Aus dem Universitätsklinikum Münster

Institut für Medizinische Informatik und Biomathematik

– Direktor: Univ.-Prof. Dr. W. Köpcke –

Webapplikation zur universitätsübergreifenden Administration von Prüfungsfragen für modulare Studiengänge der Medizin

Inaugural-Dissertation

zur Erlangung des

doctor rerum medicinalium

der Medizinischen Fakultät

der Westfälischen Wilhelms-Universität Münster

vorgelegt von Urich, Wolfram

aus Rheda-Wiedenbrück

2007

Gedruckt mit Genehmigung der Medizinischen Fakultät der Westfälischen Wilhelms-Universität Münster

Dekan: Univ.-Prof. Dr. V. Arolt

1. Berichterstatter: Prof. Dr. F. Ückert

2. Berichterstatter: Univ.-Prof. Dr. R. Lellé

Tag der mündlichen Prüfung: 24.05.07

Aus dem Universitätsklinikum Münster
Institut für Medizinische Informatik und Biomathematik
– Direktor: Univ.-Prof. Dr. W. Köpcke –
Referent: Prof. Dr. F. Ückert
Koreferent: Univ.-Prof. Dr. R. Lellé

Zusammenfassung

Webapplikation zur universitätsübergreifenden Administration von Prüfungsfragen für modulare Studiengänge der Medizin

Urich, Wolfram

Die neue Approbationsordnung für Ärzte vom 27. Juni 2002 führte an der Medizinischen Fakultät der WWU Münster im Zuge einer neuen Studienordnung im klinischen Studienabschnitt zu semesterweiten Klausuren, die aus Multiple-Choice-Fragen bestehen. Der Vorgang des Einforderns der Prüfungsfragen von den beteiligten Dozenten erwies sich als aufwendig. Die Qualität mancher Fragen war gering.

Ein Ziel dieser Arbeit war die Ausarbeitung der genauen Workflows zum Einsammeln der Fragen und die Implementierung des Prüfungsmanagement-Systems. Zur Qualitätssteigerung der Prüfungsfragen sollten die Arbeitsabläufe für ein Reviewverfahren entwickelt werden. Die Begutachtung sollte fakultätsintern oder in Kooperation mit anderen Medizinischen Fakultäten Nordrhein-Westfalens erfolgen. Das System war anschließend zu realisieren.

Bis auf die Implementierung des Reviewsystems wurden die Ziele erfüllt. Entgegen der ursprünglichen Planung musste zunächst ein Prüfungssystem implementiert werden. Das Gesamtsystem trägt den Namen **E**lektronisches **P**rüfungsmanagement in der **M**edizin, kurz EPM.

Das Prüfungssystem ist seit Anfang 2005 in Betrieb, Teile des Prüfungsmanagements seit dem Sommersemester 2006. Der Einsatz des EPMs ermöglichte die Handhabung der Studienordnung und führte zu einer Minimierung des Arbeitsvolumens. Ein Vergleich mit ähnlichen Systemen, der kombinierte Einsatz und Perspektiven hinsichtlich eines weiteren Ausbaus des Systems werden ausgeführt.

Tag der mündlichen Prüfung: 24.05.07

Für meinen Vater

Danksagung

In den vergangenen zwei Jahren haben mich zahlreiche Menschen im beruflichen und privaten Umfeld unterstützt, ohne deren Anteilnahme und Hilfe diese Arbeit nicht in dieser Form hätte entstehen können.

Ich danke Prof. Dr. Frank Ückert für seine Unterstützung und Hilfestellung bei meinem Dissertationsvorhaben und das offene und kollegiale Miteinander.

Univ.-Prof. Dr. Ralph Lellé gilt mein Dank, da er ohne Zögern das Koreferat für diese Arbeit übernommen hat.

Den Mitarbeitern der Abteilung ITZ-FL gebührt mein Dank für die freundschaftliche und konstruktive Arbeitsatmosphäre und dafür, dass ich mich dort einfach zu Hause gefühlt habe. Insbesondere gilt mein Dank Ulrich Janßen und Dennis Toddenroth für die fruchtbare Zusammenarbeit bei der Entwicklung des Prüfungssystems. Ulrich Janßen möchte ich auch für seine wertvolle Mitarbeit beim CADS-Projekt und seine fachlichen Ratschläge danken. Bei Felix Gieseke bedanke ich mich für die sehr gute Zusammenarbeit beim CADS-Projekt und die ergiebigen Debatten bezüglich organisatorischer und fachlicher Herangehensweisen. Weiterhin gilt mein Dank Cliff Pereira, dessen Hilfe beim Testen und Programmieren einiger Teile des EPMs unverzichtbar war.

