Konzeption und Realisierung einer eTesting - Umgebung für multimediale Lehr- und Lernsysteme

Stefan H. Kleemann
161531

DV-Labor
FH-Aachen

Betreuer:
Prof. Dr.rer.nat. Wilhelm Hanrath
Prof. Dr.rer.nat. Gisela Engeln-Müllges

Dezember 2005

Für Linus und Diana

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig im Rahmen der an der Fachhochschule Aachen üblichen Betreuung angefertigt habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Aachen, den 19. Dezember 2005 Stefan H. Kleemann

Kurzfassung

Das DV-Labor der Fachhochschule Aachen entwickelte in Zusammenarbeit mit der Freien Universität Berlin, der Fachhochschule Münster und der Fachhochschule Südwestfalen, Abt. Soest, das Lehr- und Lernsystem NUMAS (www.numas.de), ein modernes eLearning - System für die Numerische Mathematik und Statistik. Das eigens entwickelte didaktische Konzept von NUMAS sieht eine Aufteilung des Lehrstoffes in Lernfelder und Lernobjekte vor, dabei stellen die Lernobjekte Unterkapitel der Lernfelder da. Die meisten Lernobjekte schließen derzeit mit Übungsaufgaben, den Testfragen zur eigenen Lernkontrolle ab. Diese Testfragen sind nur unter großem Aufwand änder- und erweiterbar, das bedeutet, dass die Autoren keine Möglichkeit haben Testfragen schnell und unkompliziert in das System einzufügen oder vorhandene Testfragen zu ändern.

Eine Erweiterung des eLearning - Systems um ein Übungs- und Prüfungsmodul schafft die Möglichkeiten, neue Übungsaufgaben in das System einfach zu integrieren, diese Übungsaufgaben in geeigneter Weise als Prüfungen zusammenzustellen und die Prüfungen online durchzuführen. Das eTesting - System (Übungs- und Prüfungsmodul; im Internet verwendeter Begriff in Analogie zu eLearning) wurde extra für die Belange von NUMAS entwickelt, es lässt sich aber auch jederzeit problemlos in andere eLearning - Systeme integrieren, sowie der Betrieb als eigenständige Internet - Anwendung ist möglich.

Die Aufgaben mit den dazugehörigen Lösungen werden durch das eTesting - System verwaltet. Im Übungsmodus werden die Aufgaben auf Anforderung angezeigt und es besteht für den Nutzer die Möglichkeit sich die richtige Lösung anzusehen. Im Prüfungsmodus wählt das System mit Hilfe von bestimmten Auswahlkriterien (festgelegt vom Autor) die Aufgaben aus und es findet eine automatische Korrektur der Prüfung statt.

Inhaltsverzeichnis

1 Was ist eTesting?

Seit einiger Zeit prägt der Begriff eLearning die IT-Branche. Egal ob als lokale Version oder als Version im Internet, die Vorteile des „Computer-Based-Training“ liegen auf der Hand. Multimedial aufbereitete Lehrinhalte stehen zu jeder Zeit zur Verfügung, unabhängig vom Terminkalender des Dozenten.
Schulen, Hochschulen und auch Firmen, die interne Mitarbeiter-Schulungen durchführen, haben diesen Trend erkannt und handeln verstärkt mit der Veröffentlichung neuer eLearning-Systeme mit unterschiedlichsten Lehrinhalten. Egal was man sucht, Mathematik, Chemie oder sogar den Kochkurs für Single-Männer, man findet fast alles im Internet.
Natürlich bleibt es nicht aus, dass der Ruf nach Prüfungen, die online absolviert werden können, immer lauter wird. Zum Beispiel die Studenten der Fern-Hochschule Darmstadt können jegliche Vorlesungen zu Hause erarbeiten bzw. teilweise die Online-eLearning-Systeme der Hochschule nutzen, zu den Prüfungen müssen sie dennoch an die Hochschule nach Darmstadt, was einen erheblichen Zeitaufwand bedeutet und den Vorteil der „Zeitunabhängigkeit“ zum Teil wieder zu Nichte macht.
eTesting, was kurz gesagt soviel bedeutet wie die Prüfung online zu absolvieren, wird also immer öfter Bestandteil von eLearning-Systemen. Beim eTesting spricht man allerdings nicht nur von der Möglichkeit, die Prüfungen online durchzuführen, sondern vielmehr auch von der sofortigen Korrektur des Testes durch das IT-System. Das heißt, dass das System in der Lage ist, zwischen „Richtig“ und „Falsch“ zu unterscheiden und die einzelnen Prüfungsaufgaben bewertet. Letztendlich werden Dozent und Student automatisch über das Ergebnis informiert. eTesting soll nicht nur den Studenten die freie Zeiteinteilung ermöglichen, sondern auch den Lehrenden die unheimlich arbeitsintensive Korrekturarbeit erleichtern. Die Lehrkraft „füttert“ das System mit den entsprechenden Prüfungsfragen und legt den Bewertungsschlüssel fest. Bei jeder Frage muss das Prüfungsverfahren vom Prüfer festgelegt werden.

2 Welche verschiedenen Prüfungsverfahren gibt es?

Um die Leistung des Schülers eindeutig und fair bewerten zu können, muss bei jeder Prüfung, egal ob mündlich, schriftlich oder am Computer, das für das Prüfungsthema geeignete Prüfungsverfahren gewählt werden. Die bekannteste und weit verbreitete Variante ist wohl die Variante der „freien Textantworten“. Aber auch die anderen Prüfungsmethoden werden immer häufiger angewandt. Bei dem Führerschein-Prüfung zum Beispiel wird seit Jahren erfolgreich das Multiple-Choice-Verfahren angewendet. Im Folgenden soll auf die verschiedenen Prüfungssysteme näher eingegangen und einige Begriffsklärungen getroffen werden.

2.1 Freie Textantworten

Die Variante der „freien Textantworten“ ist die am häufigsten gewählte Methode. Meines Wissens ist dies das derzeit einzig zugelassene Prüfungsverfahren an der FH-Aachen. Bei dieser Methode gibt der Student dem Dozenten seine Antwort in frei formulierter Art und Weise. Dieser muss nun jede Antwort individuell prüfen und bewerten. Der große Vorteil liegt mit Sicherheit bei den fast unendlichen Möglichkeiten, wie die Frage gestellt werden kann und natürlich auch die Antwort lässt viele Möglichkeiten zu, die Frage mehr oder weniger präzise zu beantworten. Dadurch hat der zu Prüfende die Möglichkeit die Antwort so zu beantworten, wie es sein Wissen am besten zulässt. Er kann die Frage eher kurz und wage beantworten um an anderer Stelle sein Wissen in aller Fülle dem Prüfer zu präsentieren. Auch die Möglichkeit einen „entfallenen“ Begriff zu umschreiben ist aus Sicht des Studenten wohl nicht zu verachten. Allerdings ist der Aufwand für den Lehrenden sehr hoch. Wie schon erwähnt, muss jede Antwort individuell korrigiert und bewertet werden. Die mitunter eigentümlichen Antworten machen es dem Korrektor nicht immer leicht zu entscheiden, ob die Frage zu aller Zufriedenheit beantwortet oder am Ziel vorbei geschossen wurde.

2.2 Single Choice

Obwohl es deutliche Unterschiede gibt, wird das so genannte „Single Choice“ - Verfahren oft mit dem „Multiple Choice“ - Verfahren verwechselt bzw. in einen Topf geschmissen. Bei dieser Methode gibt der Prüfer n Antworten vor, von denen genau **eine** Richtig ist. Der Student muss nur die richtige Antwort entsprechend der vorgegebenen Form markieren. Die Korrektur beschränkt sich hier auf die Überprüfung der Markierungen. Entgegen dem „Multiple Choice“ - Verfahren (siehe dort) gibt es nur die richtige bzw. falsche Beantwortung der Frage, dementsprechend einfach ist die Bewertung.

2.3 Multiple Choice

Die Variante „Multiple Choice“ lässt entgegen der vorher erwähnten „Single Choice“ - Methode bei n möglichen Antworten *eine* bis n richtige Antworten zu, in Sonderfällen ist es sogar möglich, dass **keine** der Antworten richtig ist. Es kann also eine, zwei oder gar alle Antworten von dem Studenten als richtig gekennzeichnet werden. Hier ist die Bewertung schon wieder wesentlich aufwändiger als bei einer „Single Choice“ - Prüfung.
Es gibt eben nicht nur die Unterscheidung zwischen „Richtig“ und „Falsch“, sondern auch eine unvollständig gegebene Antwort muss entsprechend gewertet werden. Das heißt, es muss ein Bewertungsschlüssel gefunden werden, der alle Antwort-Möglichkeiten abdeckt und trotzdem nicht zu kompliziert ausfällt. Wählt man einen Bewertungsschlüssel bei dem jede Antwortmöglichkeit separat auswertet wird, das heißt, jede Markierung der Antwort wird separat gewertet, kann es unter bestimmten Voraussetzungen dazu kommen, das systematisch gegebene Antworten als richtig gewertet werden. Beinhaltet die Frage mehr richtige Antworten als falsche kann der Student durch konsequentes Ankreuzen aller Antworten ein Ergebnis von mehr als 50% erzielen. Genau diese Möglichkeit muss ausgeschlossen werden.
Da bei den meisten Bewertungsschlüsseln jede Frage für sich betrachtet werden muss (sind alle Voraussetzungen für diese Frage geschaffen?) ist die Bewertung oftmals aufwändig. Der einzige Bewertungsschlüssel, der unabhängig von der Frage und somit schnell angewendet werden kann, ist die Methode, wie sie seit einigen Jahren erfolgreich in der Prüfung zum Führerschein eingesetzt wird. Die Frage wird nur als „Richtig“ gewertet, wenn alle Markierungen richtig gesetzt wurden. Hier ist die Schnelligkeit von wesentlicher Bedeutung, da der Schüler nur bei bestandener Theorie-Prüfung an der anschließenden praktischen Prüfung teilnehmen darf.

2.4 Lückentexte, Schlüsselwörter

Lückentexte bzw. das Abfragen von so genannten Schlüsselwörtern ist als eine vereinfachte Form der „freien Textantworten“ anzusehen. Da jedoch die Korrektur wesentlich einfacher ist als bei den „freien Textantworten“, haben wir hier eine echte Alternative zu den Single / Multiple Choice - Fragen. Bei den Schlüsselwörtern beantwortet der Student die Frage in Stichpunkten oder auch in ganzen Sätzen. Es kommt bei dieser Art der Prüfung letztendlich nur auf die Nennung bestimmter Begriffe an. Das heißt, dass der Dozent eine Liste von ge-

suchten Begriffen vorgibt und nach genau diesen in der Antwort sucht. Wurden alle Begriffe genannt, so wird die Frage als „Richtig“ gewertet. Die Bewertung ist entsprechend einfach und lässt auch einen einfachen Bewertungsschlüssel für unvollständige Antworten zu.
Bei Lückentexten gibt der Prüfer dem Studenten einen entsprechenden Text vor, in dem die Begriffe fehlen, die der Dozent abfragen möchte. Die Aufgabe des Studenten ist es nun die entsprechenden Lücken zu füllen. Da Lückentexte im engeren Sinn auf der Methode der Schlüsselwörter basieren, liegt auch hier der große Vorteil bei der sehr einfachen Korrektur, da die Antwort des Studenten nur mit dem korrekten Begriff verglichen werden muss. Allerdings gibt es hier wiederum nur „Richtig“ oder „Falsch“. Bei beiden Methoden liegt der große Nachteil auf der Seite des Geprüften. Er muss alle Begriffe exakt kennen und hat nicht einmal die Möglichkeit eine der gesuchten Antworten zu umschreiben. Dennoch kann, wie schon erwähnt, diese Methode in einigen Lehrgebieten eine Alternative zu den „freien Textantworten“ sein.

3 Wie könnte ein eTesting-System für NUMAS aussehen?

Bei der Konzeption einer eTesting-Umgebung für das Lehr- und Lernsystem „NUMAS“ wäre es natürlich denkbar, einen einfachen „Single Choice“ - Fragenkatalog zu erarbeiten und diesen als Testumgebung zu präsentieren. Allerdings würde man, bei der Vielfalt von Prüfungsmethoden, dem äußerst aufwendigen und komplexen Lehr- und Lernsystem mit dieser einfachen Vorgehensweise nicht gerecht. Außerdem wäre es wünschenswert, wenn die Möglichkeit der Erweiterung des Fragenkataloges zu jeder Zeit geben ist. Es muss also die Frage gestellt werden: „Wie könnte ein eTesting-System für Numas aussehen und wie lässt sich die erarbeitete Lösung mit dem heutigen Stand der Technik realisieren?“

3.1 Wie sehen andere Systeme aus und welche Möglichkeiten bieten diese?

Um der Frage nach dem „Stand der Technik“ ein Gesicht zu geben, wurde als erstes recherchiert, wie andere Systeme heute aussehen und welche Möglichkeiten diese bieten, sprich welche Prüfungsmethoden implementiert wurden.
Sucht man mit der Internet-Suchmaschine „Google“ nach dem Begriffen „eLearning“ und „Test“ spuckt das System 38600 Treffer aus (Stand: Oktober 2005). Diese enorme Zahl der Internetseiten macht eine erneute Auswahl des Suchbegriffes notwendig. Die anschließende Suche nach dem Begriff „eTesting“ bringt immerhin noch stolze 145 Treffer.
Obwohl ich längst nicht alle gefundenen Testsysteme ausprobieren konnte, die meisten lassen sich nur mit einer gültigen Anmeldung in der jeweiligen Schule besuchen, ist das Ergebnis dieser Recherche äußerst ernüchternd. Die meisten Systeme sind reine „Single Choice“ - Tests und verfügen in den seltensten Fällen über die ausreichenden Fähigkeiten, um als eTesting-System genannt zu werden. So bietet zum Beispiel die Universität in Melbourne, Australien ein „Online-Test-System“ an, bei dem es keine IT-basierende Auswertung gibt. Das System bietet zwar „freie Textantworten“, diese werden aber von einem Mitarbeiter der Universität korrigiert und wieder online zum Abruf bereit gestellt.
Des Weiteren treten gehäuft Mängel, wie zum Beispiel das Fehlen einer Zeitbegrenzung (Leistung ist Arbeit in einer gewissen Zeit!) auf. Auch Mängel in der Auswertung, wie zum Beispiel eine sinnvolle Stufung der Bewertung der „Multiple Choice“ - Fragen, sind keine Seltenheit.
Ich möchte an dieser Stelle auf zwei besonders hervorstechende Systeme eingehen. Zu den restlichen von mir besuchten Webseiten, die ein annähernd vollständiges „Online-Test-System“ zur Verfügung stellen, lässt sich die Internetadresse im Anhang finden.

Die Firma Berg Communications hat in Zusammenarbeit mit der Firma Leybold Didactic die Software „OTeS“ entwickelt. „OTeS“ besteht im wesentlichen aus drei Komponenten: Das Modul „Test“ das Modul „Editor“ sowie das Modul „Net“.
Das Modul „OTeS Test“ ist das Prüfungsprogramm, welches auf dem Prüflings - PC ausgeführt wird. Das Prüfungsprogramm können die Prüflinge auf ihrem eigenen Rechner installieren und die Fragen im Prüfungs- oder Übungsmodus beantworteten. Wie am Anfang schon einmal erwähnt, weist das System alle Merkmale eines echten eTesting-Systems auf. Sogar die Möglichkeit einer Zeitbegrenzung ist implementiert. Die Studenten haben die Möglichkeit, nach der Eingabe der persönlichen Daten, sich in aller Ruhe wichtige Informationen wie erreichbare und zum Bestehen benötigte Punktzahl, Anzahl der Aufgaben und verfügbare Zeit anzusehen. Ist der Test einmal gestartet, so beginnt die Uhr zu laufen. Der Student hat zu jeder Zeit die Möglichkeit sich einen Überblick über den Stand des Tests zu verschaffen und eine Aufgabenübersicht gibt ihm die Möglichkeit die Aufgaben in der von ihm gewünschten Reihenfolge zu beantworten. Im Übungsmodus kann der Student sich die Lösung der aktuell gezeigten Aufgabe anzeigen zu lassen.
Das wichtigste Merkmal einer eTesting - Umgebung ist die automatische Korrektur, welche auch beim OTeS - System sofort nach der Beendigung des Tests stattfindet. Die Testergebnisse werden vom System gespeichert und können vom Lehrer eingesehen werden.

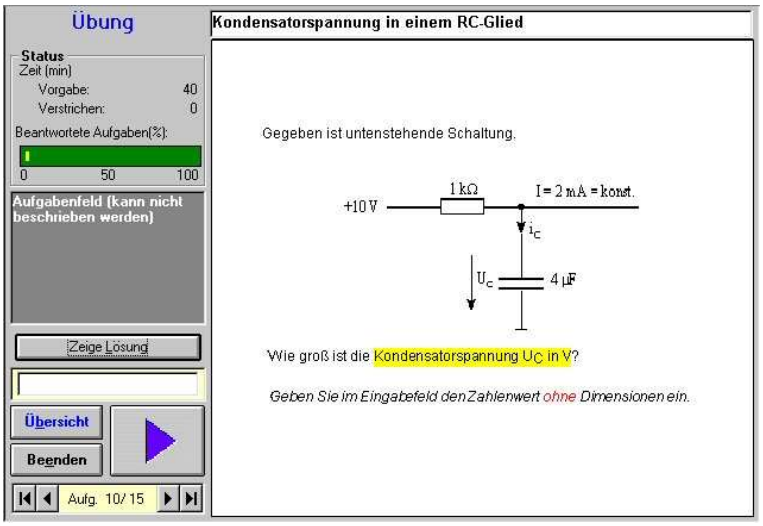

Abbildung 3.1: Modul „OTeS - Test“ im Übungsmodus [1]

Mit dem Autorentool „OTeS Editor“ lassen sich nicht nur neue Prüfungsaufgaben erstellen, sondern auch Prüfungen zusammenstellen und der Zugriff auf die Prüfungsergebnisse wird ermöglicht. Selbstverständlich lassen sich auch bereits erstellte Aufgaben editieren und löschen.
Um eine neue Aufgabe anzulegen, wählt hier der Prüfer den entsprechenden Aufgabentyp (Multiple Choice, Schlüsselwort) aus und kann den entsprechenden Text einfügen. Es lassen sich ebenso Bilder und Grafiken in den Fragentext integrieren.
Das Programm sieht den Import von gängigen Formaten, wie zum Beispiel MS Word-Dokument vor, um die Eingabe der Aufgabenstellung zu erleichtern.

Leider lässt OTeS nur zwei Aufgabentypen zu: Multiple Choice und Schlüsselwort (bei OTeS als Kurzantwort bezeichnet). Es wird nicht zwischen Single- und Multiple Choice unterschieden und auch das Verfahren der Schlüsselwörter ist sehr begrenzt. Es ist lediglich die Eingabe von einem Wort möglich, das heißt, dass der Prüfer an eine einfache Antwort gebunden ist. Das System lässt zum Beispiel bei einer Rechenaufgabe nur einen einfachen Zahlenwert **ohne** Dimension zu.

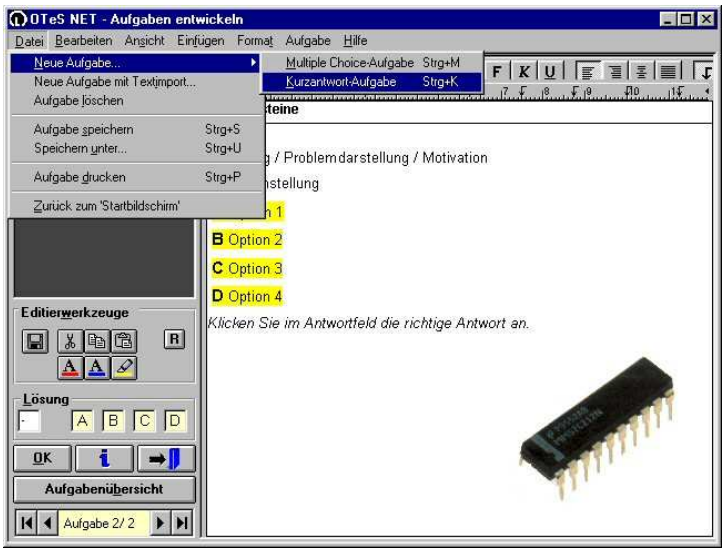

Abbildung 3.2: Modul „OTeS - Editor“ im Modus „Aufgaben entwickeln“ [2]

Die Fragen werden in fachspezifischen Sammlungen geordnet, die der Autor selbst auswählen kann. Die Aufgabenbestände können also übersichtlich und einfach verwaltet werden.
Eine Filterfunktion ist in die Datenbank integriert, um bestimmte Aufgaben ausfindig zu machen. Im Editor-Modus lassen sich dann manuell Prüfungen aus dem Fragenkatalog zusammenstellen, der Test wird dann auf den Schülerrechner zu transferiert.

