an einer MuSofT-eigenen Videoproduktion genauer eingeschätzt werden kann.

Sofern sich der Bedarf der MuSofT-Videoproduktion bewahrheitet, ist als nächste wesentliche Aktivität die Planung und Durchführung des Videodrehs. Sie sollte innerhalb des ersten Halbjahres 2003 abgeschlossen sein, damit die Materialien noch angemessen in die Entwicklung der Lerneinheiten einfliessen können.

Der Jahresbericht 2002 des Koordinationsteams 6: Nachhaltigkeit

Klaus Alfert

Inhaltsverzeichnis

1 Einleitung

Das Koordinationsteam 6 (Nachhaltigkeit) wurde auf dem dritten Plenum Anfang März 2002 ins Leben gerufen. Die Aufgabe des KT ist es, die Grundlagen für die nachhaltige Wirkung des MuSofT-Projektes zur Verfügung zu stellen.

Die Nachhaltigkeit von MuSofT wird durch drei verschiedene Faktoren wesentlich beeinflußt. Zum einen müssen produzierten Materialien eine Qualität besitzen, die eine Nutzbarkeit über das Projektende und über die Projektmitglieder hinaus überhaupt erst ermöglichen. Dies liegt primär in der Verantwortung der jeweiligen Teilprojektleiter. Zum zweiten setzt die Nutzung von Materialien, die fremde Autoren erstellt haben – unabhängig davon ob sie zu MuSofT gehören oder nicht – voraus, dass die Nutzungsrechte von den Rechteinhabern eingeholt wurden und somit einer Nutzung der fremden Materialien nichts entgegensteht. Der dritte Faktor betrifft die Wartung und Weiterentwicklung der Materialien nach dem Ende von MuSofT mit Ende des Jahres 2003. Hier ist notwendig sicherzustellen, dass die Finanzierung und Personalsituation den Aufgaben entsprechend gesichert ist. Dies kann z.B. durch Nachfolgeprojekte gewährleistet werden.

Erst wenn diese Faktoren zufriedenstellend beachtet worden sind, ist ein nachhaltiger Einsatz der MuSofT-Materialien sowohl in der Hochschullehre als auch in der kommerziellen Weiterbildung sinnvoll und möglich. Die vordringliche Aufgabe und der Auftrag des KT6 beziehen sich auf die Regelung der rechtlichen Rahmenbedingungen für MuSofT, da diese als das wesentliche teilprojektübergreifende Problem angesehen werden.

2 Bisherige Aktivitäten und Ergebnisse

Vorbereitend und zur Gründung des KT6 führend war der Besuch von Klaus Alfert auf dem Workshop „Recht einfach. Rechtemanagement für Multimediaprojekte“, der vom Kompetenznetzwerk Universitätsverbund MultiMedia Nordrhein-Westfalen (UVM) im Auftrages des Projektträger veranstaltet [UVM01] wurde. Die für diese Veranstaltung erstellte Broschüre zum Rechtemanagement [Ved01] wurde für alle Projektpartner beschafft und zum Jahreswechsel 2001/2002 verteilt. Auf dem dritten Plenum hat Frau Dusch vom UVM einen Vortrag zum Rechtemanagement gehalten [Dus02]. Aus den nachfolgenden Diskussionen hat sich die Notwendigkeit ergeben, das Rechtemanagement in MuSofT explizit zu betrachten, dies ist die Aufgabe des daraufhin gegründeten KT6.

Bei der Betrachtung der Rechteproblematik für MuSofT stellt sich heraus, dass der Kooperationsvertrag zwischen den Projektpartnern einige wesentliche Fragen nicht regelt bzw. dem Geist des Projektantrages widersprechen, z.B.:

- Die Nutzung der Materialien nach Ende des Projektes ist nicht geregelt, insbesondere die kommerzielle Nutzung.
- Die Überlassung der Materialien auf Quellcode-Ebene inkl. dem Änderungsrecht (also als *Open Content*, dies entspricht im Wesentlichen dem Begriff *Open Source* aus dem Projektantrag), wird im Kooperationsvertrag im Gegensatz zum Projektantrag nicht zugelassen.

Daher ist es notwendig, den alten Kooperationsvertrag, der der Mustervertrag für Projektkooperationen des BMBF ist, in angemessener Weise für die Belange von MuSofT zu überarbeiten. Auf der Basis eines überarbeiteten Vertrages kann dann die weitere Gestaltung der rechtlichen Situation in MuSofT erfolgen.

Da die Vertragsausarbeitung juristische Kenntnisse voraussetzt, ist dies keine Tätigkeit, die ein Projektpartner selbst übernehmen kann. Daher wurde ein Auftrag an die Juristen des UVM erteilt, einen Vorschlag für einen Kooperationsvertrag zu entwickeln. Ein zweiter Auftrag an den UVM sieht vor, dass die Justiare der Partnerhochschulen sensibilisiert und geschult werden sollen, die Rechteproblematik in Multimediaprojekten zu erkennen und zu bearbeiten. Die Einbindung der Justiare ist notwendig, da die Hochschulen auch immer Rechteinhaber werden, wenn Multimediaproduktionen an Hochschulen durchgeführt werden. Für die weitere Verwertung ist daher immer eine Kooperation mit den Hochschulen notwendig und erfordert daher entsprechende Kenntnisse in den Verwaltungen.