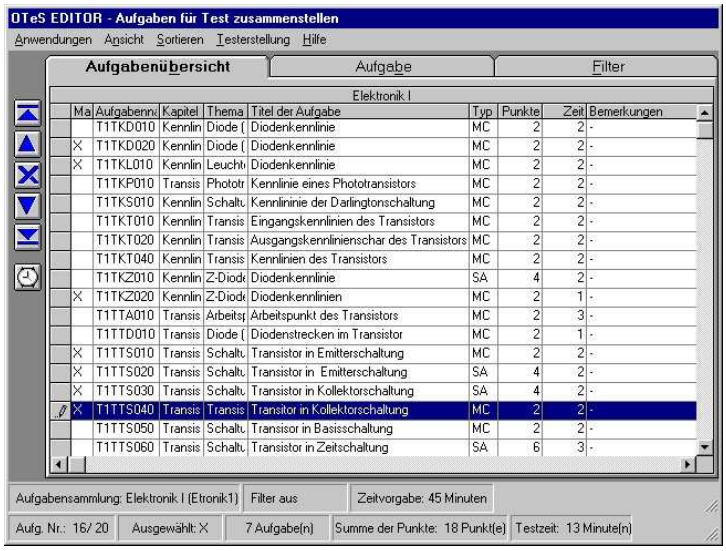

Abbildung 3.3: Modul „OTeS - Editor“ im Modus „Test zusammenstellen“ [2]

Das System korrigiert die Testergebnisse automatisch. Der Lehrer kann allerdings bei Bedarf in die Korrektur eingreifen und eventuell die Ergebnisse manuell korrigieren.
Letztendlich kann der Prüfer Zeugnisse und Zertifikate über das Testergebnis ausdrucken, wobei das Aussehen des Dokuments durch den Prüfer bestimmt werden kann.

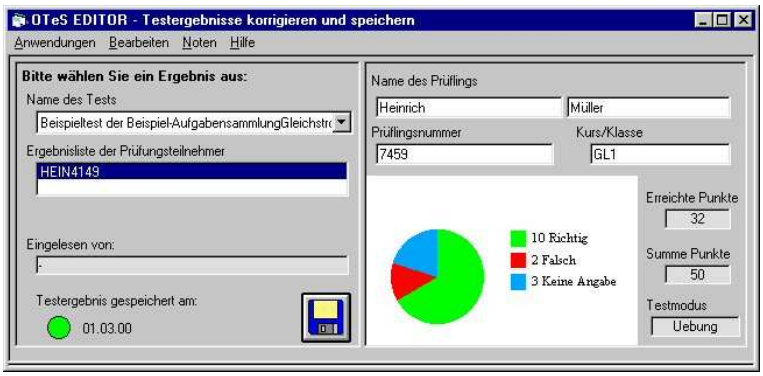

Abbildung 3.4: Modul „OTeS - Editor“ im Modus „Auswertung einer Prüfung“ [2]

Um Online-Prüfungen im PC-Netzwerk durchführen zu können, stellt die Firma Berg Communications das Modul „OTeS Net“ zur Verfügung. Das Modul enthält zu Funktionalitäten des Editor weitere Tools um den Betrieb im Netzwerk zu erleichtern. Nach dem Einrichten des OTeS Netzwerkes stehen dem Prüfer Funktionalitäten zur Verfügung, um die Tests direkt für die einzelnen Prüflingsrechner bereit zu stellen. Ebenso lässt sich auf dem Prüfer-PC darstellen, auf welchen PC's gerade aktiv ein Test durchgeführt wird, sowie die Anzeige der aktuellen Teststände auf den Prüflingsrechnern. Das Einlesen der Testergebnisse und das anschließende Löschen des Tests auf dem Prüflingsrechner geschieht auch über das Netzwerk.

Abschließend ist zu bemerken, dass das vorgestellte System nur ein eingeschränktes eTesting-System ist, da es lediglich die Prüfung via Intranet vorsieht und keine Möglichkeit zur Benutzung via Internet bietet. In der Basis-Version mit dem Modul „OTeS Editor“ geschieht die Kommunikation zwischen dem Lehrer- und dem Prüflingsrechner über einen Wechseldatenträger.
Die Firma Berg Communications stellt, laut ihrer Internetseite, auch eine Internetversion zur Verfügung, leider lässt sich diese nicht beurteilen, da Informationen und Preis darüber nur auf konkrete Anfragen erteilt werden. Auf meine E-Mail-Anfrage reagierte die Firma Berg Communications nicht.
Die Internetversion findet auch nur an einer Stelle auf der Homepage eine kurze Bemerkung. Konkret heißt es dort: „Wenn Sie Ihren vorhanden OTeS-Aufgabenpool im Internet einem definierten OTeS-Nutzerkreis zum Download zur Verfügung stellen möchten, übernimmt BERG Communications alle dafür notwendigen Serviceleistungen. Von der Bereitstellung der OTeS-Plattform für die Internetnutzung bis hin zum Installation der Bank auf einem Internetserver bietet BERG Ihnen hierfür ein Gesamtpaket auf höchstem technischen Niveau. Preis auf Anfrage“ [3]
Das Modul „OTeS Editor“ kostet 2.958,00 €. [3]
Das Modul „OTeS Net“ kostet 4.141,20 €. [3]
Das Modul „OTeS Test“ kann man kostenlos downloden [3]

Ein weiterer, zu erwähnender Test wird von der Firma „iTrain GmbH, Schweiz“ durchgeführt. Die Firma bietet verschiedene Kurse, wie zum Beispiel den „MCSA - Microsoft Certified System Administrator“ an. Bevor man sich zu dem Kurs anmelden kann, wird man aufgefordert online einen Einstufungstest zu absolvieren. Dieser Test weist alle Merkmale eines eTesting-Systems auf: Die Prüfung wird via Internet abgehalten, hat eine Zeitbegrenzung, enthält verschiedene Prüfungsverfahren und wird letztendlich sofort vom System ausgewertet.Das Ergebnis wird anschließend angezeigt. Leider war es nicht möglich hinter die Kulissen des Kurses zu sehen, somit ist es nur möglich die Prüflings-Umgebung zu kommentieren. Die Art der Fragenerstellung und Verwaltung blieb im Verborgenen.
Zu den verschiedenen Prüfungsverfahren zählen Single- und Multiple Choice, sowie die Abfrage eines Schlüsselwortes. Wie auch schon bei OTeS wird hier nur nach einem Wort gefragt.

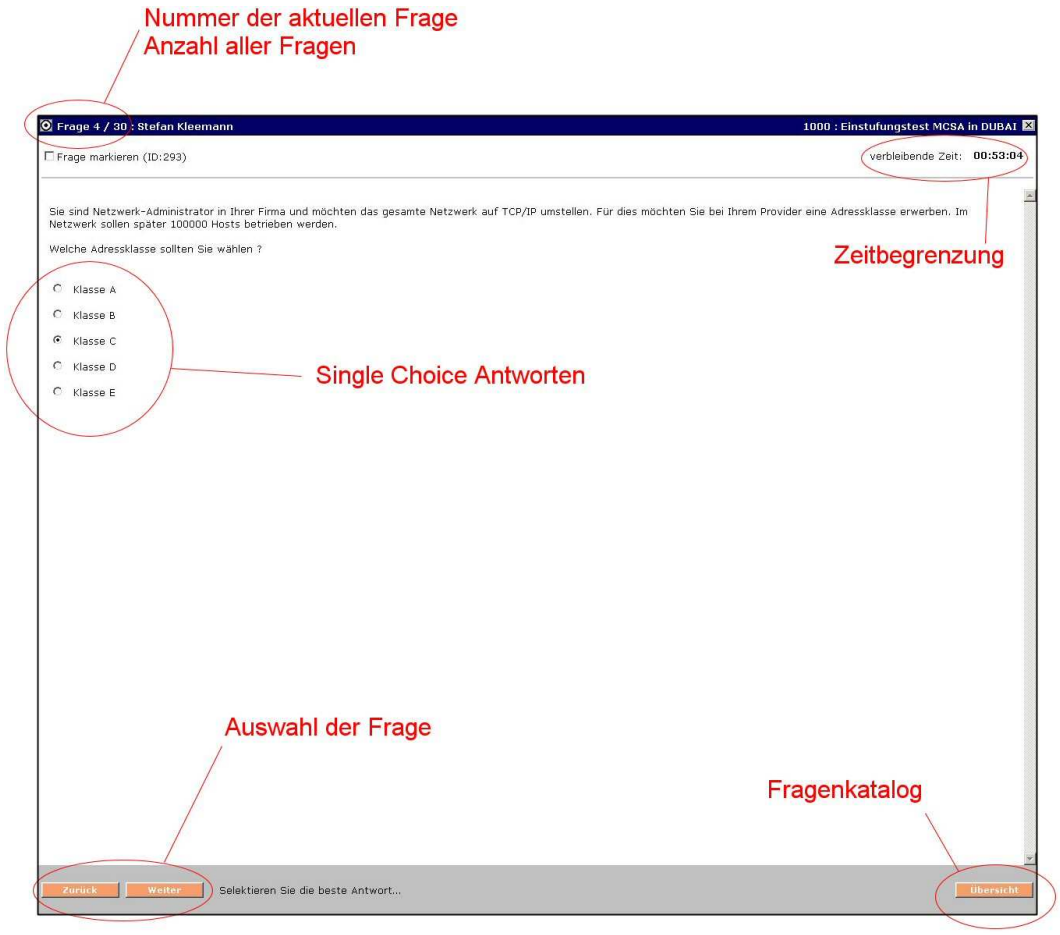

Abbildung 3.5: MCSA - Einstufungstest: Bildschirmaufbau [4]

Der grundsätzliche Bildschirmaufbau ist in der Abbildung 3.5 dargestellt. Alle für den Bewerber relevanten Informationen sind übersichtlich angeordnet und befinden sich ständig im Blickfeld.
Im Hauptteil des Fensters befindet sich die Fragestellung und die Antwortmöglichkeit. In diesem Fall wurde das Single-Choice-Verfahren mit 5 Antworten gewählt.
In der Titelleiste wird die Gesamtanzahl der Fragen und die Nummer der aktuellen Frage eingeblendet um den Prüfling eine Übersicht zu geben, wie viele Fragen er noch zu beantworten hat.
Eine Zeitanzeige in der linken oberen Ecke gibt Auskunft über die noch verbleibende Zeit. Nach Ablauf der Zeit beendet das System die Prüfung und wertet die bis dahin gegebenen Antworten aus.
Am unteren Rand befinden sich die Bedienelemente zur Fragenauswahl. Der Nutzer hat die Möglichkeit, mit den Buttons „Weiter“ und „Zurück“ Schritt für Schritt durch den Fragenkatalog zu gelangen. Er hat allerdings ebenso die Möglichkeit sich eine Gesamtübersicht der Fragen anzusehen und die entsprechende Frage auszuwählen.

Abbildung 3.6: MCSA - Einstufungstest: Schlüsselwort-Verfahren [4]

Wie schon erwähnt werden verschiedene Prüfungsverfahren angewendet. Bei der dritten Frage der Prüfung kam eine Schlüsselwortabfrage zum Einsatz. Es wurde nach einem Parameter zum Befehl „ipconfig“ gefragt und nicht nach dem gesamten Befehl mit allen Parametern. Das lässt vermuten, dass das System nicht in der Lage ist nach mehreren Schlüsselwörtern zu fragen, sondern nur einzelne Worte vergleichen kann. Mir blieb leider auch im Verborgenen, wie genau der Begriff genannt werden muss. So wäre es ja denkbar, dass das System über Rechtschreibfehler oder typische Tippfehler hinweg sieht.

Im Test ist klar zwischen Single- und Multiple-Choice unterschieden: Der Programmierer wählte für die Single-Choice Antworten „Radio Buttons“, beim Multiple Choice Verfahren kamen „Check Boxes“ zum Einsatz. (siehe Abbildungen 3.5 und 3.7)

Abbildung 3.7: MCSA - Einstufungstest: Multiple Choice Frage [4]

Besonders auffällig waren die Single-Choice Fragen, da sie mit Grafiken unterlegt sind. In diesem Fall wurde das Antwortfeld mit einer Grafik unterlegt, auf der man per Mausklick ein rotes Kreuz setzten und damit die Antwort geben konnte. In der Abbildung 3.8 wird diese Abwandlung des Single-Choice dargestellt. Die allgemein bekannte Windows-Systemsteuerung wurde als Grafik in die Fragestellung eingefügt und der Bewerber kann nun die einzelnen Symbole per Mausklick markieren. Nachdem der Test durch den Prüfling oder den Ablauf der Zeit abgeschlossen wird, wird er automatisch vom System ausgewertet und das Testergebnis wird auf einer HTML-Seite mitgeteilt. Außerdem wird eine E-Mail mit dem Testergebnis an den Geprüften versandt. Wie das Ergebnis an die Firma iTrain GmbH übermittelt wird, ist unbekannt.

Abbildung 3.8: MCSA - Einstufungstest: Single Choice Frage mit Grafik unterlegt [4]

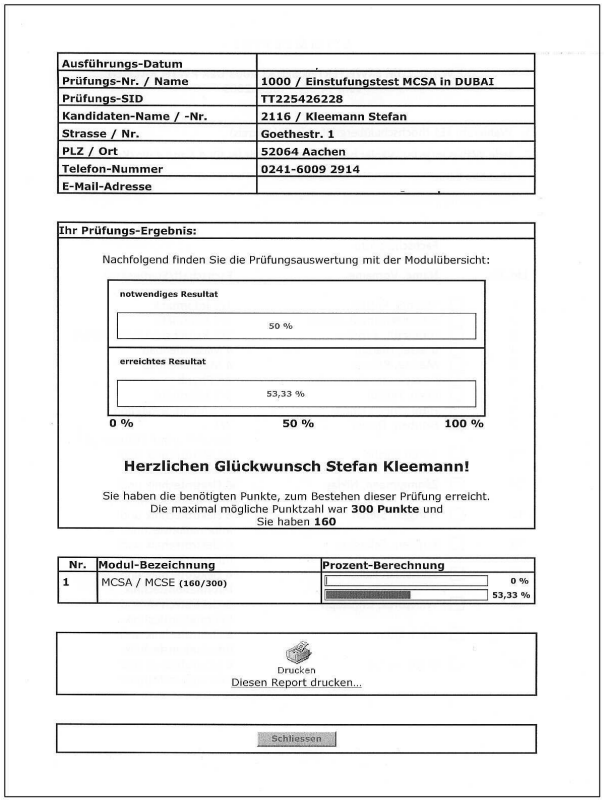

Abbildung 3.9: MCSA - Einstufungstest: HTML-Seite mit dem Test-Ergebnis [4]

3.2 Was sollte unser System können?

Die im Moment angebotenen Testfragen am Ende verschiedener Lernobjekte sind äußerst schwierig wartbar und kaum erweiterungsfähig.

Abbildung 3.10: Testfrage im Lernfeld Quadratur des NUMAS Systems [5]

Da die Realisierung dieser Testfragen auf einer statischen HTML-Seite mit eingebetteten Javascript Funktionen basiert, ist für die Änderung bzw. Erweiterung der Fragen ein erfahrener Web-Entwickler notwendig. Änderungen oder Erweiterungen können in den meisten Fällen nicht von den Autoren selbst eingebracht werden, sondern es Bedarf auch der Mitwirkung eines Programmierers. Gerade der Punkt der Flexibilität des Systems ist ein enorm wichtiges Kriterium für die problemlose Verwendung eines Übungs- bzw. Prüfungsmoduls. In den meisten Fällen wächst ein Katalog von Übungs- und Prüfungsfragen mit der Zeit, das heißt, ein solcher Fragenkatalog entsteht über die Semester und verändert sich ständig. Viele Lehrende bemühen sich die Übungsaufgaben und vor allem auch die Prüfungsaufgaben möglichst abwechslungsreich zu gestalten, das bedeutet, dass sich mit jedem Semester die Aufgaben ändern und sich damit der Fragenkatalog stetig wandelt. Deshalb ist es notwendig, dass die Autoren eine Möglichkeit haben diese Veränderungen schnell und unkompliziert in das System einzupflegen und bereitzustellen. Es ist also notwendig eine Möglichkeit zu bieten, die es ermöglicht Fragen zu erstellen, ohne sich über Design oder Programmieraufwand Gedanken machen zu müssen. Lediglich der Inhalt, das bedeutet die Fragestellung, das Prüfungsverfahren und die Antwort sollen vom Autor bereitgestellt werden. Es darf keine Notwendigkeit bestehen, dass der Autor weitere Schritte, wie zum Beispiel die Beauftragung eines Programmierers, einleiten muss, um die neuen Übungs- bzw. Prüfungsaufgaben zu präsentieren. Die Flexibilität des Aufgabensammlung ist eine der wichtigsten Anforderungen an

das neue System, da die Erstellung einer solchen Fragenpools, im Gegensatz zu den Lehrinhalten, ein dynamischer Prozess ist. In diesem Zusammenhang sei darauf hingewiesen, dass NUMAS, wie viele andere Projekte auch, nicht nur von einem Autor betreut wird, sondern eine Vielzahl von Autoren das eLerning System mit Inhalt füllen. Die Autoren sind im Fall von NUMAS räumlich getrennt (Aachen, Berlin und Münster), was die Forderung nach einem Zugriff auf den Fragenkatalog, der in einfacher Art und Weise für alle Autoren gleichermaßen zur Verfügung stehen sollte, nicht vereinfacht. Es ist also wichtig, dass verschiedene Autoren gleichermaßen auf das System zugreifen und den Fragenkatalog entsprechend manipulieren können, egal ob sie sich vor Ort befinden oder nicht.
NUMAS deckt eine weite Bandbreite an Themengebieten ab und auch das Spektrum der Nutzer ist groß. Dem Anfänger und auch dem versierten Anwender, der sich eingehender mit der Materie beschäftigen möchte, bietet das System Lerninhalte zum Selbststudium bzw. als studiumsbegleitende Informationen. Diese Komplexität von NUMAS verlangt ein Prüfungs- / Übungsmodul mit einem Angebot an verschiedenen Prüfungsverfahren, damit möglichst viel der Inhalte sinnvoll abgefragt werden kann. Wie schon in den letzten Kapiteln angesprochen, gibt es im Prinzip nur drei Prüfungsverfahren, die sich für ein eTesting System eignen. Alle drei Prüfungsverfahren sollten im System angeboten werden, um NUMAS gerecht zu werden. Im Moment werden alle Testfragen des Systems mit den Methoden „Single bzw. Multiple Choice“ abgedeckt. In der Abbildung 3.10 wird eine „Multiple Choice“ Frage gezeigt. Zu der gestellten Frage gehören drei Lösungsmöglichkeiten, von denen zwei (a und b) richtig sind. Im Gegensatz zu den meisten anderen Systemen (siehe letztes Kapitel) kommen hier keine „Checkboxen“ zum Einsatz, sondern die Antworten werden hier mit „wahr“ oder „falsch“ markiert. Auf die gleiche Art und Weise wurden „Single Choice“ Fragen realisiert. Für diese Fälle sollte man auf die „Standard Verfahren“ („Checkboxen“ und „Radiobuttons“) zurückgreifen, da sich viele Leute schon an bestimmte Vorgehensweisen am Computer und im Internet gewöhnt haben. Es macht wohl keinen Sinn von diesen Möglichkeiten keinen Gebrauch zu machen.
Die Frage nach den Schlüsselwörtern bzw. Lückentexten gestaltet sich schon wesentlich schwieriger. Es ist relativ unstrittig, dass es wohl Sinn macht eine derartige Prüfungs-/Übungsmöglichkeit vorzusehen. Mathematische Inhalte fordern förmlich die Eingabemöglichkeit von Zahlen: Eine Rechenaufgabe ist im „Single bzw. Multiple Choice“ Verfahren nicht unbedingt einfach zu stellen. Die Möglichkeit Ergebnisse direkt abzufragen ist mit Sicherheit eine große Bereicherung für das Gesamtsystem.
Auch Lückentexte sind gerade bei der Vermittlung von Grundlagen ein wichtiges Instrument, da gerade hier oftmals neue Begriffe erklärt und entsprechend geprüft werden sollen. Dabei ist zu beachten, dass für eine Lücke auch mehrere Antworten richtig sein können. Der Autor sollte also die Möglichkeit haben, mehrere Antworten für eine Lücke bereitzustellen.

Die eindeutige Forderung an das System ist es, sowohl ein Übungsmodul als auch ein Prüfungsmodul bereitzustellen. Es ist völlig einleuchtend (und die Recherche bei anderen Systemen hat das bestätigt), dass es deutliche Unterschiede zwischen den beiden Systemen gibt und dennoch gibt es auch einige grundlegende Dinge, die von beiden Modulen benötigt werden. So verfügen alle beide über einen Aufgabenpool mit entsprechenden Antworten, eine Darstellungsmöglichkeit der Fragen und auch die Möglichkeit einer Auswertung ist immer

nötig. Beide Module sollten sich unkompliziert in jedes eLerning System einfügen lassen und entsprechende Möglichkeiten zur Designänderung bieten.
Der markanteste Unterschied zwischen den Modulen ist wohl der weitere Ablauf nach der Beantwortung bzw. der Auswertung einer Frage. Im Gegensatz zum Übungsmodul bekommt der Prüfling im Prüfungsmodul die Lösung nach der Beantwortung nicht präsentiert und ihm wird sofort die nächste Frage gestellt. Im Übungsmodul wird ihm die richtige Antwort gezeigt und seine Fehler kenntlich gemacht, außerdem hat er die Möglichkeit sich weiterführende Erklärungen anzusehen. Eine Prüfung enthält in der Regel ein zeitliches Limit und die gesamte Prüfung wird nach Beendigung (entweder wurden alle Fragen beantwortet oder die Zeit ist um) ausgewertet. Nach der Auswertung muss der Prüfer entsprechend informiert werden.
Eine Forderung nach einem solch komplexen und flexiblen System wirft die Frage nach der Realisierbarkeit auf.

3.3 In wie weit kann die Realisierbarkeit jetzt schon abgeschätzt werden?

Die Flexibilität des Systems kann mit Sicherheit mit Hilfe gängiger Techniken von modernen Content-Management-Systemen erreicht werden. Das heißt, dass lediglich die Inhalte abgespeichert werden und das Design zur Laufzeit erzeugt wird. Es wird also nur die Frage und die Antwort abgespeichert und erst bei Aufruf der Internetseite wird die Aufgabe formatiert und angezeigt. Nach der Beantwortung durch den Nutzer erfolgt die Auswertung dynamisch mit der hinterlegten Antwort. Diese Erzeugung zur Laufzeit hat sich im Internet inzwischen als gängige und stabile Lösung etabliert. Somit stehen auch viele Entwicklungshilfen und Bibliotheken für die verschiedenen Programmiersprachen zur Verfügung. Das bedeutet, dass die Verwaltung des Fragenkatalogs und die geforderte Flexibilität mit Hilfe der im Internet üblichen Programmiersprachen und Tools (z.B. Datenbanken) realisiert werden kann.
Ein größeres Problem dürfte sich bei der Bewertung der Aufgaben ergeben. Im Gegensatz zu den „Single Choice Fragen“ (hier gibt es nur „richtig“ oder „falsch“) ist es bei den „Multiple Choice Fragen“ schwierig zu entscheiden, ob die Frage „richtig“ oder „falsch“ beantwortet wurde. Die einfachste Methode hierbei ist die Methode, die bei der deutschen Führerscheinprüfung benutzt wird: Die Frage gilt nur als richtig, wenn alle Kästchen richtig markiert wurden. Es wäre ja auch denkbar, jede Antwort für sich zu bewerten („wahr oder falsch“). Die Realisierung beschränkt sich hierbei dann auf eine Methode, das heißt, dass nicht jeder Autor für sich entscheiden kann, wie bewertet wird. Der beste Ansatz ist hier wohl in der „Führerschein-Methode“ zu sehen.
Auch bei der Realisierung der Schlüsselwort-Antworten wird es zu deutlichen Einschränkungen kommen. Es stellt sich dann natürlich die Frage wann eine Lösung als „Richtig“ gewertet wird. Werden Rechtschreibfehler automatisch als eine falsche Antwort gewertet? Wo ist die sinnvolle Grenze? So kann zum Beispiel eine Antwort mit zwei falschen Buchstaben schon nicht mehr als „Richtig“ gewertet werden. So verändert zum Beispiel die Vorsilbe „Un-“ den Sinn des Wortes komplett. Eine Antwort mit Zahlen verträgt keinerlei Ungenauigkeit. Die Realisierung von Schlüsselwörtern wird sich wohl, zu Ungunsten des Prüflings, auf eine ge-

naue Lösung beschränken.
Letztendlich wird wohl die Unterstützung für den Autor, Fragen in den Fragenkatalog aufzunehmen, auf eine eingeschränkte Lösung hinauslaufen. Der Editor zur Erstellung /Änderung der Fragen wird sich wohl am besten in der Form eines Assistenten realisieren lassen. Der Autor bekommt also bestimmte Auswahlmöglichkeiten angezeigt und die Frage wird entsprechend der Eingabe automatisch erstellt und abgespeichert. Der Autor hat also keinen direkten Einfluss auf die Formatierung, sondern nur entsprechende Auswahlmöglichkeiten (z.B.„Zentriert“) zur Verfügung. Diese Vorgehensweise wird notwendig sein, da jetzt schon abzusehen ist, das die Speicherung der Aufgabeninhalte relativ komplex wird.
Alles in allem ist aber zu erwarten, dass sich ein System mit den genannten Einschränkungen realisieren lässt.

4 Konzeption

„Eine Holzhütte kann man noch ohne Plan bauen, ein Hochhaus nicht."
Ähnliches gilt auch für die Softwareentwicklung. Für die meisten Projekte ist es fast unmöglich ohne ein „wasserdichtes" Konzept mit der Realisierung anzufangen. In diesem Kapitel soll die Konzeption des eTesting Systems für NUMAS vorgestellt werden. Da eine gründliche Recherche Basis für ein Konzept ist, orientiert sich das Konstrukt an der vorangegangenen Recherche und baut auf den Überlegungen der vorangegangenen Kapitel auf.

4.1 System Requirements

Requirements sind Beschreibungen der Anforderungen an ein System. Ein System definiert sich also durch seine Requirements und ist durch diese *vollständig* charakterisiert. In der Praxis werden die Requirements normalerweise durch den System-Ingenieur in Zusammenarbeit mit dem Kunden erarbeitet. Letztendlich geht es darum Requirements zu schaffen, die das System einfach und trotzdem klar beschreiben, keine Zweifel und Ungenauigkeiten offen lassen und sich letztendlich durch „Test oder Demonstration" verifizieren lassen. In den meisten Fällen hängt die erfolgreiche Abnahme durch den Kunden (der sogenannte Acceptance Test) von dem erfolgreichen Nachweis der Einhaltung der Requirements ab. Deshalb ist es so wichtig, dass die Requirements erfüllbar sind und trotzdem den Kundenwünschen entsprechen. Diese Gratwanderung ist eine nicht zu unterschätzende Herausforderung an den System-Ingenieur.
Ein paar Regeln für Requirements:

- Ein Requirement soll kurz und prägnant sein. Lange Requirements sollten geteilt werden.
- Ein Requirement muss verifizierbar sein.
- Ein Requirement muss eindeutig sein. (Z.B. „Der Hintergrund soll eine schöne Farbe haben" ist subjektiv und daher nicht kontrollierbar.)
- Ein Requirement soll eindeutig identifizierbar sein. (Z.B. durch eine Nummer.)
- Alle Requirements zusammen ergeben eine **vollständige** Definition des Systems.

Die sogenannten „System Requirements" beschreiben das System im Ganzen, die Einzelkomponenten sind zu diesem Zeitpunkt noch nicht bekannt. Es wird eine globale Übersicht über die Funktionalität des Gesamtsystems gegeben. Hier befinden sich auch die Richtlinien zum Design.

4.1.1 Design Requirements

Design Requirements geben in der Regel Vorgaben zur Formatierung des Quellcodes.

REQUIREMENT 01
Es sollen nur Programmiertechniken verwendet werden, die auch im NUMAS System verwendet werden.

REQUIREMENT 02
Jede Quellcode Datei soll mit einem eindeutigen Header versehen werden:

```
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Dateiname
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------------
//
//    Funktion zur Auswertung
//
//--- History ------------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------------
//    Datum      SHKleemann     erstellt
//---------------------------------------------------------------------------
```

REQUIREMENT 03
Es sollen die „Style Guides“ (Programmierrichtlinien) des DV-Labors der FH-Aachen verwendet werden.

4.1.2 Funktionale Requirements

Dieses Kapitel gibt Auskunft über die Funktionalität des Systems. In diesem Fall lehnen sich die Requirements an das Kapitel „Was sollte unser System können?“ an.

REQUIREMENT 04
Das System soll aus zwei Modulen bestehen, einem Prüfungsmodul und einem Übungsmodul.

REQUIREMENT 05
Beide Module sollen eigenständig und in NUMAS integriert lauffähig sein.

REQUIREMENT 06
Beide Module sollen in einem Internet Browser aufgerufen werden.

REQUIREMENT 07
Das System soll einen Fragenkatalog besitzen aus dem sich beide Module bedienen.

REQUIREMENT 08
Der Fragenkatalog soll alle Fragen und Antworten beinhalten.

REQUIREMENT 09
Es soll ein Editor für den Fragenkatalog mitgeliefert werden.

4.2 Software Architektur

Nachdem die Beschreibung der Grundfunktionalität bekannt und auch die grobe Einteilung in zwei Module vorgegeben ist, kann mit dem Entwurf des Softwaredesigns begonnen werden.

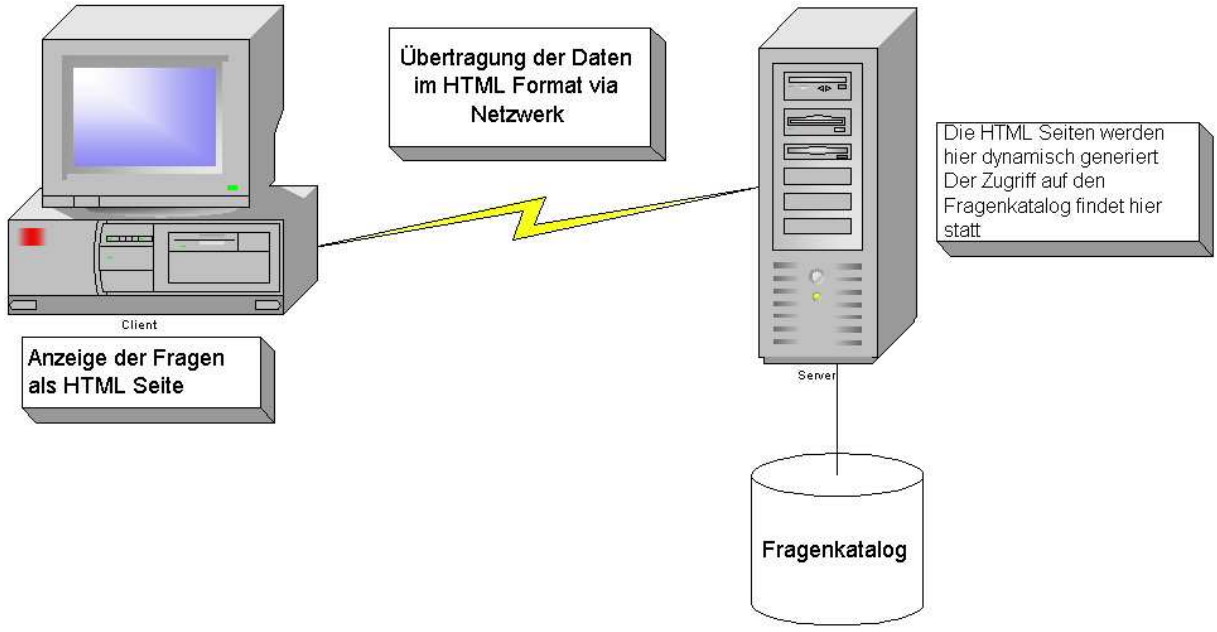

Abbildung 4.1: Überblick über das Gesamtsystem

Um einen ersten Überblick über alle für das System notwendigen Aktivitäten darzustellen, lässt sich für beide Module jeweils ein Anwendungsfall-Diagramm erstellen:

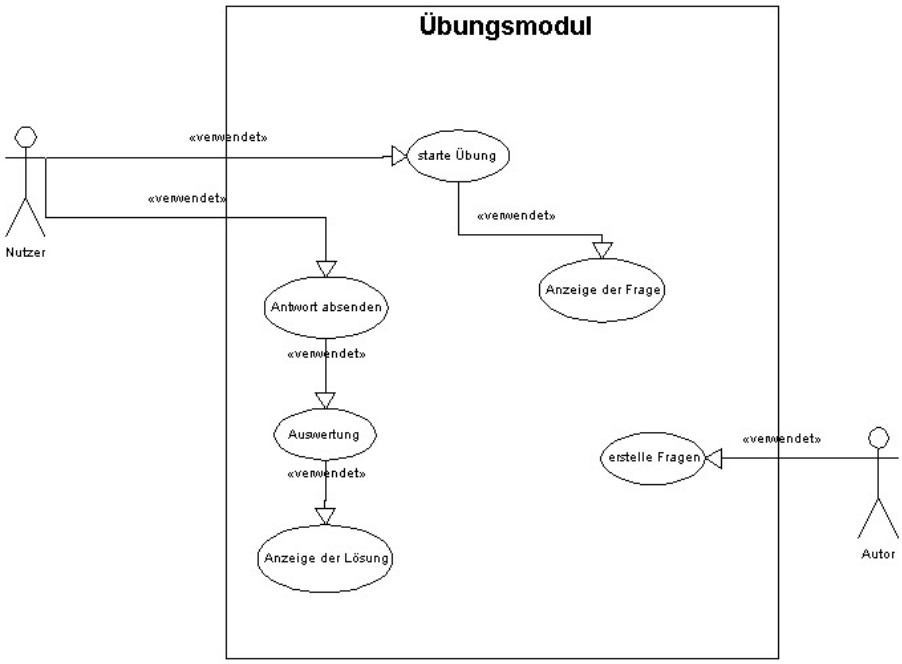

Abbildung 4.2: Anwendungsfall Diagramm Übungen

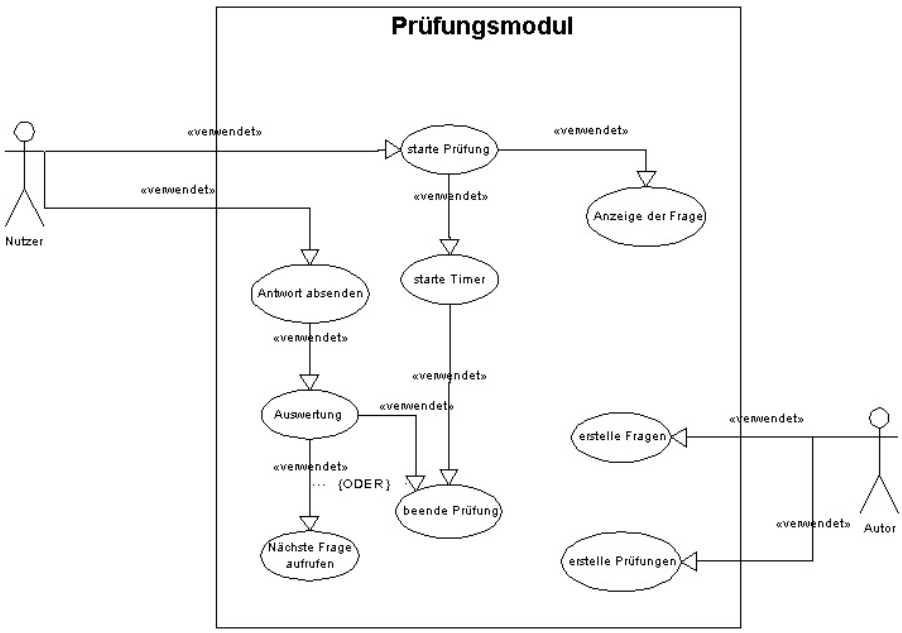

Abbildung 4.3: Anwendungsfall Diagramm Prüfungen

Aus den Diagrammen ist sofort ersichtlich, dass sich bestimmte Aktivitäten in beiden Modulen wiederholen. Es ist also sinnvoll nicht nur den Fragenkatalog, sondern auch bestimmte Prozeduren für beide Module gemeinsam nutzbar zu machen. So ist es in beiden Modulen

notwendig Fragen anzuzeigen und nach Eingabe der Antwort eine Auswertung durchzuführen.
Um eine Manipulation auszuschließen, sollte die Auswertung serverseitig geschehen. Die Antworten werden dann vom Client an den Server geschickt und dort verarbeitet. Der User hat somit keine Möglichkeit in den Prozess der Auswertung einzugreifen.
Jedes der Module start seine Funktionen aus einer HTML Seite heraus. Diese HTML Seite besteht aus einzelnen Frames, was die Flexibilität des Systems erhöht.

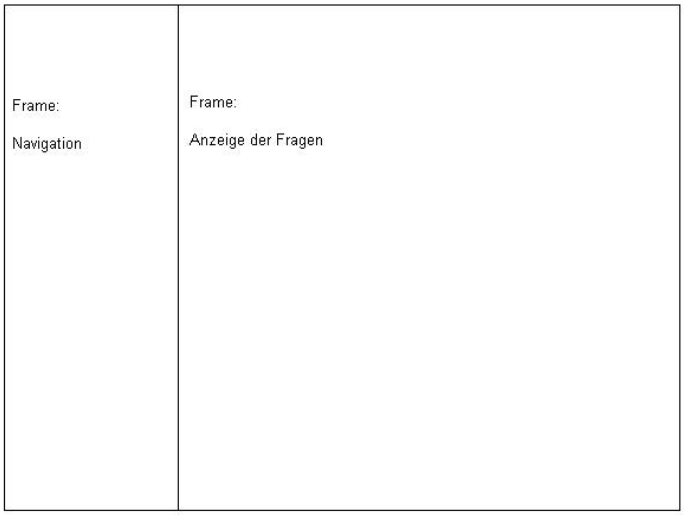

Abbildung 4.4: Frames der Startseite der Übungen

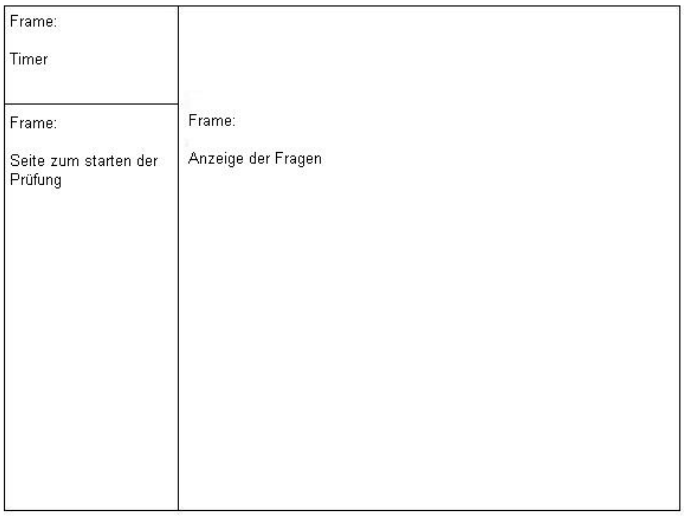

Abbildung 4.5: Frames der Startseite der Prüfungen

Die Verwendung von Frames bietet den folgenden Vorteil: Das System kann problemlos durch den Austausch der Navigationsseite (im Übungsmodul) bzw. der Prüfungsseite (im Prüfungsmodul) an die neue Umgebung angepasst werden. Im Fall der Nutzung als „eigenständiges“ Modul übernimmt der entsprechende Frame die Auflistung aller Fragen (im Übungsmodul) bzw. den Start der Prüfung. Wird das Modul in ein Gesamtsystem eingefügt, übernimmt der entsprechende Frame die Verbindung zwischen dem aufrufenden System und dem eTesting System. Der Einbau in ein anderes Lernsystem beschränkt sich also auf den Aufruf der eTesting Startseite und die Anpassung dieser einen Verbindungsseite. Es reichen wenige Quellcodezeilen aus, die Verbindung herzustellen und es ist keine Änderung am eigentlichen eTesting System notwendig. Keine notwendige, aber problemlos durchführbare Änderung, ist die Änderung des Designs. In allen Fällen bleiben die funktionalen Teile des eTesting Systems unberührt und es besteht keine Gefahr einer ungewollten Veränderung der zum Betrieb notwendigen Bereiche.
Das Anzeigen der Fragen geschieht in beiden Modulen auf die gleiche Art und Weise und vollkommen dynamisch. Das heißt, dass der abgespeicherte Inhalt geladen und nach einem vorgegebenen Design formatiert wird. Das Design wird in einer separaten Datei abgespeichert und lässt sich auch autark ändern.
Die Auswertung beginnt mit dem Absenden der Antwort durch den Nutzer und endet mit der Bereitstellung einer Auflistung, ob die Fragen der Aufgabenseite richtig oder falsch beantwortet wurden. Im Fall der Übung wird dann die richtige Lösung angezeigt und der Nutzer erhält die Möglichkeit sich zusätzliche Informationen anzusehen, wenn diese der Autor entsprechend bereitgestellt hat. Im Prüfungsmodus speichert das System die Liste und ruft die nächste Frage auf bzw. beendet nach der letzten Frage die Prüfung.
Für das eTesting System bietet es sich an, den Fragenkatalog in eine Datenbank zu speichern, wie es für die meisten Content-Management-Systeme üblich ist. Die Datenbank bietet zudem die Möglichkeit auch die Prüfungen abzulegen und eine Gruppierung der Fragen einzuführen. Es ist durch NUMAS eine Ordnung in Lernfelder und Lernobjekte vorgegeben.
Die Bestimmung der Einzelkomponenten ist einfach:

- Jedes Modul für sich wird als eigene Komponente betrachtet.
- Die beiden gemeinsam genutzten Funktionalitäten (Generierung der Aufgaben und Auswertung) werden jeweils als eigene Komponente erzeugt und stehen damit beiden Modulen zu Verfügung.
- Auch die Datenbank muss beiden Modulen zur Verfügung stehen und wird als eigene Komponente geführt.
- Der Editor für die Datenbank ist ein separater Teil des Systems und wird als einzelne Komponente geführt.