3 Planung für das nächste Jahr

Ein erster Entwurf des Kooperationsvertrages soll zum fünften MuSofT-Plenum Anfang März 2003 vorliegen. Dieser Entwurf wird die Basis für weitere Arbeiten legen, die zum Abschluss eines neuen Kooperationsvertragen führen sollen. Auf dieser Basis kann dann das Rechtemanagement für MuSofT organisiert werden, dies ist insbesondere für die geplanten Videoproduktionen und den Einsatz von Firmenvideos (z.B. Bertelsmann/Avarto, DaimlerChrysler u.a.) wesentlich.

Neben der Behandlung des Rechteproblematik wird in 2003 die Sicherung der Ergebnisse von MuSofT über das Projektende hinaus von wesentlicher Bedeutung sein. Dafür werden Möglichkeiten für Nachfolgeprojekte recherchiert und gegebenenfalls aktiv mit Projektanträgen angegangen.

Literatur

[Dus02] DUSCH, CHRISTIANE: *Rechtemanagement in Multimediaprojekten*. Vortrag auf dem 3. Plenum des Projektes MuSofT, März 2002.

[UVM01] KOMPETENZNETZWERK UNIVERSITÄTSVERBUND MULTIMEDIA NRW (UVM): *Tagungband Recht einfach – Rechtemanagement in Multimediaprojekten für Beteiligte am Bundesprogramm „Neue Medien in er Bildung“*, November 2001.

[Ved01] VEDDERN, MICHAEL: *Update – Ratgeber. Multimediarecht für die Hochschulpraxis*. Ministerium für Wissenschaft und Forschung des Landes Nordrhein-Westfalen, November 2001.

/99/ T. Bühren, M. Cakir, E. Can, A. Dombrowski, G. Geist, V. Gruhn, M. Gürgrn, S. Handschumacher, M. Heller, C. Lüer, D. Peters, G. Vollmer, U. Wellen, J. von Werne
Endbericht der Projektgruppe eCCo (PG 315)
Electronic Commerce in der Versicherungsbranche
Beispielhafte Unterstützung verteilter Geschäftsprozesse
Februar 1999

/100/ A. Fronk, J. Pleumann,
Der DoDL-Compiler
August 1999

/101/ K. Alfert, E.-E. Doberkat, C. Kopka
Towards Constructing a Flexible Multimedia Environment for Teaching the History of Art
September 1999

/102/ E.-E. Doberkat
An Note on a Categorial Semantics for ER-Models
November 1999

/103/ Christoph Begall, Matthias Dorka, Adil Kassabi, Wilhelm Leibel, Sebastian Linz, Sascha Lüdecke, Andreas Schröder, Jens Schröder, Sebastian Schütte, Thomas Sparenberg, Christian Stücke, Martin Uebing, Klaus Alfert, Alexander Fronk, Ernst-Erich Doberkat
Abschlußbericht der Projektgruppe PG-HEU (326)
Oktober 1999

/104/ Corina Kopka
Ein Vorgehensmodell für die Entwicklung multimedialer Lernsysteme
März 2000

/105/ Stefan Austen, Wahid Bashirazad, Matthais Book, Traugott Dittmann, Bernhard Flechtker, Hassan Ghane, Stefan Göbel, Chris Haase, Christian Leifkes, Martin Mocker, Stefan Puls, Carsten Seidel, Volker Gruhn, Lothar Schöpe, Ursula Wellen
Zwischenbericht der Projektgruppe IPSI
April 2000

/106/ Ernst-Erich Doberkat
Die Hofzwerge — Ein kurzes Tutorium zur objektorientierten Modellierung
September 2000

/107/ Leonid Abelev, Carsten Brockmann, Pedro Calado, Michael Damatow, Michael Heinrichs, Oliver Kowalke, Daniel Link, Holger Lümkemann, Thorsten Niedzwetzki, Martin Otten, Michael Rittinghaus, Gerrit Rothmaier
Volker Gruhn, Ursula Wellen
Zwischenbericht der Projektgruppe Palermo
November 2000

/108/ Stefan Austen, Wahid Bashirazad, Matthais Book, Traugott Dittmann, Bernhard Flechtker, Hassan Ghane, Stefan Göbel, Chris Haase, Christian Leifkes, Martin Mocker, Stefan Puls, Carsten Seidel, Volker Gruhn, Lothar Schöpe, Ursula Wellen
Endbericht der Projektgruppe IPSI
Februar 2001