Es ergibt sich folgendes Bild:

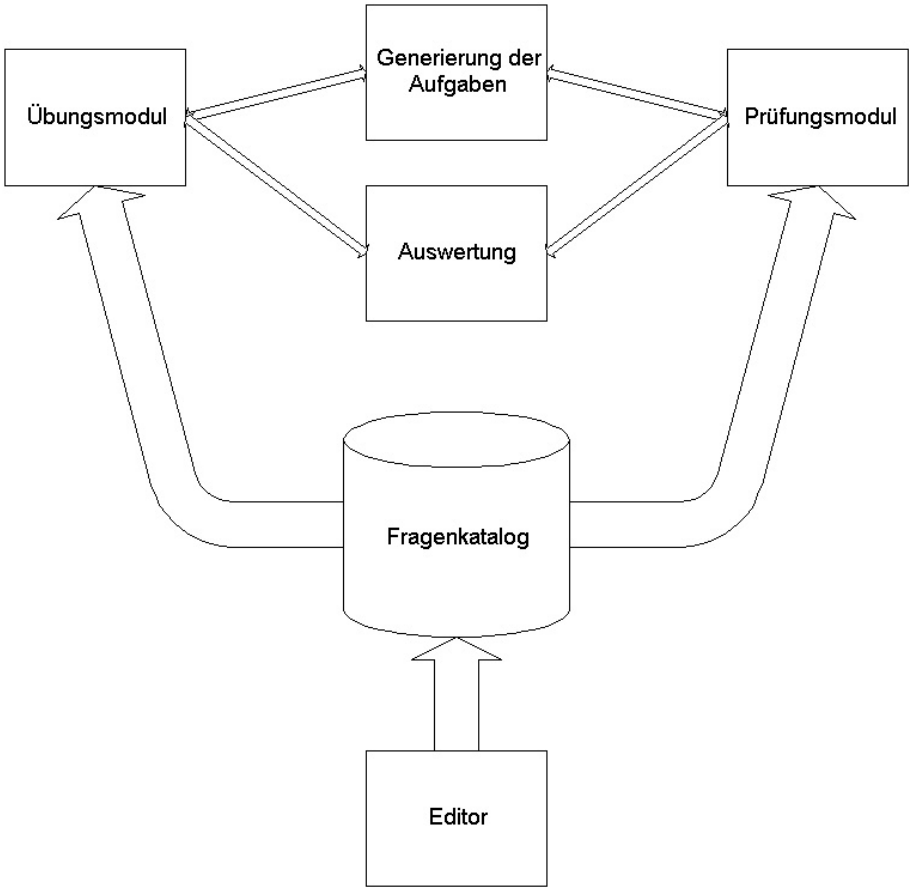

Abbildung 4.6: Zusammenspiel der Einzelkomponenten

4.3 Requirements für die Einzelkomponenten

Nachdem die Software Architektur festgelegt wurde, sind die Einzelkomponenten bekannt und die Requirements für die Einzelkomponenten können festgelegt werden. In der Praxis ist zu diesem Zeitpunkt wieder ein Treffen mit dem Kunden notwendig.

4.3.1 Übungssmodus

REQUIREMENT 10
Das Übungsmodul soll nach der Beantwortung die richtige Lösung anzeigen und Fehler farblich markieren, soweit von den Browsern unterstützt.
Anmerkung:
Nicht alle Browser unterstützen die farbliche Änderung von „input“ Feldern.

REQUIREMENT 11
Das Übungsmodul soll die Möglichkeit geben Zusätze zu den Antworten anzuzeigen.

4.3.2 Prüfungsmodus

REQUIREMENT 12
Die Userverwaltung soll von NUMAS übernommen werden. Im „eigenständigen“ Betrieb soll nur nach Name und E-Mail-Adresse gefragt werden. In diesem Modus ist keine Speicherung der Daten notwendig, er dient ausschließlich zu Testzwecken.

REQUIREMENT 13
Die Prüfungen sollen in einer MySQL Datenbank abgelegt werden.

REQUIREMENT 14
Die Prüfungen sollen mit den Lernobjekten verknüpft werden. Jeder Prüfung soll eine Anzahl an Fragen aus einem Lernobjekt zugewiesen werden.

REQUIREMENT 15
Die Fragen sollen per Zufallsgenerator ausgewählt werden.

REQUIREMENT 16
Prüfer und Prüfling sollen per E-Mail über das Ergebnis informiert werden. Außerdem soll das Ergebnis in der Benutzerverwaltung hinterlegt werden.

4.3.3 Fragenkatalog

REQUIREMENT 17
Der Fragenkatalog soll in einer MySQL Datenbank abgelegt werden.

REQUIREMENT 18
Der Fragenkatalog soll lediglich die Inhalte und nicht das Design oder eine Formatierung speichern.

REQUIREMENT 19
Es soll die Möglichkeit geben die Fragen über ein externes Style Sheet (CSS) zu formatieren

REQUIREMENT 20
Die Anzeige von Bildern in einer Frage soll möglich sein.

REQUIREMENT 21

Der Fragenkatalog soll die Fragen in Lernobjekte, diese wiederum in Lernfelder aufteilen.

REQUIREMENT 22
Der Fragenkatalog soll Fragen mit den Prüfungsverfahren „Single- und Multiple Choice“ sowie „Schlüsselwörter“ zulassen

REQUIREMENT 23
Multiple Choice Fragen sollen mit „Checkboxen“ realisiert werden.

REQUIREMENT 24
Single Choice Fragen sollen mit „Radio Buttons“ realisiert werden.

REQUIREMENT 25
Schlüsselwortabfragen sollen die Möglichkeit mehrerer Antworten haben.

REQUIREMENT 26
Schlüsselwortabfragen sollen sich als „Lücken“ in den Fragentext einfügen lassen.

4.3.4 Datenbank-Editor

REQUIREMENT 27
Der Editor soll aus dem Internet bedienbar sein.

REQUIREMENT 28
Der Editor soll die Möglichkeit geben Fragen mit Antworten hinzuzufügen.

REQUIREMENT 29
Der Editor soll die Möglichkeit geben Fragen mit Antworten zu löschen.

REQUIREMENT 30
Der Editor soll die Möglichkeit geben Zusätze zu den Antworten hinzuzufügen.

REQUIREMENT 31
Der Editor soll die Möglichkeit geben Zusätze zu den Antworten zu löschen.

REQUIREMENT 32
Der Editor soll die Möglichkeit geben Lernobjekte hinzuzufügen.

REQUIREMENT 33
Der Editor soll die Möglichkeit geben Lernobjekte zu löschen.

REQUIREMENT 34

Der Editor soll die Möglichkeit geben Lernfelder hinzuzufügen.

REQUIREMENT 35
Der Editor soll die Möglichkeit geben Lernfelder zu löschen.

REQUIREMENT 36
Der Editor soll die Möglichkeit geben Prüfungen hinzuzufügen.

REQUIREMENT 37
Der Editor soll die Möglichkeit geben Prüfungen zu löschen.

REQUIREMENT 38
Der Editor soll die Möglichkeit einer Datenbank-Validierung beinhalten.

4.4 Die Datenbank

Die Verwendung einer MySQL Datenbank ist die einzig sinnvolle Wahl, da das NUMAS System die komplette „User“ Verwaltung über die Open Source Datenbank abwickelt. Die benötigten Tabellen können in das bereits vorhandene System eingepflegt werden, außerdem ist MySQL eine moderne, unkomplizierte Datenbank die sich von den meisten Programmiersprachen aus problemlos bedienen lässt.
Die Datenbank stellt das Herzstück des Systems dar, sie beinhaltet den Fragenkatalog, die entsprechenden Antworten und die Prüfungen. Die Inhalte, mit denen Übungen gleichermaßen wie Prüfungen versorgt werden müssen, sind in der Datenbank abgelegt und werden hier verwaltet und so wird schnell klar, dass keine oder eine schlechte Datenbank das Gesamtkonzept gefährdet und eine gelungene Realisierung wesentlich von der Konzeption der Datenbank abhängt.
Deshalb ist die Auswahl der Tabellen und die Bestimmung der Tabellenfelder enorm wichtig. Während der Konzeption muss also die gesamte System-Struktur definiert sein, damit die Datenbank ausreichend genau definiert werden kann. In dieser Phase reicht es allerdings, die Struktur festzulegen. Genaue Bezeichnungen von Tabellen oder Feldern sind an dieser Stelle noch nicht notwendig.

Es wurden folgende Tabellen für das eTesting System kreiert:

1. Fragen
2. Antworten
3. Ergänzungen zu den Antworten
4. Lernobjekte

5. Lernfelder
6. Beziehung Lernfelder - Lernobjekte
7. Prüfungen
8. Beziehung Prüfungen - Lernobjekte

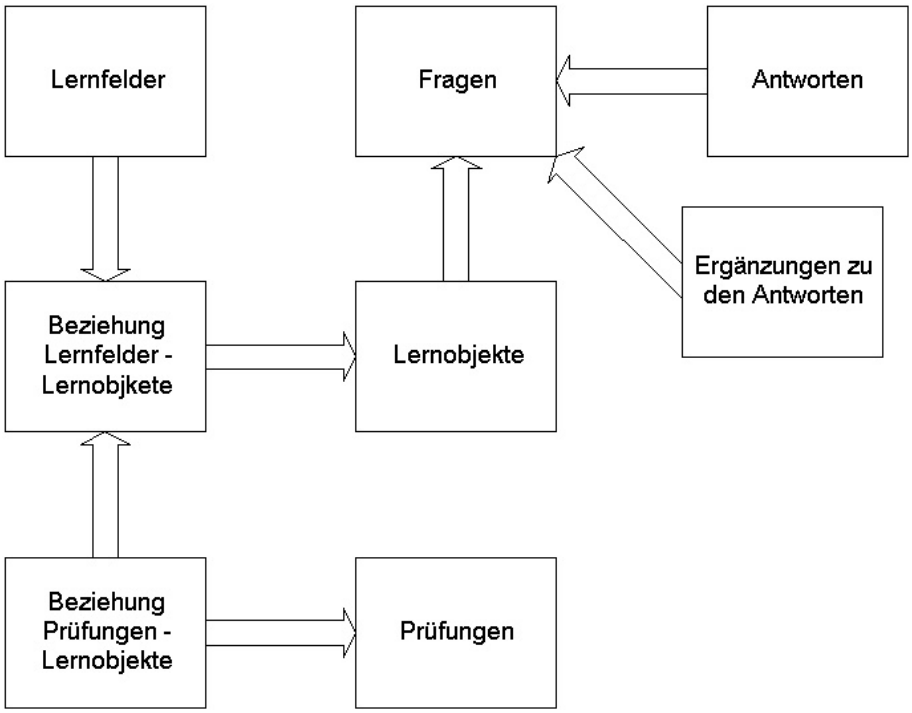

Abbildung 4.7: Zusammenspiel der Tabellen

Fragen
In dieser Tabelle werden die Inhalte der Fragen unter genau identifizierbaren Nummern abgespeichert. Die Fragen werden hier direkt mit den Lernobjekten verknüpft.
Jede Frage kann mehrere Antwortmöglichkeiten haben. Das bedeutet, dass es möglich ist Text, Frage und Antwortmöglichkeit in beliebiger Reihenfolge zu platzieren. So kann eine Frage aus Text einem Schlüsselwort, Text und weiteren Schlüsselwörtern bestehen. Letztendlich könnte in die Frage zum Beispiel noch ein „Multiple Choice“ Test eingefügt werden.
Antworten
Die Antworten werden hier nach einem genau definierten Schema abgespeichert. Jede Antwort ist genau zu einer Antwortmöglichkeit einer Frage zugeordnet. Dies geschieht über die Nummer der Frage, den Antworttyp (Schlüsselwort(SW), Multiple(MC)- bzw. Single Choice(SC)) und der Nummer des Antworttyp. Beispiel: Drittes Schlüsselwort der ersten Frage: SW3.
Die Antwort ist als Text abgelegt. Bei Schlüsselwörtern sind verschiedene Antwortmöglichkeiten mit dem Trennzeichen „;“ zu trennen. Bei Single Choice antworten wird die Nummer des richtigen Radio Buttons eingetragen. (Gezählt wird von oben nach unten). Bei Multiple

Choice Fragen wird ein Bitmuster abgelegt. In diesem Fall wird von unten nach oben gezählt. Für jedes Richtige Kästchen wird eine 1 für jedes Falsche eine 0 eingetragen.
Beispiel:
☐Antwort 1
☑Antwort 2
☑Antwort 3
würde einen Eintrag „110“ erfordern.

Ergänzungen zu den Antworten
Hier können vom Autor ergänzende Hinweise zu den Antworten (Beispielrechnung oder ähnliches) eintragen werden.

Lernobjekte
Hier werden die Lernobjekte gelistet, mit denen die Fragen verknüpft werden. Jedes Lernobjekt ist unter einer genau identifizierbaren Nummer abgespeichert.

Lernfelder
Hier werden die Lernfelder gelistet, mit denen die Fragen verknüpft werden. Jedes Lernfelder ist unter einer genau identifizierbaren Nummer abgespeichert.

Beziehung Lernfelder - Lernobjekte
Hier wird lediglich eine Angabe darüber gemacht, welche Lernobjekte zu den entsprechenden Lernfeldern gehören. Jedes Lernobjekt kann nur einem Lernfeld zugewiesen werden.

Prüfungen
In dieser Tabelle werden die verschiedenen Prüfungen abgelegt. Jede Prüfung ist unter einer genau identifizierbaren Nummer abgespeichert.
Alle relevanten Daten:

1. Name der Prüfung (des Faches)
2. Nummer der Prüfung (des Faches)
3. Name des Prüfers
4. E-Mail Adresse des Prüfers und
5. eine Zeitvorgabe

werden hier abgelegt.

Beziehung Prüfungen - Lernobjekte
Letztlich wird hier eine Angabe darüber gemacht, welche Lernobjekte zu den entsprechenden Prüfungen gehören. Es wird eine Angabe darüber gemacht, wie viele Fragen aus einem Lernobjekt in dieser Prüfung vorkommen sollen. Ein Zufallsgenerator entscheidet dann während der Prüfung über die genaue Auswahl der Fragen.

4.5 Generierung der Aufgaben

Die dynamische Gestaltung der Fragen ist ein Lösungsansatz, der sich auf zwei Anforderungen auswirkt:
Erstens soll sich der Autor lediglich mit den Inhalten und nicht mit Design oder Programmierung auseinander setzen müssen.
Zweitens soll sich das System einfach in verschiedene Lernumgebungen einfügen, dazu ist es notwendig, dass auch das Design einfach austauschbar ist.
Es hat sich in den letzten Jahren deutlich herauskristallisiert, dass die Methode Inhalte in XML-Dokumenten zu speichern als brauchbarste Lösung für Aufgaben im Content-Management Bereich anzusehen ist.
Definition:
„Die Extensible Markup Language, abgekürzt XML, ist ein Standard zur Erstellung maschinen- und menschenlesbarer Dokumente in Form einer Baumstruktur.“[6]
Für die Umwandlung von XML-Dokumenten in verschiedene Formate (HTML, PDF, usw.) mit Hilfe von XSL-Dokumenten (XML-Dokumente, welche die Formatierung beinhalten) gibt es einige Tools zur freien Benutzung. Die als XML abgespeicherten Inhalte können also von unserem System in ein HTML-Format gewandelt und entsprechend angezeigt werden. Das XML Dokument kann direkt in die Datenbank gespeichert werden.
Da Änderungen am Aussehen der Seite von den funktionalen Teilen des eTesting Systems getrennt werden sollen, wird die erzeugte HTML-Seite in Abhängigkeit zu einem CSS-Style-Sheet gestellt.
Die Generierung beschränkt sich also auf die Umwandlung eines XML-Dokuments in Verbindung mit einer Definition des Formates (XSL-Dokument) in eine HTML-Seite.

4.6 Auswertung

Nachdem die Antwort bestätigt wurde, wird die Auswertung aufgerufen. Dort wird eine Liste erzeugt, in der jede der Antworten von der Aufgabenseite mit „Richtig“ oder „Falsch“ markiert wird. Diese Liste wird dann zur weiteren Verarbeitung an die Folgeseiten übergeben. Die weitere Verarbeitung hängt vom entsprechenden Modus ab, in dem die Frage gestartet wurde: Im Übungsmodus wird die Antwort angezeigt, im Prüfungsmodus wird das Ergebnis zwischengespeichert und die nächste Frage aufgerufen bzw. nach der letzten Frage wird die Prüfung beendet.
Die Bewertung der Fragen geschieht nach folgendem Schema:
Eine „Single Choice“ Frage gilt nur dann als richtig, wenn der richtige Radiobutton markiert wurde.
Eine „Multiple Choice“ Frage gilt nur dann als richtig, wenn alle notwendigen Checkboxen markiert wurden und keine der falschen Antworten angekreuzt ist.
Für die Schlüsselwörter gilt die folgende Vorgehensweise:
Die Antwort wird komplett in Kleinbuchstaben gewandelt, so dass auf Gross- und Kleinschreibung keine Rücksicht genommen wird.
Stehen mehrere Antworten zu einer Eingabemöglichkeit zur Verfügung, werden diese durch „;“ getrennt in der Datenbank abgelegt. Sobald eines der Wörter mit dem abgegebenen Lö-

sungsvorschlag übereinstimmt, wird die Antwort als richtig gewertet.
Die Antworten müssen zu 100% richtig geschrieben werden. Es gibt keine automatische „Rechtschreibkorrektur“, die auch den Sinn der Wörter überprüft. Bei dem Zusatz von zwei Buchstaben (z.B. „un“) ergibt das Wort schon einen völlig anderen Sinn, das bedeutet, wenn man bei der Überprüfung eine Unschärfe von zwei Buchstaben erlaubt, läuft man Gefahr, dass der Sinn des genannten Wortes nicht mehr derselbe ist. Bei der Eingabe von Zahlen wird es noch kritischer, hier entscheidend jede Zahl; eine ungenaue Prüfung wäre sinnlos. Aus diesem Grund ist nur eine absolut genaue Prüfung möglich; einzige Einschränkung: Groß- und Kleinschreibung wird nicht beachtet.

4.7 Das Übungsmodul

Zeitliche Abfolge:

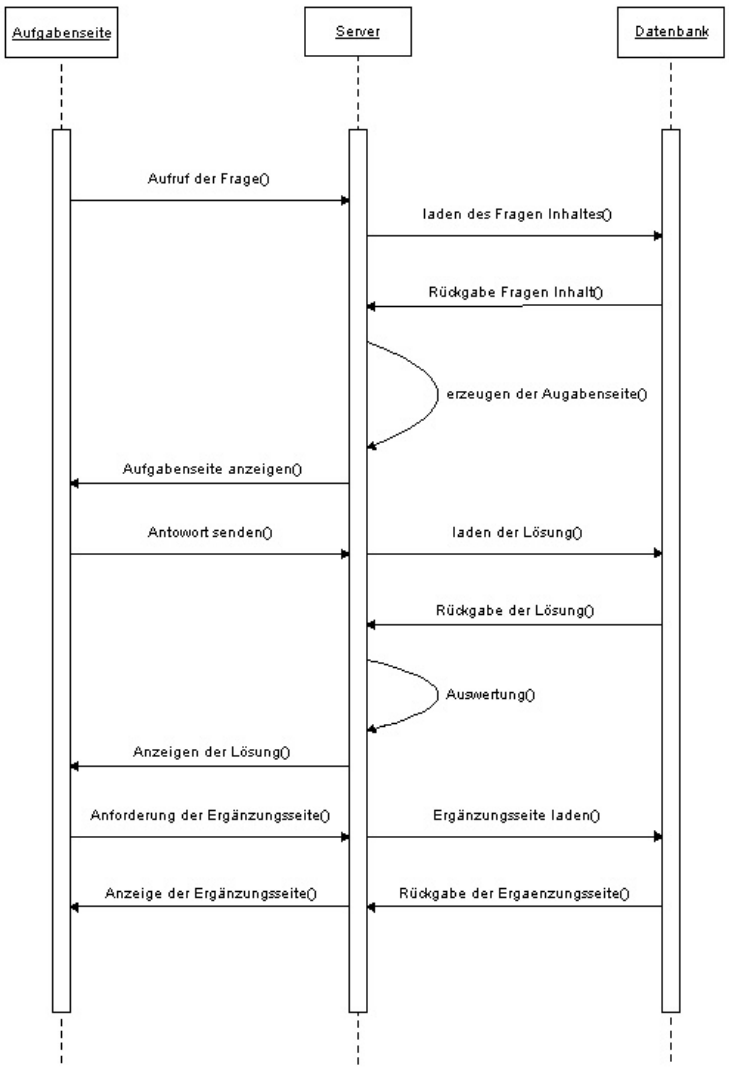

Abbildung 4.8: Zeitlicher Ablauf einer Übung

Die Homepage des linken Frames des Übungsmoduls ist für den Aufruf der Fragen zuständig. Hier soll nichts anderes geschehen, als die Seite zur Anzeige von Fragen mit dem entsprechenden Übergabeparameter im linken Frame aufzurufen und anzuzeigen. Mit Hilfe des Übergabeparameters wird dort die entsprechende Frage aus der Datenbank geladen, in HTML Format übersetzt und angezeigt. Die Aufgabenseite enthält einen „Antwort senden“ Button, mit dem die Eingaben bestätigt werden. Nach der Bestätigung wird das Modul zur Auswertung aufgerufen, das letztendlich die Anzeige der Lösung in den rechten Frame lädt. Die Lösungsseite soll sich kaum von der Aufgabenseite unterscheiden. Lediglich die folgenden

Änderungen sollen erfolgen:

- Der Button zum Absenden soll verschwinden.
- Die richtigen Lösungen werden vom Server eingetragen.
- Soweit es der Browser zulässt, sollen bei falschen Antworten farbliche Markierungen erscheinen.
- Der User soll über den Erfolg seiner Aufgabe informiert werden. Eine Angabe über die Erfolgsquote (in %) ist ausreichend.
- Falls in der Datenbank ein Hinweis oder eine Ergänzung zur Lösung hinterlegt wurde, soll diese durch einen Button aufrufbar sein.

Es kann zu jedem Zeitpunkt eine andere Frage in den Frame geladen werden.

4.8 Das Prüfungsmodul

Von der Implementierung her unterscheidet sich das Prüfungsmodul vom Übungsmodul kaum. Auch das Prüfungsmodul ist in Frames unterteilt, die Homepage des linken, unteren Frames dient hier als Schnittstelle. In dieser Schnittstelle wird die Identität des Prüflings festgestellt, sowie die Wahl der Prüfung durchgeführt. Diese Daten können natürlich auch aus dem übergeordneten Lernsystem bezogen werden. Zum Start der Prüfung lädt die Schnittstellen Homepage in den linken, unteren Frame eine Homepage, welche folgende Schritte ausführt:

1. Es wird die ausgewählte Prüfung aus der Datenbank geholt.
2. Es wird aus den entsprechenden Lernobjekten per Zufallsgenerator ein Set von Fragen erzeugt.
3. In den linken, oberen Frame wird eine Timer Homepage geladen.
4. In den rechten (Main) Frame wird nun die erste Frage geladen.

Sobald der Prüfling nun die erste Frage beantwortet hat, wird eine neue Frage in den Main Frame geladen, solange noch weitere Fragen in dem zuvor angelegten Set vorhanden sind oder die Timer Homepage die Prüfung unterbricht indem der Main Frame die Homepage zur Prüfungsauswertung erhält.
Auch nach Beantwortung der letzten Frage wird die Homepage zur Prüfungsauswertung in den Main Frame geladen.
Bei Aufruf der Prüfungsauswertung wird aus den Einzelbewertungen der Fragen eine Gesamtbewertung errechnet. Letztendlich wird das Gesamtergebnis in einer Zusammenfassung anzeigt und an den Prüfer sowie den Prüfling via E-Mail gesendet. Für den Fall, dass der

Zeitliche Abfolge:

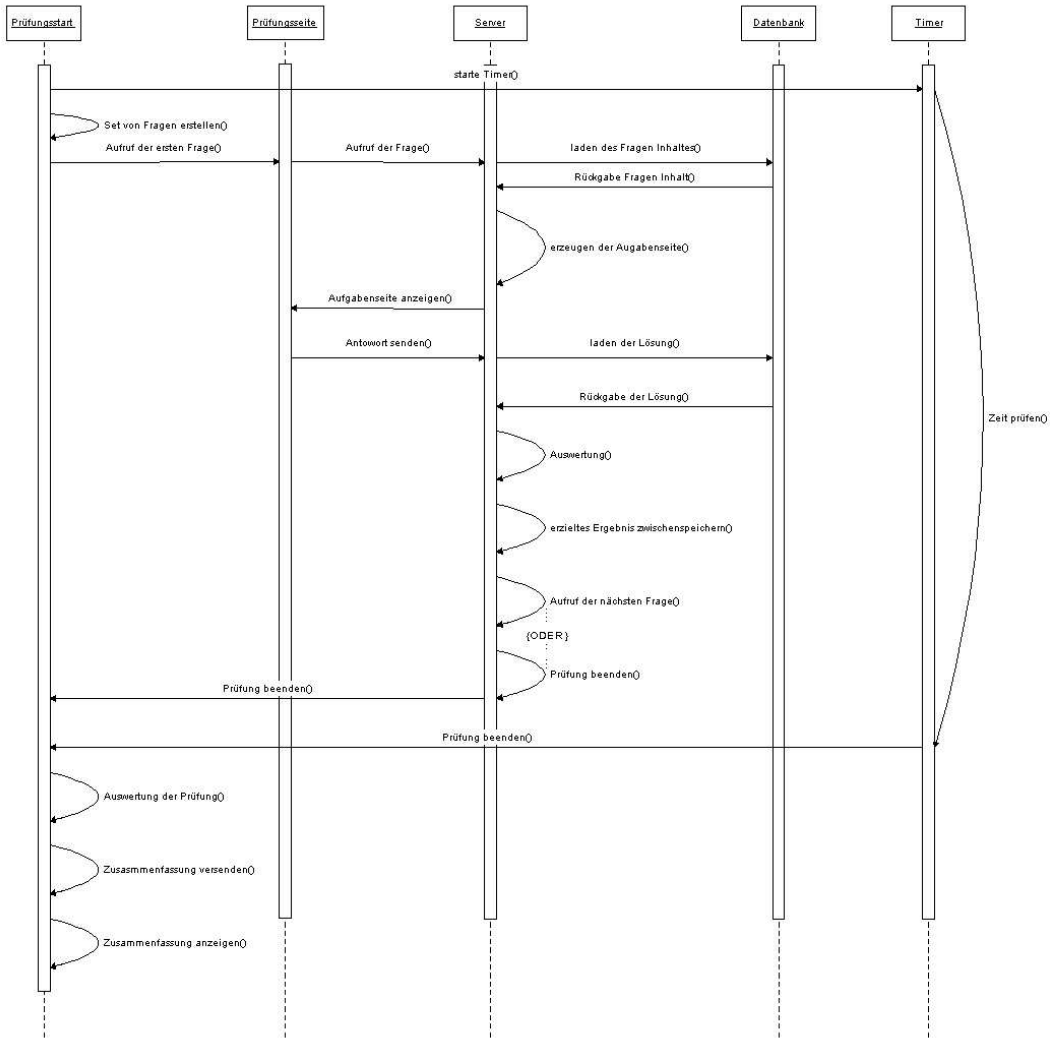

Abbildung 4.9: Zeitlicher Ablauf einer Prüfung

E-Mail Versand versagt wird die Zusammenfassung auch auf dem Server abgespeichert und steht dem Prüfer zum Abruf zur Verfügung.
Zur der Zusammenfassung gehören die folgenden Punkte:

1. Name des Prüflings
2. Name der Prüfung
3. Nummer der Prüfung
4. Datum und Uhrzeit
5. Erzieltes Gesamtergebnis
6. Verbleibende Restzeit
7. Alle Fragen
8. Die abgegebenen Lösungen

4.9 Der Datenbank - Editor

Damit die Autoren die Datenbank erweitern können, ist es zwingend erforderlich einen Editor zu haben, mit dem es problemlos möglich ist, Fragen, Antworten, Lernfelder, Lernobjekte und Prüfungen in die Datenbank einzutragen. Gerade die Inhalte der Fragen, welche im XML Format abgespeichert werden, wären ohne einen Assistenten schwer herstellbar. Die Aufgabe des Editors ist es also, den Autor bei der Änderung der Datenbank zu unterstützen. Der Editor teilt sich in fünf Bereiche:

1. Editor für die Fragen
2. Editor für die Lernobjekte
3. Editor für die Lernfelder
4. Editor für die Prüfungen
5. Datenbankvalidierung

Im Teil zur Änderung der Fragen stehen nur die Optionen „Neu anlegen“ und „Löschen“ zur Verfügung. Eine Funktion zum Ändern der Frage gibt es nicht. Eine Änderung der Frage hätte zur Folge, dass unter einer bestimmen Identifizierungsnummer eine neue Frage auftaucht, was bedeutet, dass bei einer Prüfungseinsicht dem Prüfling eventuell eine falsche Frage vorgelegt wird und der Fehler nur schwer lokalisierbar ist. Wurde die Frage komplett gelöscht ist sofort erkennbar, dass eine Änderung der Datenbank vorliegt, da auch die entsprechende Nummer fehlt. Fehlende Nummern werden in der MySQL Datenbank nicht ersetzt.
Durch die Fragenerstellung führt ein Assistent. Das heißt, dass der Autor vorgibt welches

Element an nächster Stelle eingefügt werden soll und es werden dann nacheinander die Bausteine abgefragt. Soll zum Beispiel eine Überschrift eingefügt werden, so fragt das System lediglich nach dem Text der erscheinen soll. Bei einer Frage wird der Autor nach dem Prüfungsverfahren, dem Text und den Antwortmöglichkeiten, sowie den Lösungen gefragt. Der Assistent erstellt dann aufgrund der Eingaben das XML Dokument und speichert es in der Datenbank. Der Autor wird auch nach der Zuordnung zu einem Lernobjekt gefragt. Auch diese wird mit in die Datenbank aufgenommen.
Die Option „Löschen“ entfernt die entsprechende Frage mit den dazugehörigen Lösungen und den Ergänzungen.
Für die Bereiche Lernobjekte und Lernfelder stehen die Optionen „Neu anlegen“, „Ändern“ und „Löschen“ zur Verfügung.
Es können neue Lernfelder und deren Verknüpfungen zu Lernobjekten angelegt werden. Außerdem besteht die Möglichkeit die Namen der Lernfelder sowie die Zuordnungen der Lernobjekte zu ändern. Schließlich können auch einzelne Verknüpfungen oder Lernfelder gelöscht werden. Wird ein Lernfeld gelöscht, so werden automatisch die dazugehörigen Verknüpfungen gelöscht.
Auch bei den Lernobjekten lassen sich die Namen der Lernobjekte verändern, Lernobjekte neu anlegen bzw. löschen. Wird ein Lernobjekt gelöscht werden automatisch die dazugehörigen Fragen gelöscht, da die Fragen Lernobjekten exakt zugeordnet sind.
Bei den Prüfungen sind auch die Optionen „Neu anlegen“, „Ändern“ und „Löschen“ zur Auswahl. Der Autor kann also neben der Neuerstellung einer Prüfung auch schon angelegte Prüfungen verändern. Hierzu gehört neben der Änderung der Stammdaten „Prüfungsname, Prüfungsnummer, Prüfername, Prüfer E-Mail und Prüfungszeit“ auch die Zuordnung von einer bestimmten Anzahl an Fragen aus einem Lernobjekt zu einer Prüfung. Prüfungen können selbstverständlich auch gelöscht werden.
Letztendlich gibt es noch den Bereich der Datenbankvalidierung. Um auszuschließen, dass nach einer Veränderung in der Datenbank Fehler in den Übungen oder Prüfungen auftreten, muss eine Datenbankvalidierung durchgeführt werden.
Ein Beispiel soll die Notwendigkeit unterstreichen:
Wird eine Frage gelöscht, so enthält das Lernfeld logischerweise nun eine Frage weniger. Angenommen wir hatten zehn Fragen einem bestimmten Lernobjekt zugeordnet, dann zählt dieses jetzt nur noch neun. Enthält die Datenbank aber eine Prüfung mit einer Beziehung zu diesem Lernobjekt, welche der Prüfung zehn Fragen zuweist, wäre diese Prüfung nicht mehr durchführbar. Um ähnliche Fehler zu vermeiden ist eine Validierung nach der Manipulation der Datenbank zwingend erforderlich.

5 Realisierung

Die folgenden Programmiersprachen wurden zu Realisierung verwendet:

- PHP
- JavaScript
- HTML
- Java

Als Datenbank wurde eine MySQL Datenbank verwendet.
Alle Programmiersprachen finden auch im NUMAS Projekt Verwendung, es ist also nicht notwendig bei der Implementierung in NUMAS weitere Serverpakete zu installieren.
PHP bietet sich für die Erstellung der Module an, da bei dem Aufruf der Internetseiten auf dem Server eine HTML Seite erstellt wird und reiner HTML Code an den Browser geschickt wird. Das bedeutet, dass sämtliche Funktionalität auf dem Server bleibt. Die Auswertung und Bewertung findet also auf dem Server statt und ist somit weitgehend vor Manipulation geschützt.
Die Zwischenspeicherung von Daten wurde durch so genannte Session-Variablen realisiert. Es wird beim Start der Homepage ein globaler Speicherbereich mit einer eindeutigen Identifikationsnummer auf dem Server erzeugt. Auf diesen Speicherbereich haben alle Internetseiten, die sich entsprechend identifizieren können, Zugriff. Wird der Browser auf dem Client geschlossen, also die Session beendet, wird dieser Speicher zerstört.

Es werden beim Start des eTesting Systems folgende Variablen erzeugt:

TESTTYPE	Prüfungs- oder Übungsmodus
STARTTIME	Die Start Zeit.
ENDTIME	Die errechnete Endzeit.
RESTTIME	Die verbleibende Zeit für die Prüfung.
TESTEENAME	Name des Testkandidaten
TESTEEMAIL	E-Mail Adresse Testkandidaten
TESTID	Prüfungsnummer (Identifizierung in der Datenbank)
UNANSWEREDQUESTIONS	Liste mit Fragennummern (Identifizierung in der Datenbank) die noch nicht beantwortet wurden.
ANSWEREDQUESTIONS	Liste mit Fragennummern (Identifizierung in der Datenbank) die schon beantwortet wurden.
ACHIEVMENT	Liste mit den Bewertungen der beantworteten Fragen

5.1 Allgemeine Dateien

5.1.1 Globals.php

Die „Globals.php“-Datei dient als so genannte „Properties“-Datei. In ihr befinden sich alle wichtigen Daten, die für die Funktionalität notwendig sind, sich aber ändern können. Dazu gehören die Angaben des Datenbankservers. Name des Servers und die Login-Daten.

5.1.2 Random.php

Die „Random.php“-Datei enthält eine Funktion, um eine Zufallszahl zwischen vorgegebenen Grenzen zu erzeugen.

5.1.3 FormatInput.php

Die „FormatInput.php“-Datei enthält eine Funktion um die Übergabeparameter aus dem Fragenformular in das Format zu wandeln, in dem auch die Antworten in der Datenbank abgespeichert sind. Damit wird der Vergleich zwischen den User-Eingaben und den richtigen Lösungen aus der Datenbank bedeutend einfacher.

5.1.4 EMail.php

Die „EMail.php“-Datei enthält eine Funktion, um E-Mails über den lokalen SMTP Server zu versenden.

5.2 Das XML-Format der Fragen

Der Inhalt der Fragen wird in ein XML Dokument gespeichert. Das Dokument darf folgende Tags enthalten:

```
<aufgabe>
    <ueberschrift>
      Text, der als Überschrift erscheinen soll.
    </ueberschrift>
    <text>
      Text, der als Text erscheinen soll.
    </text>
    <bild img="Name des Bildes"
     width="Breite des Bildes" height="Höhe des Bildes">
    </bild>
    <formular type=„TYP">
        <group>
          z.B. SC1
        </group>
        <value>
          erzeugt bei Radio Buttons und Checkboxen ein Inputfeld mit Text
        </value>
    </formular>
</aufgabe>
```

Anmerkungen:
TYP: radio, checkbox, schluesselwort

ueberschrift	Erzeugt eine Überschrift
text	Erzeugt Fließtext
bild	Bindet ein mit Bild vorgegebener Breite und Höhe ein
formular	Erzeugt Eingabefelder: - Radiobuttons mit Text oder - Checkboxen mit Text oder - Eingabefelder für Schlüsselwörter

Das XSL-Dokument enthält für jedes der aufgeführten Tags eine Regel, um den entsprechenden HTML-Code zu erzeugen. Jedes Tag wird also in HTML-Code umgesetzt und erzeugt ein Element in der HTML-Seite.

5.3 Die Datenbank

Die Tabellenstruktur wurde während der Konzipierungsphase definiert. Der endgültige, detaillierte Entwurf der Datenbank findet während der Realisierung statt, wenn die komplette Funktionalität des Systems bekannt ist.
Die Variablen innerhalb der Tabellen wurden folgendermaßen definiert:

fragen		
FragenID	int	Automatisch generierte Identifikationsnummer
FrageXML	text	Inhalt der Frage im XML Format
fLernObjektID	int	Zuordnung zu einem Lernobjekt (Identifikationsnummer)
antworten		
aFragenID	int	Zugehörigkeit zu der Frage (Identifikationsnummer)
aTyp	char(2)	Typ des Prüfungsverfahrens (SC, MC, SW)
aNummer	int	Nummer der
Antwort	text	Antwort SC: Nummer des richtigen Radiobuttons MC: Bitmuster, siehe Kapitel Konzeption SW: Lösungen. Trennzeichen „;“
antwortenergaenzung		
aeFragenID	int	Zugehörigkeit zu der Frage
AntwortHTML	text	Link zur Ergänzungsseite
lernobjekte		
LernObjektID	int	Automatisch generierte Identifikationsnummer
LernObjekt	text	Name des Lernobjekts
lernfelder		
LernFeldID	int	Automatisch generierte Identifikationsnummer
LernFeldName	text	Name des Lernfeldes
beziehung_lf_lo		
bll_LernFeldID	int	Lernfeld (Identifikationsnummer)
bll_LernObjektID	int	Lernobjekt (Identifikationsnummer)
pruefungen		
PruefungsID	int	Automatisch generierte Identifikationsnummer
Pruefungsname	text	Name der Prüfung bzw. des Fachs
Fachnummer	text	Fachnummer
Pruefername	text	Name des Prüfers
Prueferemail	text	E-Mail Adresse des Prüfers
Pruefungszeit	text	Prüfungszeit
beziehung_lo_pruefung		
bePruefungsID	int	Prüfung (Identifikationsnummer)
beLernObjektID	int	Lernobjekt aus dem
AnzahlderFragen	int	die Anzahl von Fragen ausgewählt wird.

Die Tabellenstruktur wurde exportiert und als SQL Befehle in einer Textdatei abgespeichert. Mit dieser Textdatei können die Tabellen jederzeit wieder hergestellt werden. Die Textdatei ist auf der beliegenden CD unter dem Namen „Datenbankstruktur.txt“ zu finden.

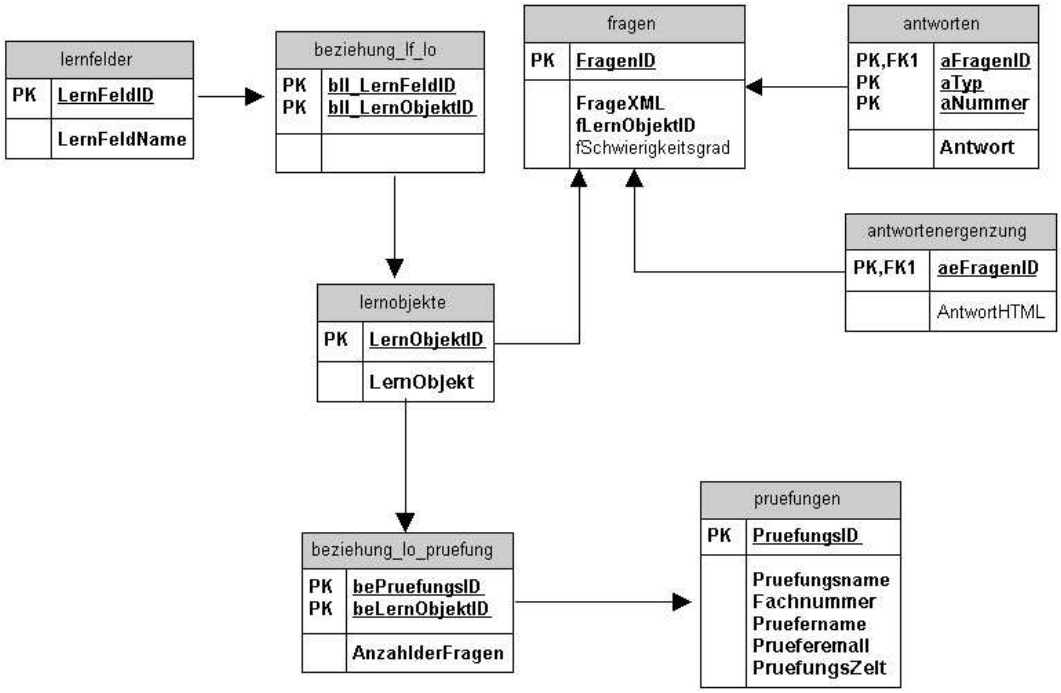

Abbildung 5.1: Datenbankmodell

5.4 Generierung der Aufgaben

Die Seite „Fragen.php“ erzeugt die Aufgabenseite und zeigt diese an. Die PHP Seite erwartet den folgenden Übergabeparameter: FragenID. Der Parameter identifiziert die entsprechende Frage anhand der Identifikationsnummer aus der Datenbanktabelle „fragen“. Wird kein Parameter angegeben wird lediglich „Keine Frage ausgewählt.“ angezeigt. Anhand der Identifikationsnummer wird der XML-Inhalt aus der Tabelle geladen. Zusammen mit dem XSL-Dokument „aufgabenstellung.xsl“ wird die Aufgabenseite produziert. Für die Erzeugung dieser Seite wird die Sablotron-Bibliothek benutzt. Die Bibliothek erhält man auf „http://www.gingerall.com“ und sie kann kostenfrei benutzt werden. Die Bibliothek lässt sich ohne Probleme in die PHP Umgebung integrieren und die bereitgestellten PHP Funktionen zur Erzeugung von HTML-Seiten sind einfach zu bedienen.
Die erzeugte HTML-Seite enthält ein Formular, in dem die entsprechenden Eingabefelder, Radio Buttons und Checkboxes erzeugt werden. Das Formular wird mit einem “Antwort absenden“ Button versehen. Die Eingaben aus dem Formular werden nach der Bestätigung an das „Auswertung.php“ gesendet. Als Übertragungsart wurde die POST-Methode gewählt.

5.5 Auswertung

Die Seite „Auswertung.php“ wird von der Fragen.php aufgerufen und erhält von dieser die Formulardaten zur Auswertung. Die Identifikationsnummer der Frage ist im Formular ent-

halten. Eine Frage kann mehrere Antworten zugewiesen haben. Das kommt aus der Tatsache, dass Fragen mehrere Prüfungsverfahren hintereinander anwenden können. So könnte zum Beispiel eine Frage aus einer Multiple Choice Auswahl, einem Schlüsselwort und einer weiteren Multiple Choice Auswahl bestehen. Die Identifizierung geschieht nach folgendem Schema: aus dem Typ des Prüfungsverfahren (SC, MC und SW) und einer fortlaufenden Nummer. Für das Beispiel gilt also: MC1, SW1, MC2. Die Antwort für die erste Multiple Choice Auswahl findet man also in der Datenbank unter dem Prüfungsverfahren „MC“ in Verbindung mit der Identifikationsnummer der Frage, der Nummer „1“.
Die Antworten für alle Auswahlmöglichkeiten der Fragen werden der Reihe nach aus der Datenbank geladen und mit den Eingaben des Users verglichen. Es wird ein Array mit dem Namen „$Ergebnis“ erstellt. In dieses Array wird die Angabe „TRUE“ bzw. „FALSE“ in Verbindung mit dem Typ des Prüfungsverfahren und der fortlaufenden Nummer gespeichert.
Die Session Variable „TESTTYPE“ gibt an, ob die Funktion „zeigeLoesungUebung ($Ergebnis, $FID)“ der Datei „Loesung_Uebung.php“ oder die Funktion „Auswertung ($Ergebnis, $FID)“ der Datei „Loesung_Pruefung.php“ aufgerufen wird. In beiden Fällen werden die gleichen Parameter übergeben: Das erzeugte Array „Ergebnis“ und die Identifikationsnummer der Frage „FID“.