/109/ Leonid Abelev, Carsten Brockmann, Pedro Calado, Michael Damatow, Michael Heinrichs, Oliver Kowalke, Daniel Link, Holger Lümkemann, Thorsten Niedzwetzki, Martin Otten, Michael Rittinghaus, Gerrit Rothmaier
Volker Gruhn, Ursula Wellen
Zwischenbericht der Projektgruppe Palermo
Februar 2001

/110/ Eugenio G. Omodeo, Ernst-Erich Doberkat
Algebraic semantics of ER-models from the standpoint of map calculus.
Part I: Static view
März 2001

/111/ Ernst-Erich Doberkat
An Architecture for a System of Mobile Agents
März 2001

/112/ Corina Kopka, Ursula Wellen
Development of a Software Production Process Model for Multimedia CAL Systems by Applying Process Landscaping
April 2001

/113/ Ernst-Erich Doberkat
The Converse of a Probabilistic Relation
Oktober 2002

/114/ Ernst-Erich Doberkat, Eugenio G. Omodeo
Algebraic semantics of ER-models in the context of the calculus of relations.
Part II: Dynamic view
Juli 2001

/115/ Volker Gruhn, Lothar Schöpe (Eds.)
Unterstützung von verteilten Softwareentwicklungsprozessen durch integrierte Planungs-, Workflow- und Groupware-Ansätze
September 2001

/116/ Ernst-Erich Doberkat
The Demonic Product of Probabilistic Relations
September 2001

/117/ Klaus Alfert, Alexander Fronk, Frank Engelen
Experiences in 3-Dimensional Visualization of Java Class Relations
September 2001

/118/ Ernst-Erich Doberkat
The Hierarchical Refinement of Probabilistic Relations
November 2001

/119/ Markus Alvermann, Martin Ernst, Tamara Flatt, Urs Helmig, Thorsten Langer, Ingo Röpling,
Clemens Schäfer, Nikolai Schreier, Olga Shtern
Ursula Wellen, Dirk Peters, Volker Gruhn
Project Group Chairware Intermediate Report
November 2001

/120/ Volker Gruhn, Ursula Wellen
Autonomies in a Software Process Landscape
Januar 2002

/121/ Ernst-Erich Doberkat, Gregor Engels (Hrsg.)
Ergebnisbericht des Jahres 2001
des Projektes “MuSofT – Multimedia in der SoftwareTechnik”
Februrar 2002

/122/ Ernst-Erich Doberkat, Gregor Engels, Jan Hendrik Hausmann, Mark Lohmann, Christof Veltmann
Anforderungen an eine eLearning-Plattform — Innovation und Integration —
April 2002

/123/ Ernst-Erich Doberkat
Pipes and Filters: Modelling a Software Architecture Through Relations
Juni 2002

/124/ Volker Gruhn, Lothar Schöpe
Integration von Legacy-Systemen mit Eletronic Commerce Anwendungen
Juni 2002

/125/ Ernst-Erich Doberkat
A Remark on A. Edalat’s Paper *Semi-Pullbacks and Bisimulations in Categories of Markov-Processes*
Juli 2002

/126/ Alexander Fronk
Towards the algebraic analysis of hyperlink structures
August 2002

/127/ Markus Alvermann, Martin Ernst, Tamara Flatt, Urs Helmig, Thorsten Langer
Ingo Röpling, Clemens Schäfer, Nikolai Schreier, Olga Shtern
Ursula Wellen, Dirk Peters, Volker Gruhn
Project Group Chairware Final Report
August 2002

/128/ Timo Albert, Zahir Amiri, Dino Hasanbegovic, Narcisse Kemogne Kamdem,
Christian Kotthoff, Dennis Müller, Matthias Niggemeier, Andre Pavlenko, Stefan Pinschke,
Alireza Salemi, Bastian Schlich, Alexander Schmitz
Volker Gruhn, Lothar Schöpe, Ursula Wellen
Zwischenbericht der Projektgruppe Com42Bill (PG 411)
September 2002

/129/ Alexander Fronk
An Approach to Algebraic Semantics of Object-Oriented Languages
Oktober 2002

/130/ Ernst-Erich Doberkat
Semi-Pullbacks and Bisimulations in Categories of Stochastic Relations
November 2002

/131 Yalda Ariana, Oliver Effner, Marcel Gleis, Martin Krzysiak,
Jens Lauert, Thomas Louis, Carsten Röttgers, Kai Schwaighofer,
Martin Testrot, Uwe Ulrich, Xingguang Yuan
Prof. Dr. Volker Gruhn, Sami Beydeda
Endbericht der PG nightshift:
Dokumentation der verteilten Geschäftsprozesse im FBI und Umsetzung von Teilen dieser Prozesse im Rahmen eines FBI-Intranets basierend auf WAP- und Java-Technologie
Februar 2003

/132/ Ernst-Erich Doberkat, Eugenio G. Omodeo
ER Modelling from First Relational Principles
Februar 2003

/133/ Klaus Alfert, Ernst-Erich Doberkat, Gregor Engels (Hrsg.)
Ergebnisbericht des Jahres 2002 des Projektes “MuSofT – Multimedia in der SoftwareTechnik”
März 2003