5.6 Das Übungsmodul

Das Übungsmodul benötigt die folgenden Dateien:

- index_Uebungen.html
- Auswertung.php
- FormatInput.php
- Fragen.php
- Globals.php
- Loesung_Uebung.php
- Navigation.php
- aufgabenstellung.xsl
- loesung.xsl

Die „index_Uebungen.html“ erzeugt die beiden Frames der Hauptseite. In den linken, schmalen Frame wird die „Navigation.php“ geladen. Im Demonstrationsmodus, das heißt das System wurde nicht in ein Lernsystem implementiert, listet die „Navigation.php“ alle Fragen auf. Die Fragen werden entsprechend der Gruppierungen Lernfelder und der Untergruppen Lernobjekte sortiert. Die Fragen werden als Links angezeigt, bei der Aktivierung durch den

User wird in den rechten, großen Frame (Main Frame) die „Fragen.php“ geladen. Nach dem Absenden des Formulars wird in der durch die „Auswertung.php“ die Funktion „zeigeLoesungUebung“ der Datei „Loesung_Uebung.php“ aufgrufen. In der Funktion „zeigeLoesungUebung“ wird wiederum aus dem Inhalt der Frage (XML-Format) und dem „loesung.xsl“ eine HTML-Seite erzeugt. Das Formular enthält diesmal keinen Button zum Absenden der Antwort. Die richtigen Antworten werden via Javascript in das Formular eingetragen. Bei den Eingabefeldern, die durch den User falsch beantwortet wurden, wird die Hintergrundfarbe via Javascript geändert.
Anmerkung:
Das Ändern der Hintergrundfarbe funktioniert nur im MS Internet Explorer. Alle anderen Internet Browser unterstützen diese Funktionalität nicht.
Außerdem wird die Tabelle der Datenbank „antwortenergaenzung“ nach der aktuellen Fragen-Identifikationsnummer durchsucht. Wird ein entsprechender Eintrag gefunden, so wird ein Querverweis zu der angegebenen Homepage angezeigt.

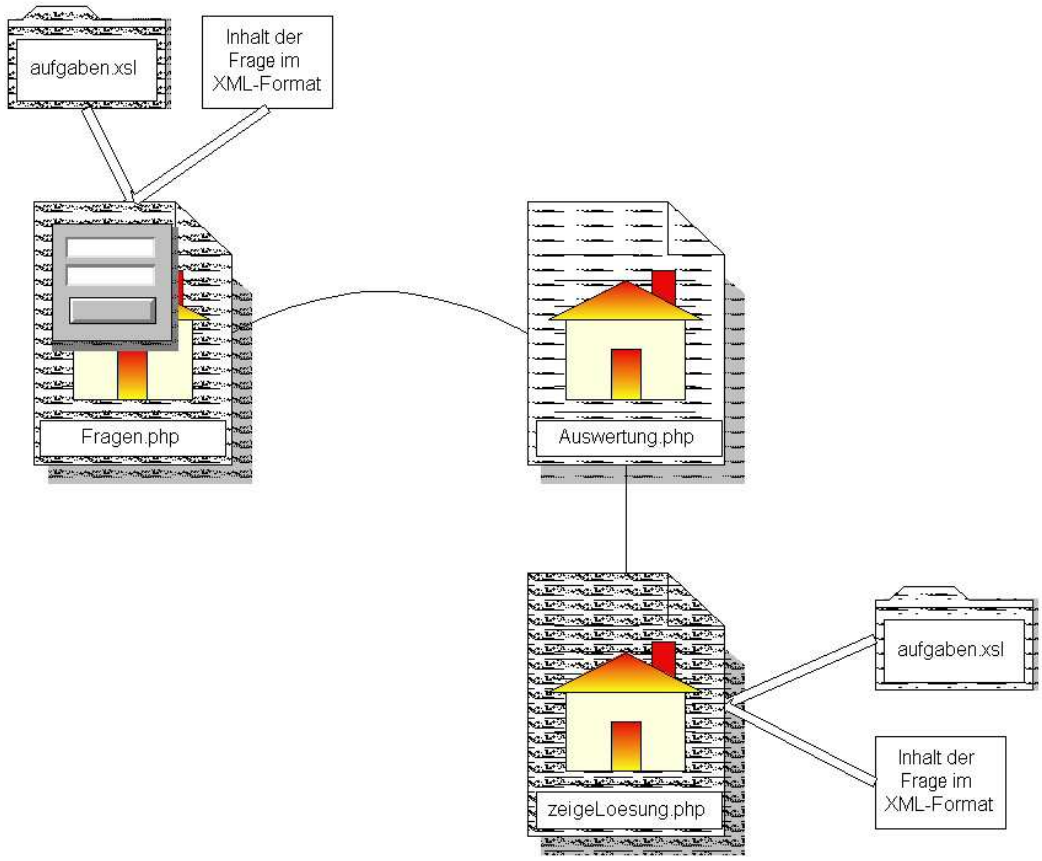

Abbildung 5.2: Aufbau des Main Frame der Übung

5.7 Das Prüfungsmodul

Das Prüfungsmodul benötigt die folgenden Dateien:

- index_Pruefung.html
- leer.html
- Auswertung.php
- Email.php
- Erfolgsmeldung.php
- FormatInput.php
- Fragen.php
- Globals.php
- Loesung_Pruefung.php
- Pruefung.php
- Random.php
- StartePruefung.php
- timer.php
- aufgabenstellung.xsl

Nach dem Aufruf der Hauptseite „index_Pruefung.html“ ist das Fenster in drei Frames geteilt. Der rechte Frame (Main Frame) und der linke, obere Frame (Timer Frame) sind mit der „leer.html“ gefüllt. Im Demonstrationsmodus ist im linken, unteren Frame (Prüfungs Frame) die „Pruefung.php“ Homepage geladen. Dort werden alle verfügbaren Prüfungen gelistet. Nach der Auswahl einer Prüfung, Eingabe des Namens und der Email-Adresse kann die Prüfung gestartet werden. Jetzt im Prüfungs Frame die Seite „StartePruefung.php“ geladen. Als erstes wird eine Auswahl an Fragen ermittelt. Die Identifikationsnummern der Fragen werden in der Session Variable „UNANSWEREDQUESTIONS“ gespeichert. Die Session Variable „ANSWEREDQUESTIONS“ und die Session Variable „ACHIEVEMENT“ bleiben leer. Die Session Variablen „TESTTYPE“, „TESTEENAME“, „TESTEEMAIL“ werden entsprechend der Eingaben gefüllt. Die Session Variable „TESTID“ wird mit der Identifikationsnummer der Prüfung gefüllt, in den Timer Frame wird die „timer.php“ Seite geladen, in den Main Frame wird die Fragen.php“ Seite geladen.
Die „Timer.php“ wird alle zehn Sekunden neu geladen und berechnet jedes mal die verbleibende Restzeit neu. Nach Ablauf der Zeit wird in den Main Frame die Seite „Erfolgsmeldung.php“ geladen. Der Timer Frame erhält wieder die Seite „leer.html“.

Der Main Frame zeigt die aktuelle Frage an. Nach dem Absenden des Fragen-Formulars wird, wie auch schon bei den Übungen, die „Auswertung.php“ aufgerufen, aus der, wie oben beschrieben, letztendlich die Funktion „Auswertung“ der „Loesung_Pruefung.php“ aufgerufen wird. Diese streicht die Frage aus der Session Variable „UNANSWEREDQUESTIONS“ und trägt die Frage in die Session Variable „ANSWEREDQUESTIONS“ ein. In die Session Variable „ACHIEVEMENT wird das Ergebnis der Frage gespeichert. Sind noch weitere Fragen in der Session Variable „UNANSWEREDQUESTIONS“ vorhanden, so wird die nächste Frage aufgerufen. Ist keine mehr vorhanden wird die Seite „Erfolgsmeldung.php“ geladen. Diese Seite berechnet den Gesamterfolg der Prüfung und zeigt das Ergebnis an. Diese Angaben werden auch per E-Mail an den Prüfer und den Studenten geschickt.

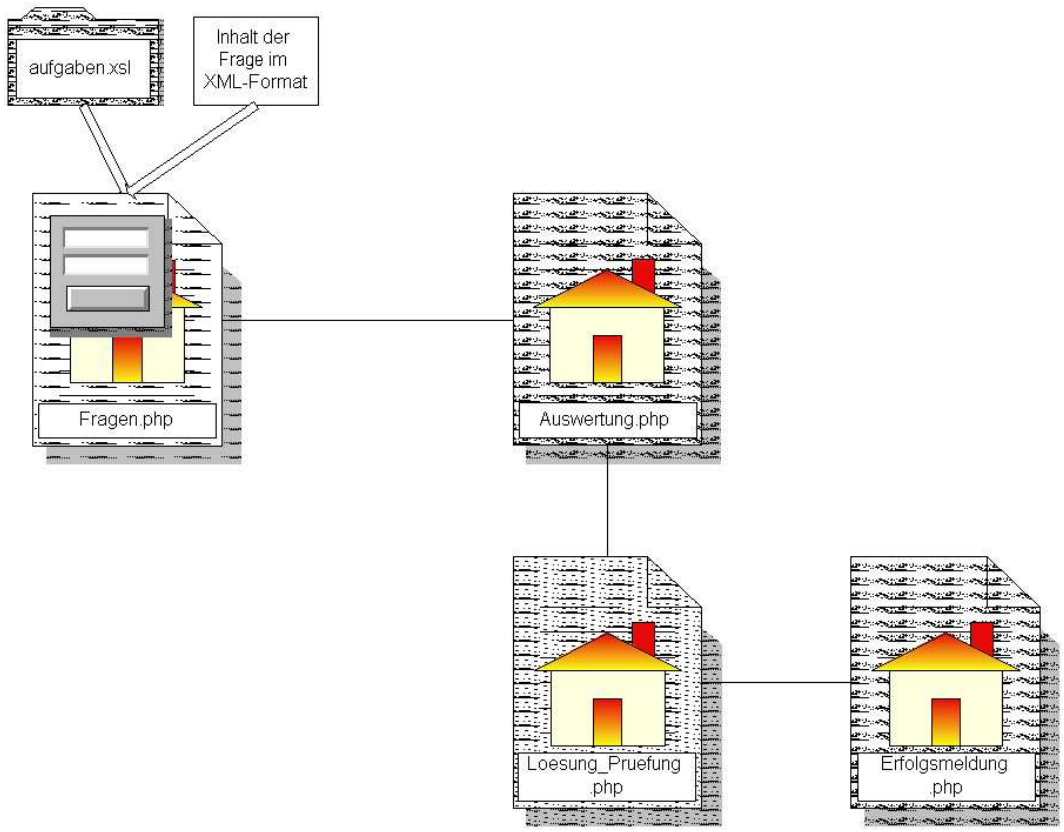

Abbildung 5.3: Aufbau des Main Frame der Prüfung

5.8 Der Datenbank - Editor

Der Datenbank Editor wurde als JAVA Applet realisiert. Somit kann der Editor über das Internet in einem Browserfenster geöffnet werden oder auch lokal direkt am System betrieben werden. Als Entwicklungsumgebung wurde eclipse 3.1 benutzt.
Der Editor besteht aus folgenden Dateien:

- CGenericDialog.java
- eTestingEditor.java
- MsgBox.java
- Panel_AddFields.java
- Panel_AddQuests.java
- Panel_AddTests.java
- Panel_AddThemes.java
- Panel_ValidateDatabase.java
- Properties.java
- SQLConnection.java

Die Erzeugung des Applets findet in der Datei „eTestingEditor.java“ statt. Jede der fünf Bereiche wurde auf eine eigene Seite (Panel) gelegt, welche sich hinter einem der fünf Tabreiter der Hauptseite verbirgt. Für jedes dieser Panels wurde eine eigene JAVA Datei erzeugt.
Die Datei „MsgBox.java“ enthält eine Klasse zur Erzeugung einer Message Box.
Die Datei „CGenericDialog.java“ enthält eine Klasse zur Erzeugung eines Dialogfeldes.
Die Datei „SQLConnection.java“ enthält eine Klasse mit der die Benutzung der Datenbank realisiert wird. Die Verbindungsdaten sind in der Datei „Properties.java“ gespeichert. Für die Datenbankverbindung nützt die Klasse „SQLConnection“ die Bibliothek „mysql-connector-java-3.1.12-bin“ der Firma Sun Microsystems. Die Bibliothek darf kostenlos benutzt werden.

6 Acceptance Test

In der Praxis schließt das Projekt mit dem so genannten Acceptance Test ab. Hier prüft der Kunde, ob die volle Funktionalität vorhanden ist und alle Requirements erfüllt wurden. Die erfolgreiche Abnahme stützt sich in der Regel auf diesen Test. Die Testumbebung muss vorher mit dem Kunden abgesprochen werden.
Die Testumgebung wurde für das eTesting-System wie folgt festgelegt:
Auf einen Windows XP PC wurde mit Hilfe des XAMPP Paketes in einen Webserver, eine PHP Engine, sowie eine MySQL Datenbank aufgespielt. Das XAMPP Paket ist eine Distribution des Apache Servers, der PHP Engine, einer MySQL Datenbank und einer Perl Engine. Das XAMPP Projekt ist ein Projekt der Organization „apachefriends“ die im Internet unter der Adresse „www.apachefriends.org“ erreichbar ist. Es wurde die XAMPP Version 2.1 verwendet.
Die PHP Engine von XAMPP beinhaltet bereits den Sablotron Parser.
Auch ein E-Mail Server ist im XAMPP Paket enthalten. Diesem wurden die E-Mail Adressen „pruefer@localhost“ und „pruefling@localhost“ hinzugefügt, um die E-Mail Funktionalität des eTesting Systems nachzuweisen.
Außerdem wurde das Java Runtime Environment in der Version 1.4.2.09 installiert.
Weitere Modifikationen des PC waren nicht notwendig.
Für alle Tests wurde der MS Internet Explorer 6 verwendet.
Für den Test wurden die Testfragen aus dem Lernfeld Quadratur des aktuellen NUMAS Systems (Stand Dezember 2005) in die Datenbank übertragen. Außerdem wurden neue Fragen von Prof. Dr.rer.nat. Wilhelm Hanrath entwickelt, um die Prüfungsverfahren „Lückentexte“ und „Single-Choice“ abzudecken. Diese Fragen sowie die dazugehörigen Lernfeld und Lernobjekt Namen wurden in die Datenbank aufgenommen. Letztendlich wurden noch zwei Prüfungen in die Datenbank eingetragen. Alle Eintragungen wurden mit dem Datenbank Editor vorgenommen.
Als Design wurde das NUMAS Design verwendet.
Die beiden Prüfungen wurden lokal auf dem System durchgeführt. Die erste Prüfung wurde mit der letzten Frage abgeschlossen, die zweite Prüfung durch den Ablauf der Zeit beendet. Es wurde darauf geachtet, dass in beiden Fällen richtig ausgewertet und das Gesamtergebnis präsentiert wurde. Es wurde der E-Mail Versand kontrolliert: Die E-Mails wurden vom Mailserver des XAMPP Systems empfangen.
Die Übungen wurden über das Netzwerk ausgeführt. Nach der Beantwortung der Fragen wurde besonders auf die Darstellung der Lösung geachtet. Die Darstellung der Lösung durfte sich nicht von der Darstellung der Frage unterscheiden. Es wurde geprüft, dass in Fällen mit Antwortergänzungen diese auch angeboten wurden.
Die beiden Module wurden nur als eigenständige Module getestet. Auf den Integrationstest in NUMAS wurde verzichtet.

7 Dokumentation

7.1 Datenbank Editor

Nach jeder Änderung in der Datenbank **muss** eine Validierung der Datenbank durchgeführt werden!
Der Editor wurde als Assistent realisiert. Das bedeutet, der Autor wird durch die Erstellung geführt und ihm wird das fertige Ergebnis zum Schluss präsentiert. Hierzu wurde das Applet in fünf Bereiche eingeteilt:

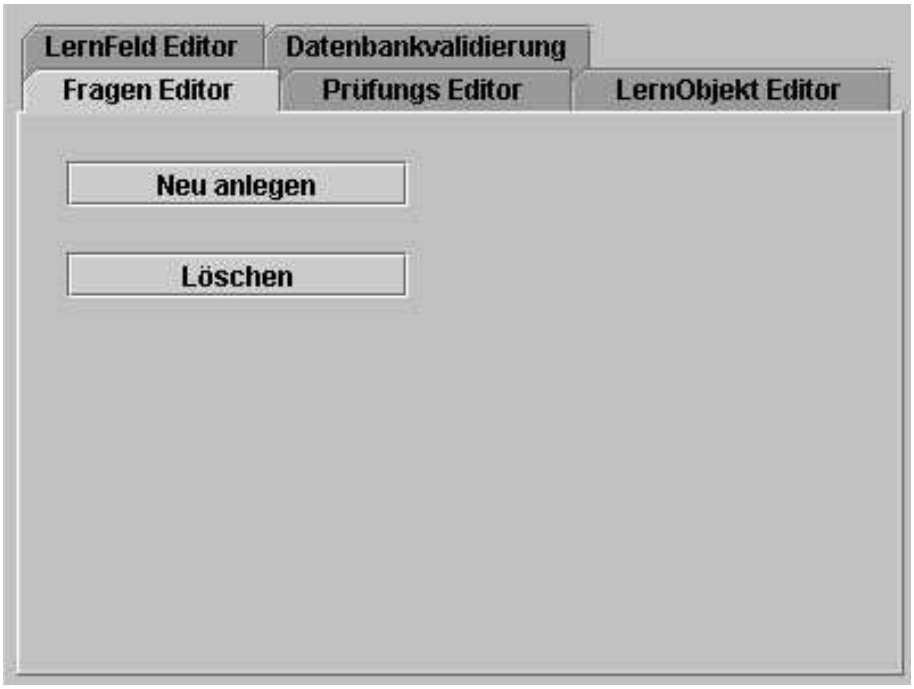

Abbildung 7.1: Der Datenbank Editor

Die drei Bereiche „LernFeld, LernObjekt und Prüfungs Editor“ listen jeweils die Einträge der Datenbank und haben die Möglichkeit das ausgewählte Element zu verändern bzw. zu löschen. Ebenfalls steht die Möglichkeit ein neues Element einzufügen zur Verfügung.
Bei den Lernobjekten und Lernfeldern wird für die Änderungsfunktionen ein Dialogfeld mit einem Eingabefeld geöffnet. Die Änderung kann direkt im Eingabefeld vorgenommen werden. Mit der Bestätigung durch den „OK“-Button wird die Änderung in die Datenbank übernommen. Auch bei der Auswahl einen neuen Eintrag zu erstellen erscheint ein Dialogfeld mit einem Eingabefeld. Der Name des neuen Lernfeldes, -objekts wird in das Eingabefeld eingetragen und mit mit der Bestätigung durch den „OK“-Button in die Datenbank aufgenommen.

Bei der Löschen-Funktion wird lediglich noch einmal die Bestätigung zum endgültigen Löschen eingefordert. Danach werden die Elemente aus der Datenbank gelöscht.
Wird ein Lernfeld gelöscht, so werden alle mit ihm verknüpften Lernobjekte auch gelöscht! Wird ein Lernobjekt gelöscht, so werden alle ihm zugeteilten Fragen und die entsprechenden Antworten mit einer eventuell vorhandenen Antwortergänzung gelöscht!

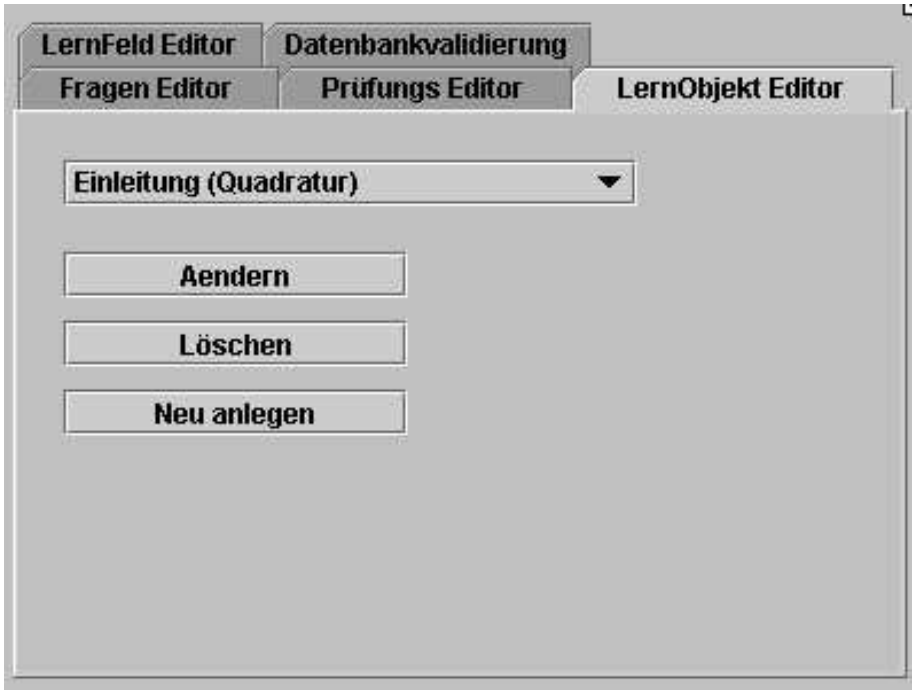

Abbildung 7.2: Mainpage Lernobjekt Editor

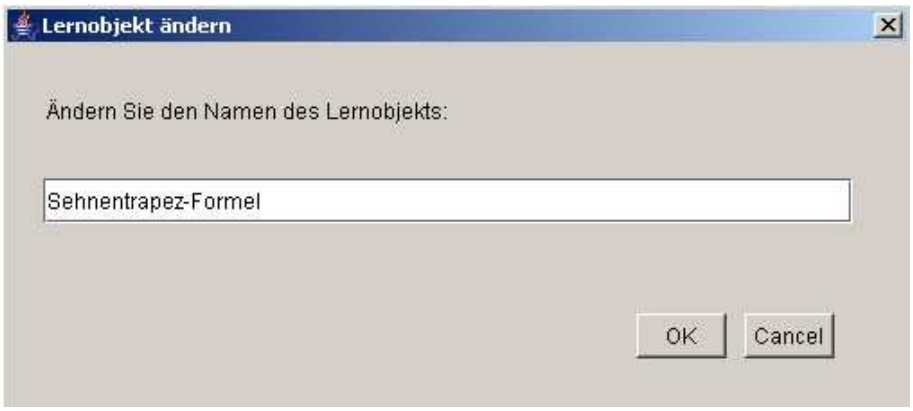

Abbildung 7.3: Der Datenbank Editor

Das Dialogfeld zur Erstellung bzw. Änderung einer Prüfung enthält zusätzlich zum Namen der Prüfung auch noch Eingabefelder für die Fachnummer, den Namen des Prüfers, die E-Mail-Adresse des Prüfers und die Prüfungszeit. Die zugeteilten Lernobjekte werden in einer Liste zur Auswahl gestellt.

Der Fragen Editor enthält lediglich zwei Möglichkeiten: Neue Fragen können erstellt werden, vorhandene Fragen können gelöscht werden. Die Auswahlbox enthält alle vorhandenen Fragen der Datenbank. Nach der Auswahl einer Frage kann diese gelöscht werden. Es erscheint noch ein Dialogfeld zur Bestätigung, dann wird die Frage mit der entsprechenden Antwort und einer eventuell vorhandenen Antwortergänzung aus der Datenbank gelöscht. Für die Erstellung einerFrage erscheint folgendes Dialogefeld:

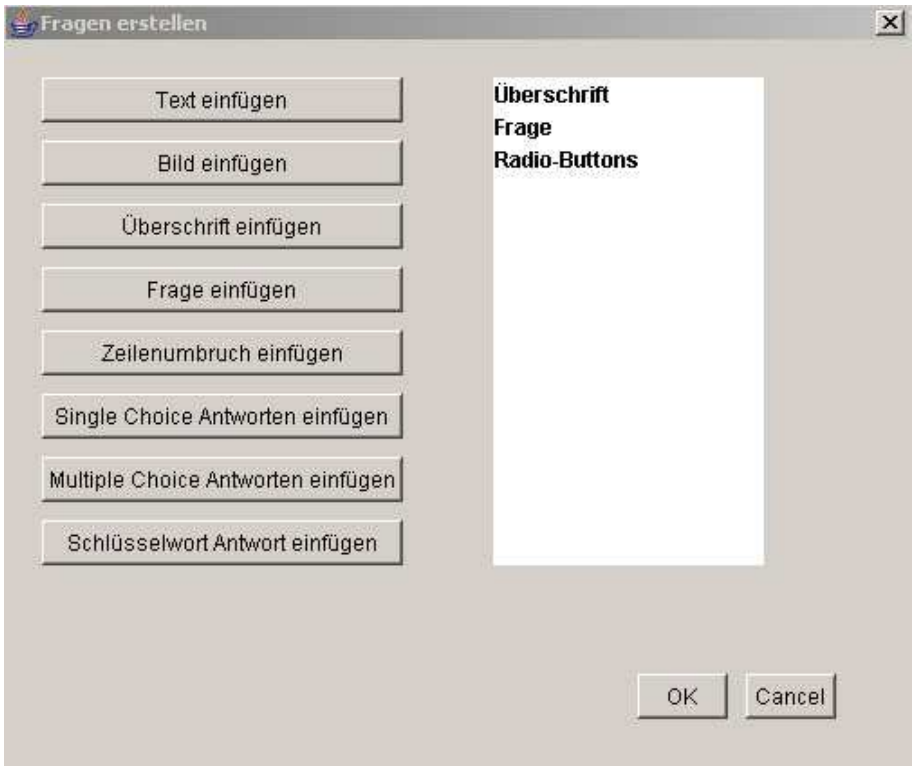

Abbildung 7.4: Dialogfeld Fragenerstellung

Mit Aufruf des Dialogfensters wird eine neue Frage erzeugt. Dieser Frage können nun nacheinander die angegebenen Elemente in beliebiger Reihenfolge zugefügt werden. Die folgenden Elemente stehen zur Verfügung:

TEXT	Text
Bild	Bild mit Angabe der Größe
Frage	Text
Überschrift	Text
Radiobuttons	Radiobutton mit Text
Checkboxen	Checkboxen mit Text
Schlüsselwort	Eingabefeld
Zeilenumbruch	manueller Zeilenumbruch

In dem Textfeld auf der rechten Seite wird angezeigt, welche Elemente der Frage schon hinzugefügt wurden. Bevor die Frage durch Bestätigung des „OK“-Buttons in die Datenbank

aufgenommen wird kann sich der Autor das Ergebnis als HTML-Seite anzeigen lassen.
Die Datenbankvalidierung wird durch den Button „Validierung starten“ gestartet. Das Ergebnis der Prüfung wird in dem Statusfenster auf der rechten Seite angezeigt.

8 Ausblick

8.1 WYSIWYG - Editor

Der mitgelieferte Editor ist in Stil eines Assistenten designed. Es ist durchaus denkbar, dass für die Erstellung der Fragen ein „What you see is what you get“ Editor erstellt wird. Die Methode des Assistenten hat den deutlichen Nachteil, dass der Autor nicht angezeigt bekommt, wie sich letztlich die Frage präsentiert. Man ist es aus der Windows Benutzung eigentlich gewohnt, dass man das Ergebnis der Arbeit immer sofort auf dem Bildschirm hat. Die Entwicklung einer solchen Möglichkeit ist mit Sicherheit eine Herausforderung und letztlich eine gewaltige Erleichterung bei der Eingabe der Fragen.

8.2 Sicherheit

Das Thema Sicherheit wurde in dieser Arbeit eher stiefmütterlich behandelt. Die gesamte Auswertung wurde auf dem Server belassen, um das eTesting System entsprechend sicher zu gestalten. Sicherheitslücken könnten Studenten ungeahnte Vorteile verschaffen, deshalb ist unbedingt festzustellen, ob die getroffenen Maßnahmen ausreichend sind, um Manipulationen bei der Durchführung einer Prüfung zu unterbinden. Eventuell müssten entsprechende Maßnahmen wie zum Beispiel die Verschlüsselung der Übertragungswege getroffen werden.

9 Anhang A

9.1 Quellcode

9.1.1 Auswertung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Auswertung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------------
//
//    Funktion zur Auswertung
//
//--- History ------------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
  require("Globals.php");
  require("FormatInput.php");
  include("Loesung_Pruefung.php");
  include("Loesung_Uebung.php");
  @session_start ();
  $FID = 0;
  //FragenID aus den Uebergabeparametern
  $FormatedInputs = array ();
  //die restlichen, formatierten Uebergabeparameter
  $Ergebnis      = array ();
  //Speicher für die Ergebnisse
```

```
if (FormatInput())
//bringt die Uebergabeparameter in die richtige Form
{
  $mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
  //Verbindung zur Datenbank, die Verbindungsdaten stehen in "Globals"

  if (is_resource($mysqllink))
  //prueft die Verbindung
  {
    mysql_select_db ($SQLDATABASE, $mysqllink);
    //Auswahl der Tabelle, Tabellenname steht in "Globals"
    $sqlbefehl = sprintf ("SELECT * FROM ‘antworten‘
                           WHERE aFragenID = \"%s\"", $FID);
    $Antwort = mysql_query($sqlbefehl, $mysqllink);
    //SQL-Anfrage; holt alle Antworten zur entsprechenden FragenID
    while ($row = mysql_fetch_assoc($Antwort)):
    //nun wird jede Anwort, die in der Datenbank gespeichert ist, überprüft
      $temp = $row["aTyp"].$row["aNummer"];
      //aus Typ und Nummer die Kennung machen
      switch ($row["aTyp"])
      {
        case "SC":
        case "MC":
          if ($FormatedInputs[$temp] == $row["Antwort"])
          //Antwort überprüfen und
          {
            $Ergebnis[$temp] = "TRUE";
            //TRUE bzw.
          }
          else
          {
            $Ergebnis[$temp] = "FALSE";
            //FALSE in das Array schreiben
          }
          break;
        case "SW":
          $SWArray = explode (";", $row["Antwort"]);
          $Ergebnis[$temp] = "FALSE";
          foreach ($SWArray as $value)
          {
             if (utf8_encode(strtolower(utf8_decode($FormatedInputs[$temp])))

                  == utf8_encode(strtolower($value)))
              //Antwort überprüfen und
```

```
                {
                    $Ergebnis[$temp] = "TRUE";
                }
            }
            break;
          default:
            break;
        }
      endwhile;
      if ("Uebung" == $_SESSION['TESTTYPE'])           //war es eine Uebung?
      {
        zeigeLoesungUebung ($Ergebnis, $FID);          //zeige die Lösung
      }
      else if ("Pruefung" == $_SESSION['TESTTYPE'])
      //oder eine Pruefungsfrage?
      {
        Auswertung ($Ergebnis, $FID);
        //werte aus und stelle nächste Frage
      }
    }
    else
    //Datenbank nicht vorhanden!
    {?>
      <html>
        <head>
          <title>eTesting-Umgebung für NUMAS</title>
          <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
        </head>
        <body>
          <h1>Datenbank nicht vorhanden</h1>
        </body>
      </html>
  <?}
  }
?>
```

9.1.2 EMail.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  EMail.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Funktion zum Versand einer E-Mail
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------

function mailTo ($mail_empfaenger, $mail_absender, $text) {
  $betreff = "eTesting System";
  mail($mail_empfaenger, $betreff, $text, "from:$mail_absender");
}
?>
```

9.1.3 Erfolgsmeldung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Erfolgsmeldung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Bricht die Prfung ab (alle Fragen beantwortet oder Time up),
//    zeigt das Ergebnis an und versendet es per Email
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
      <link rel="stylesheet" type="text/css" href="format.css" />
      <script  type="text/javascript">
      <!--
      function stopTimer ()
      {
        parent.Timer.location.href = "leer.html";
      }
      //-->
      </script>
  </head>
  <body>
<?PHP
    require ("Globals.php");
    require ("EMail.php");
```

```
@session_start ();
$AbbruchGrund = $_GET['AbbruchGrund'];
switch ($_GET['AbbruchGrund'])
{
    case "TimeUp":
      echo "    <script  type=\"text/javascript\">\n";
      echo "      stopTimer ();\n";
      echo "    </script>\n";
      echo "<h2>Die Zeit ist leider abgelaufen</h2>\n";
      break;
    case "OutofQuests":
      echo "    <script  type=\"text/javascript\">\n";
      echo "      stopTimer ();\n";
      echo "    </script>\n";
      echo "<h2>Sie haben alle Fragen beantwortet.\n
            Die Prüfung ist zu Ende.</h2>\n";
      break;
    default:
      echo "<h1>Prüfung abgebrochen</h1>\n";
      break;
}
$FragenCounter = 0;
$GesamtProzent = 0;
$MAILTEXT = "Es wurde eine Prüfung von ".$_SESSION['TESTEENAME'].
            " durchgeführt.\n";
foreach ($_SESSION['ACHIEVEMENT'] as $key => $value)
{
  echo "Sie haben die Frage Nr. ".$key." zu ". $value.
       " Prozent richtig beantwortet.<br>\n";
  $Mail_TEMP = sprintf ("Die Frage Nr. %d wurde zu
                         %d Prozent richtig beantwortet.
                         \n", $key, $value);
  $MAILTEXT = $MAILTEXT.$Mail_TEMP;
  $GesamtProzent = $GesamtProzent + $value;
  $FragenCounter++;
}
if (0 != $FragenCounter)
{
  $PruefungsErfolg = $GesamtProzent / $FragenCounter;
}
else
{
  $PruefungsErfolg = 0;
}
```

```
    echo "Sie haben einen Prüfungs Ergebnis von ".$PruefungsErfolg.
         " Prozent erreicht.<br>\n";
    $Mail_TEMP = sprintf ("Es wurde ein Prüfungs Ergebnis von
                           %d Prozent erreicht.\n", $PruefungsErfolg);
    $MAILTEXT = $MAILTEXT.$Mail_TEMP;
    $sek = $_SESSION['RESTTIME'];
    $RestZeit = sprintf('%02d Stunde%s '.' %02d Minute%s und %02d Sekunde%s',
                        $sek / 3600 % 24,
                        floor($sek / 3600 % 24) != 1 ? 'n':'',
                        $sek / 60 % 60,
                        floor($sek / 60 % 60) != 1 ? 'n':'',
                        $sek % 60,
                        floor($sek % 60) != 1 ? 'n':''
                        );
    echo "Sie hätten noch ".$RestZeit." Zeit gehabt.<br>\n";
    $Mail_TEMP = sprintf ("Es wäre noch %s Zeit gewesen.\n", $RestZeit);
    $MAILTEXT = $MAILTEXT.$Mail_TEMP;
    mailTo ($_SESSION['TESTEEMAIL'], $_SESSION['AUDITORMAIL'], $MAILTEXT);
    mailTo ($_SESSION['AUDITORMAIL'], $SYSTEMEMAIL, $MAILTEXT);
    echo "Diese Daten wurden an die Email Adresse: ".
          $_SESSION['TESTEEMAIL']."<br> und an Ihrem Prüfer
          (E-Mail: ".$_SESSION['AUDITORMAIL'].") gesendet.<br>\n";
?>
  </body>
<html>
```

9.1.4 FormatInput.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  FormatInput.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Funktion, um die EingabeParameter zu formatieren
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
    function FormatInput ()
    //Funktion, um die EingabeParameter zu formatieren
    {
      global $FormatedInputs;
      global $FID;
      if ( isset ($_POST["FragenID"]))
      //gibt es die FragenID?
      //=> wenn nicht:Übertrageungsfehler!?!
      {

        $FID = $_POST["FragenID"];
        //FragenID auslesen
        foreach ($_POST as $key => $value):
        //alle anderen Paramter auswerten
          /*** SC - Möglichkeit ***/
          if (strstr ($key, "SC")):
          //SingleChoice-Antwort gefunden
            $FormatedInputs[$key] = $value;
          /*** MC - Möglichkeit ***/
          elseif (strstr ($key, "MC")):
```

```
      //MultipleChoice-Antwort gefunden
      //=> Array in Binärwert und dann in
      //Dezimalwert umwandeln
      rsort ($value);
      //Sortierung des Arrays
      //in umgekehrter Reihenfolge
      $MCBin = sprintf(" ");
      //anlegen eines Strings
      do
      //solange Werte im Array stehen, werden
        $MCBin{(current($value)- 1)}="1";
        //an die entsprechende Position im String
      while (next($value)!==false);
      //"1" geschrieben
      $MCBin = str_replace (" ", "0", $MCBin);
      //alle übrigen Positionen mit 0 auffüllen
      $MCBin = rtrim ($MCBin);
      $MCBin = ltrim ($MCBin);
      $MCBin = strrev ($MCBin);
      $FormatedInputs[$key] = $MCBin;
    /*** SW - Möglichkeit ***/
    elseif (strstr ($key, "SW")):
    //Schlüsselwort-Antwort gefunden
      $SWArray = array ();
      do
        array_push ($SWArray, current($value));
      while (next($value)!==false);
      sort ($SWArray);
      $SWString = implode (";", $SWArray);
      $FormatedInputs[$key] = $SWString;
    endif;
  endforeach;
}
else
{?>
    <html>
    <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type"
       content="text/html;charset=iso-8859-1">
    </head>
    <body>
      <h4>Auswertung.php: Übergabeparameter falsch</h4>
    </body>
```

```
        </html>
       <?
        return false;
      }
      return true;
    }
?>
```

9.1.5 Fragen.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Fragen.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Anzeige der Frage
//    erzeugt aus dem XML und XSL eine HTML - Seite
//    und zeigt diese an
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
    require ("Globals.php");
    $mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
    //Verbindung zur Datenbank, die Verbindungsdaten stehen in "Globals"
    if (is_resource($mysqllink))
    //Verbindung vorhanden?
    {
        mysql_select_db ($SQLDATABASE, $mysqllink);
        //Datenbank auswählen
        if (($FID = $_GET["FragenID"]))
        //FragenID aus den Übergabeparametern holen
        {
            $sqlbefehl = sprintf ("SELECT FrageXML FROM ‘fragen‘
                                   WHERE FragenID = \"%s\"", $FID);
            //XML aus der Datenbank holen
            $XML_Frage = mysql_query($sqlbefehl, $mysqllink);
            $rowXML = mysql_fetch_assoc($XML_Frage);
            $xml = $rowXML["FrageXML"];
            $xsl = $_SERVER["DOCUMENT_ROOT"]."/eTesting/aufgabenstellung.xsl";
```

```
            //XSL aus dem File holen

            $attribute = array ("/_xml" => $xml);

            $xh=xslt_create ();
            $result = xslt_process ( $xh, 'arg:/_xml', 'file://'.$xsl, NULL,
                                     $attribute, array ("FID" => $FID));
            //HTML erzeugen
            xslt_free ($xh);

            echo $result;
            //HTML anzeigen
        }
        else
        //Keine ID übermittelt
        {?>
            <html>
                <head>
                <title>Fragen</title>
                <meta http-equiv="Content-Type"
                 content="text/html;charset=iso-8859-1">
                <link rel="stylesheet" type="text/css" href="format.css" />
                </head>
                <body>
                <h1>Keine Frage ausgewählt</h1>
                </body>
            </html>
      <?}
        mysql_close($mysqllink);
    }
    else
    //Datenbank nicht vorhanden!
    {?>
      <html>
        <head>
          <title>eTesting-Umgebung für NUMAS</title>
          <meta http-equiv="Content-Type"
           content="text/html;charset=iso-8859-1">
        </head>
        <body>
          <h1>Datenbank nicht vorhanden</h1>";
        </body>
      </html>
  <?}?>
```

9.1.6 Globals.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Globals.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------------
//
//    Properties
//
//--- History ------------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
  //Globale Variablen
  $SQLHOST      = "localhost";              //Host der Datenbak
  $SQLDATABASE  = "etesting";               //Datenbankname
  $SQLUSR       = "UserName";               //User Name
  $SQLPWD       = "PASSWD";                 //Passwort
  $SYSTEMEMAIL  = "eTesting@localhost"      //Email-Adresse unter der
                                            //das Ergebnis an den Prüfer
                                            //gesendet wird
?>
```

9.1.7 Loesung_Pruefung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Loesung_Pruefung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Funktion zur Auswertung
//    Anzeige der nächsten Frage
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
    <script  type="text/javascript">
    <!--
    function loadNextQuestion (ID)
    {
      var QuestionLink = "Fragen.php?FragenID=" + ID;
      parent.MainFrame.location.href = QuestionLink;
    }
    function loadconfirmationmessage ()
    {
      var ConfirmLink = "Erfolgsmeldung.php?AbbruchGrund=OutofQuests";
      parent.MainFrame.location.href = ConfirmLink;
    }
    //-->
    </script>
<?PHP
  function Auswertung ($Ergebnis, $FID)        //Prüfungsauswertung
  {
    $GesamtAnzahlFragen = count ($Ergebnis);
```

```
    $AnzahlRichtigeAntworten = 0;
    foreach ($Ergebnis as $value)
    {
        if ("TRUE" == $value)
          $AnzahlRichtigeAntworten++;
    }
    $ProzentRichtig = ($AnzahlRichtigeAntworten / $GesamtAnzahlFragen)*100;

    $_SESSION['ACHIEVEMENT'][$FID]=$ProzentRichtig;

    //FID in die ANSWEREDQUESTIONS schreiben
    $_SESSION['ANSWEREDQUESTIONS'][] = $FID;

    //FID aus der UNANSWEREDQUESTIONS löschen
    $key = array_search($FID, $_SESSION['UNANSWEREDQUESTIONS']);
    unset ($_SESSION['UNANSWEREDQUESTIONS'][$key]);
    $_SESSION['UNANSWEREDQUESTIONS'] =
    array_values ($_SESSION['UNANSWEREDQUESTIONS']);

    if (0 ==  count($_SESSION['UNANSWEREDQUESTIONS']))
    //Gibt es keine UNANSWEREDQUESTIONS mehr?
    {
    //Nein? dann Erfolgsmeldung
      echo "    <script  type=\"text/javascript\">\n";
      echo "      loadconfirmationmessage ();\n";
      echo "    </script>\n";
    }
    else
    //doch?
    {
    //dann aufrufen
      echo "    <script  type=\"text/javascript\">\n";
      echo "      loadNextQuestion ("
                  .$_SESSION['UNANSWEREDQUESTIONS'][0].");\n";
      echo "    </script>\n";
    }
  }
?>
```

9.1.8 Loesung_Uebung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Loesung_Uebung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Funktion zur Anzeige der Auswertung & Lösung
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
function zeigeLoesungUebung ($Ergebnis, $FID)          //Lösung
anzeigen {
    require("Globals.php");
    $GesamtAnzahlFragen = count ($Ergebnis);
    $AnzahlRichtigeAntworten = 0;
    $mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
    //Verbindung zur Datenbank, die Verbindungsdaten stehen in "Globals"
    if (is_resource($mysqllink))
    //Verbindung vorhanden?
    {
      mysql_select_db ($SQLDATABASE, $mysqllink);
      //Datenbank auswählen
      $sqlbefehl = sprintf ("SELECT FrageXML FROM ‘fragen‘
                             WHERE FragenID = \"%s\"", $FID);
      //XML aus der Datenbank holen
      $XML_Frage = mysql_query($sqlbefehl, $mysqllink);
      $rowXML = mysql_fetch_assoc($XML_Frage);
      $xml = $rowXML["FrageXML"];
      $xsl = $_SERVER["DOCUMENT_ROOT"]."/eTesting/loesung.xsl";
      //XSL aus dem File holen
```

```
//Loesung XSL hat KEINEN Senden Button
$attribute = array ("/_xml" => $xml);
$xh=xslt_create ();
$result = xslt_process ( $xh, 'arg:/_xml', 'file://'.$xsl, NULL,
                         $attribute, array ("FID" => $FID));
//HTML erzeugen
xslt_free ($xh);
echo $result;                              //HTML anzeigen
foreach ($Ergebnis as $key => $value)
{
  if ("TRUE" == $value)
  {
    $COLOR = "'green'";
    $AnzahlRichtigeAntworten++;
  }
  else
  {
    $COLOR = "'red'";
  }
  echo "  <script type='text/javascript'>\n";
  echo "    for(i = 0; i < document.getElementsByName('".$key."')
              .length; i++){\n";
  echo "        document.getElementsByName('".$key."')[i]
              .disabled=true;\n";
  echo "        document.getElementsByName('".$key."')[i].
              style.backgroundColor=".$COLOR." };\n";

  $FragenTYP = $key{0}.$key{1};
  $FragenNO  = "";
  for ($z=2; $z < strlen($key); $z++)
  {
    $FragenNO .= $key{$z};
  }
  $sqlbefehl = sprintf ("SELECT Antwort FROM `antworten`
                         WHERE aFragenID = \"".$FID."\"
                         AND aTyp=\"".$FragenTYP."\" and
                         aNummer=\"".$FragenNO ."\"");
  $Antwort = mysql_query($sqlbefehl, $mysqllink);
  //SQL-Anfrage; holt die Antwort zur entsprechenden Frage
  $row = mysql_fetch_assoc($Antwort);
  switch ($FragenTYP)
  {
    case "SC":
      $RightSelection = $row["Antwort"]-1;
```

```
          echo "    document.getElementsByName('".$key."')
                    [".$RightSelection."].checked=true;\n";
          break;
        case "SW":
          $RightSelection = $row["Antwort"];
          $RightSelection = str_replace (";", " oder ",$RightSelection);
          echo "    for(i = 0; i < document.getElementsByName('".$key."')
                        .length; i++){\n";
          echo "    document.getElementsByName('".$key."')[i].size=".
                    strlen ($RightSelection.length).";\n";
          echo "    document.getElementsByName('".$key."')[i]
                    .style.color='white';\n";
          echo "    document.getElementsByName('".$key."')[i].value=\"".
                    utf8_encode($RightSelection)."\";}\n";
          break;
        case "MC":
          echo "var i = (document.getElementsByName('".$key."').length)-1;\n";
          $RightSelection = $row["Antwort"];
          for ($zaehler = 0; $zaehler < strlen ($RightSelection); $zaehler++)
          {
            if ("1" == $RightSelection{$zaehler})
              echo "document.getElementsByName('".$key."')[i-".$zaehler."]
                    .checked=true;\n";
          }
          break;
        default:
          break;
      }
      echo "  </script>\n";
    }
    echo "<html>\n";
    echo "  <body>\n";
    $ProzentRichtig = ($AnzahlRichtigeAntworten / $GesamtAnzahlFragen)*100;
    //noch die Angabe, wieviel Fragen richtig beantwortet wurden
    echo "    Sie haben ".$ProzentRichtig." % der Fragen richtig beantwortet.
              <br>\n";
    echo "  </body>\n";
    //HTML Seite abschließen!
    echo "</html>\n";
    mysql_close($mysqllink);
  }
  else
  //Datenbank nicht vorhanden!
  {
```

```
?>
      <html>
        <head>
          <title>eTesting-Umgebung für NUMAS</title>
          <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
        </head>
        <body>
          <h1>Datenbank nicht vorhanden</h1>";
        </body>
      </html>
<?  } } ?>
```

9.1.9 Navigation.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Navigation.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    zeigt alle Lernfelder und Lernobjekte an und
//    gibt die Möglichkeit zur Auswahl der Fragen zu Übungszwecken
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
      <link rel="stylesheet" type="text/css" href="format.css">
  </head>
  <body>

  <?PHP
    require("Globals.php");                          //Properties holen
    @session_start();
    //da die "Uebungs-Haupt-Seite" geladen wurde
    unset ($_SESSION);                               //die alte SESSION zerstören
    session_destroy ();
    @session_start ();                               //und eine neue erstellen
    $_SESSION['TESTTYPE'] = "Uebung";
    //es wird eine Übung abgehalten (Unterschiede in der Auswertung)
```

```
$mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
//Datenbank Verbindung aufbauen

if (is_resource($mysqllink))
{
  mysql_select_db ($SQLDATABASE, $mysqllink);
  $sqlbefehl = "SELECT LernfeldName, LernFeldID FROM lernfelder";
  //Alle Lernfelder holen
  $alleLernfelder = mysql_query($sqlbefehl, $mysqllink);
  if (is_resource($alleLernfelder))
  {
    while ($rowALF = mysql_fetch_assoc($alleLernfelder))
    {
      echo "<h4>".$rowALF["LernfeldName"]."</h4>";
      //jedes Lernfeld auflisten
      $sqlbefehl = "SELECT bll_LernObjektID FROM beziehung_lf_lo
                    WHERE bll_LernfeldID =".$rowALF["LernFeldID"];
      //und alle Lernobjekte des Lernfeldes
      $alleLernobjekte = mysql_query($sqlbefehl, $mysqllink);
      //listen
      while ($rowALO = mysql_fetch_assoc($alleLernobjekte))
      {
        $sqlbefehl = "SELECT LernObjekt FROM lernobjekte
                      WHERE LernObjektID =".$rowALO
                      ["bll_LernObjektID"];
        $Lernobjekt = mysql_query($sqlbefehl, $mysqllink);
        $rowLON = mysql_fetch_assoc($Lernobjekt);
        echo "<h5>".$rowLON["LernObjekt"]."</h5>";
        $sqlbefehl = "SELECT FragenID FROM ‘fragen‘
                      WHERE fLernObjektID = \"".
                      $rowALO["bll_LernObjektID"]."\"";
        $FragenIDs = mysql_query($sqlbefehl, $mysqllink);
        $FragenCount = 1;
        while ($rowFID = mysql_fetch_assoc($FragenIDs))
        //alle Fragen des Lernobjekts listen
        {
          $ausgabe = sprintf ("<a href=\"Fragen.php?FragenID=%s\"
                               target=Fragen>Frage %s</a><br>",
                               $rowFID["FragenID"], $FragenCount);
          echo $ausgabe;
          $FragenCount++;
        }
      }
    }
```

```
      }
      mysql_close($mysqllink);
    }
    else
    {
      echo "Datenbank nicht vorhanden";
    }
  ?>
  </body>
</html>
```

9.1.10 Pruefung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Pruefung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    User Interface zur Prüfungsauswahl
//
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
      <link rel="stylesheet" type="text/css" href="format.css" />
  </head>
  <body>

<?PHP
    require("Globals.php");       //Properties holen
    @session_start();             //da die "Haupt-Prüfungs-Seite" geladen wurde
    unset ($_SESSION);           //erstmal alte SESSION zerstören
    session_destroy ();
    @session_start ();           //und neue erstellen
    $_SESSION['TESTTYPE'] = "Pruefung";
    //es wird eine Prüfung abgehalten (Unterschiede in der Auswertung)
    $mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
    //Datenbank Verbindung Verbindungsdaten in Globals.php
```

```
    if (is_resource($mysqllink))
    {
      mysql_select_db ($SQLDATABASE, $mysqllink);
      $sqlbefehl = "SELECT PruefungsID, PruefungsName FROM pruefungen WHERE 1";
      //Alle Prüfungen holen (Namen und ID´s)
      $allePruefungen = mysql_query($sqlbefehl, $mysqllink);
      $PruefungsNamen = array ();
      if (is_resource($allePruefungen))
      {
        while ($row = mysql_fetch_array($allePruefungen))
        {
         $PruefungsNamen[$row["PruefungsID"]] = $row["PruefungsName"];
        }
      }
      mysql_close($mysqllink);
    }
    else
    {
      echo "Datenbank nicht vorhanden";
    }
?>
     <form action="StartePruefung.php" method="POST">
       <h3>Bitte Prüfungs auswählen:</h3>
<?PHP
       $AnzahlPruefungen = count ($PruefungsNamen);
       //eine DropDown Box mit den Prüfungen erstellen
       echo "<select name=\"PruefungsID\">";
       if (0 == $AnzahlPruefungen)
       {
          echo "<option> Keine Prüfung vorhanden</option>\n";
       }
       else
       {
         foreach ($PruefungsNamen as $key => $value)
         {
           echo "<option value=\"".$key."\">".$value."   ".$key."</option>\n";
         }
         echo "</select>";
         echo "<br><br>NAME:";
         echo "<input id=\"name\" name=\"Name\" size =\"30\"/><br><br>";
         //nach Name und
         echo "EMail Adresse:<br>";
         echo "<input id=\"mail\" name=\"EMail\" size =\"30\"><br><br>";
         //Email Adresse des Prüflings fragen
```

```
          echo "<input type=\"submit\" value=\"Starte Prüfung\"/>";
        }
?>
      </form>
  </body>
</html>
```

9.1.11 Random.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Random.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    Funktion um Zufallszahlen zu berechnen
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
function random ($min, $max) {
    static $done;
    if (!$done)
    {
      mt_srand ((float) microtime () * 1000000);
      $done = true;
    }
    return mt_rand ($min, $max);
} ?>
```

9.1.12 StartePruefung.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  StartePruefung.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    füllt alle notwendigen SESSION Variablen und
//    startet die Prüfung (Timer)
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
      <link rel="stylesheet" type="text/css" href="format.css" />
      <script  type="text/javascript">
      <!--
      function startTimer (time)
      {
        var timerLink = "timer.php?TIME=" + time;
        parent.Timer.location.href = timerLink;
      }
      function openQuestion (ID)
      {
        var QuestionLink = "Fragen.php?FragenID=" + ID;
        parent.MainFrame.location.href = QuestionLink;
      }
      //-->
```

```
      </script>
  </head>
  <body>
<?PHP
    require("Globals.php");
    require("Random.php");
    @session_start();                              //SESSION holen
    $PruefungsID    = $_POST["PruefungsID"];       //Übergabe Parameter auslesen
    $PrueflingsName = $_POST["Name"];
    $PrueflingsMail = $_POST["EMail"];
    $ausgewaehlteFragen = array ();
    $PrueferEmailAdresse = "";
    $Error          = 0;
    if ($PrueflingsName == "" ||
    //ist Name und Email-Adresse eingetragen?
        $PrueflingsMail  == ""  )
    {
      echo "Bitte Name und Email-Adresse eingeben!<br>";
    }
    else
    {
      $mysqllink = mysql_connect ($SQLHOST, $SQLUSR, $SQLPWD);
      //Verbindung zur Datenbank, die Verbindungsdaten stehen in "Globals"
      if (is_resource($mysqllink))
      //prueft die Verbindung
      {
        mysql_select_db ($SQLDATABASE, $mysqllink);
        //Auswahl der Tabelle, Tabellenname steht in "Globals"
        $sqlbefehl = sprintf ("SELECT beLernObjektID, AnzahlderFragen FROM
                              beziehung_lo_pruefung WHERE bePruefungsID = %s",
                              $PruefungsID);
        $LernObjekte = mysql_query($sqlbefehl, $mysqllink);
        while ($rowLO = mysql_fetch_assoc($LernObjekte))
        {
          $sqlbefehl = sprintf ("SELECT FragenID FROM fragen
                                 WHERE fLernObjektID = %s",
                                $rowLO[beLernObjektID]);
          $alleFragenausLernobjekt = mysql_query($sqlbefehl, $mysqllink);
          $Array_AlleFragenIDLO = array ();
          while ($rowAFLO = mysql_fetch_assoc($alleFragenausLernobjekt))
          {
            $Array_AlleFragenIDLO[] = $rowAFLO[FragenID];
          }
          $AnzahlderFragenLO = count ($Array_AlleFragenIDLO);
```

```
    if ($AnzahlderFragenLO >= $rowLO[AnzahlderFragen])
    {
      if ($AnzahlderFragenLO == $rowLO[AnzahlderFragen])
      {
        foreach ($Array_AlleFragenIDLO as $value)
        {
          $ausgewaehlteFragen[] = $value;
        }
      }
      else
      {
        for ($zaehler = 0; $zaehler < $rowLO[AnzahlderFragen]; $zaehler++)
        {
          $aktuellerStand = count ($Array_AlleFragenIDLO);
          $Zufallszahl = random (0, $aktuellerStand-1);
          $ausgewaehlteFragen[] = $Array_AlleFragenIDLO[$Zufallszahl];
          unset ($Array_AlleFragenIDLO[$Zufallszahl]);
          $Array_AlleFragenIDLO = array_values($Array_AlleFragenIDLO);
        }
      }
    }
    else
    {
      $Error = -1;
    }
  }
  $sqlbefehl = sprintf ("SELECT PruefungsZeit FROM pruefungen
                         WHERE PruefungsID = %s", $PruefungsID);
  $ATestTime = mysql_query($sqlbefehl, $mysqllink);
  $rowTT = mysql_fetch_assoc($ATestTime);
  $TestTime = $rowTT[PruefungsZeit];
  $sqlbefehl = sprintf ("SELECT Prueferemail FROM pruefungen
                          WHERE PruefungsID = %s", $PruefungsID);
  $APruerferMail = mysql_query($sqlbefehl, $mysqllink);
  $rowPEM = mysql_fetch_assoc($APruerferMail);
  $PrueferEmailAdresse = $rowPEM[Prueferemail];
  mysql_close($mysqllink);
}

if (0 == $Error)
{
  echo "    <h1>Prüfung Nr.".$PruefungsID."</h1>\n";
  //Prüfungsdaten listen
  echo "    Name: ".$PrueflingsName."<br>\n";
```

```
        echo "    EMail: ".$PrueflingsMail."<br>\n";
        echo "    <script  type=\"text/javascript\">\n";
        $LesbareZeit = sprintf('%02d Stunde%s '.'
                               %02d Minute%s und
                               %02d Sekunde%s',
                               $TestTime / 3600 % 24,
                               floor($TestTime / 3600 % 24)!=1?'n':'',
                               $TestTime / 60 % 60,
                               floor($TestTime / 60 % 60)!=1?'n':'',
                               $TestTime % 60,
                               floor($TestTime % 60)!=1?'n':''
                             );
        if (count ($ausgewaehlteFragen) == 1)
        {
          $MSGFragen = "wird ".count ($ausgewaehlteFragen)." Frage";
        }
        else
        {
          $MSGFragen = "werden ".count ($ausgewaehlteFragen)." Fragen";
        }
        $MSG = "      alert (\"Die Prüfung startet jetzt.\\nIhnen "
                              .$MSGFragen." gestellt.\\n
                              Zur Bearbeitung haben sie "
                              .$LesbareZeit." Zeit.\\nViel Erfolg.\");
                              \n";
        echo $MSG;
        echo "      startTimer (".$TestTime.");\n";     //Timer starten
        echo "    </script>\n";
        //SESSION Variablen erstellen und füllen
        $_SESSION['TESTEENAME']             = $PrueflingsName;
        $_SESSION['TESTEEMAIL']             = $PrueflingsMail;
        $_SESSION['TESTID']                 = $PruefungsID;
        $_SESSION['ACHIEVEMENT']            = array ();
        $_SESSION['ANSWEREDQUESTIONS']      = array ();
        $_SESSION['UNANSWEREDQUESTIONS']    = $ausgewaehlteFragen;
        $_SESSION['AUDITORMAIL'] = $PrueferEmailAdresse;
        echo "    <script  type=\"text/javascript\">\n";
        echo "      openQuestion (".$_SESSION['UNANSWEREDQUESTIONS'][0].");\n";
        echo "    </script>\n";
      }
    }
?>
  </body>
</html>
```

9.1.13 timer.php

```
<?PHP
//
//    Projekt  :  NUMAS eTesting System
//    File     :  Timer.php
//    Autor    :  Stefan H. Kleemann
//
//--- Copyright (c) ------------------------------------------------------
//
//    2005 FH Aachen
//         Prof. Dr.rer.nat. Wilhelm Hanrath
//         Prof. Dr.rer.nat. Gisela Engeln-Müllges
//
//--- Inhalt -------------------------------------------------------------
//
//    setzt einen Timer und bricht die Prüfung ab, wenn der
//    Timer zu Ende ist
//
//--- History ------------------------------------------------------------
//
//    Datum      Autor          Bemerkung
//    ---------------------------------------------------------------------
//    13.10.2005 SHKleemann     erstellt
//---------------------------------------------------------------------------
?>
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
      <link rel="stylesheet" type="text/css" href="format.css" />
      <script  type="text/javascript">
      <!--
        var refreshtime="10"
        function site_refresh()
        {
            if (refreshtime==0)
                window.location.reload()
            else
            {
                refreshtime-=1
                setTimeout("site_refresh()",1000)
            }
        }
```

```
        window.onload=site_refresh
        function callAchievementSite ()
        {
          alert ("Die Zeit ist leider abgelaufen")
          parent.MainFrame.location.href = "Erfolgsmeldung.php?
                                            AbbruchGrund=TimeUp";
        }
      //-->
      </script>
  </head>
<?PHP
    @session_start ();
    if(!isset($_SESSION['STARTTIME']))
    {
      $_SESSION['STARTTIME'] = time();
    }
    if(!isset($_SESSION['ENDTIME']))          //Timer noch nicht gestartet?
    {
      $_SESSION['ENDTIME'] = $_SESSION['STARTTIME']+$_GET["TIME"];
      //setzte den Timer
    }
    $_SESSION['RESTTIME'] = ($_SESSION['ENDTIME']-time());
    if ($_SESSION['RESTTIME'] <= 0)
    {
      echo "<body onload=\"callAchievementSite()\">";
    }
    else
    {
      echo "<body>";
    }
    echo "Verbleibende Zeit:<br>";             //Ausgabe der Restzeit
    $restTime = sprintf('%02d Stunde%s<br>'.' %02d Minute%s<br>%02d Sekunde%s',
                        $_SESSION['RESTTIME'] / 3600 % 24,
                        floor($_SESSION['RESTTIME'] / 3600 % 24) != 1 ? 'n':'',
                        $_SESSION['RESTTIME'] / 60 % 60,
                        floor($_SESSION['RESTTIME'] / 60 % 60) != 1 ? 'n':'',
                        $_SESSION['RESTTIME'] % 60,
                        floor($_SESSION['RESTTIME'] % 60) != 1 ? 'n':''
                       );
    echo $restTime;
?>
  </body>
</html>
```

9.1.14 index_Pruefung.html

```
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
  </head>
  <frameset cols="250,*">
    <frameset rows="150, *">
    <frame src="leer.html" name="Timer">
    <frame src="Pruefung.php" name="Pruefung">
    </frameset>
    <frame src="leer.html" name="MainFrame">
    <noframes>
      Ihr Browser kann diese Seite leider nicht anzeigen!
    </noframes>
  </frameset>
</html>
```

9.1.15 index_Uebungen.html

```
<html>
  <head>
      <title>eTesting-Umgebung für NUMAS</title>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1">
  </head>
  <frameset cols="250,*">
    <frame src="Navigation.php" name="Navigation">
    <frame src="Fragen.php" name="Fragen">
    <noframes>
      Ihr Browser kann diese Seite leider nicht anzeigen!
    </noframes>
  </frameset>
</html>
```

Abbildungsverzeichnis

Literaturverzeichnis

[1] Software für Prüfungen am PC
OTeS / Produktvarianten / Spezifikation
OTeS Test
Homepage der Firma Berg Communications
www.berg.de

[2] Software für Prüfungen am PC
OTeS / Produktvarianten / Spezifikation
OTeS Editor
Homepage der Firma Berg Communications
www.berg.de

[3] Homepage der Firma Berg Communications
www.berg.de

[4] Einstufungstest zum MSCA - Kurs der Firma iTrain GmbH.
Der Test wurde auf der HomePage
der Firma iTrain GmbH „www.braintors.com“ durchgeführt.

[5] NUMAS
DV-Labor der FH-Aachen
www.numas.de
Stand: 10. Dezember 2005

[6] Homepage www.wikepedia.org
Stand 10. Dezember 2005