Meine Bürokollegen und späteren Freunde Gunnar Fischer und Max Ataian haben durch ihre humorvolle und verständnisvolle Art einen nicht unerheblichen Anteil daran, dass ich mich (fast) jeden Morgen auf die Arbeit gefreut habe. Sie haben mein Leben in den vergangenen zwei Jahren sehr bereichert.

Innerhalb der Abteilung IMFL habe ich mich sehr wohl gefühlt. Hier danke ich insbesondere dem Support-Team bestehend aus Philipp Neuhaus, Tobias Ullrich

und Lars Pfannenschmidt für ihre Arbeit und Einsatzbereitschaft, die oft über die übliche Pflichterfüllung hinaus ging. Die fachliche Begeisterung, die Hiep Doan mit mir bezüglich der Java EE-Technologie teilt, führte zu anregenden Diskussionen und einem vertieften Verständnis der Java-Softwareentwicklung.

Auch die übrigen Mitarbeiter des Instituts für Medizinische Informatik und Biomathematik wurden durch ihre freundliche Art und ihre Hilfsbereitschaft zu einem sehr angenehmen Teil meines täglichen Lebens.

Zineb Nouns aus der Charité Berlin danke ich für ihre Anregungen beim Vergleich der verschiedenen Systeme mit dem EPM.

Martin Schmidt, Verena Börder, Alexander und Hermann-Josef Booms und Johanna Kling gilt mein besonderer Dank für das kritische Gegenlesen dieser Arbeit. Sie haben sehr dazu beigetragen, die Qualität dieser Dissertation zu steigern.

An dieser Stelle möchte ich mich auch bei meinen Freunden Henrik Blunck, Philipp Kornmann, Oliver Heising und Christoph und Nina Le Viseur für ihren Beistand und ihre Unterstützung während des Dissertationsvorhabens bedanken.

Nicht zuletzt gilt mein Dank meinen Komilitonen Miriam Kolar, Philipp Freitag und Florian Büther, die mir durch ihre Anwesenheit und ihr sympathisches Wesen das promotionsbegleitende Studium sehr erleichtert haben.

Vor allem gilt mein persönlicher Dank meinen Eltern Annette und Franz, deren Liebe mich auf meinem gesamten bisherigen Lebensweg begleitet hat. Ihre Unterstützung hat es mir ermöglicht, meinen Weg zu gehen und meine Ziele zu erreichen.

Zu guter Letzt wendet sich mein persönlicher Dank jedoch an meine Freundin Andrea Welslau, ohne deren Vertrauen, Liebe und Geduld ich diese Arbeit nicht in dieser Form hätte erstellen können.

Inhaltsverzeichnis

1 Einleitung

1.1 Situation

Im Zuge der Umsetzung der neuen Approbationsordnung für Ärzte vom 27. Juni 2002 wurde an der Westfälischen Wilhelms-Universität Münster eine neue Studienordnung entwickelt. Der klinische Abschnitt des Medizinstudiums ist nicht mehr ausschließlich nach Fächern gegliedert. Ein großer Teil des Unterrichtsstoffes wird themenzentriert in sogenannten Modulen unterrichtet. Ein Themengebiet wie die Medizin des Herz-Kreislaufsystems wird von verschiedenen Fachdisziplinen aus ihrer jeweiligen Perspektive beleuchtet. In einem Modul sind mehrere Fächer, im Sinne von Fachdisziplinen, vertreten. Die Verteilung der Fächer auf mehrere Module brachte die Notwendigkeit einer kompletten Überarbeitung des bisherigen Prüfungsverfahrens mit sich. Die Fachvertreter der verschiedenen Fachdisziplinen können aufgrund der Zersplitterung ihrer Fächer nur noch im begrenzten Umfang eigene Prüfungen abhalten. Stattdessen ergab sich die Notwendigkeit einer semesterweiten Abschlussklausur, in der Fragen zu sämtlichen Modulen aus jeweils einem klinischen Semester gestellt werden. Diese Klausuren bestehen aus Multiple-Choice-Prüfungsfragen mit einer richtigen bzw. falschen Antwort bei fünf Antwortmöglichkeiten. Das Studiendekanat der Medizinischen Fakultät muss seitdem in einer aufwendigen Prozedur die Prüfungsfragen von den Dozenten der beteiligten Institute einfordern. Aufgrund der Vereinheitlichung der Prüfungsleistung ergab sich die Möglichkeit, die Qualität der Prüfungsfragen statistisch auszuwerten. Dabei trat ein zum Teil eklatanter Qualitätsmangel einiger Fragen zu Tage.

1.2 Problemstellung

Um dem hohen Arbeitsaufwand beim Einsammeln der Prüfungsfragen und dem Qualitätsdefizit der Fragen entgegen zu treten, bestand der Wunsch nach einem System, das sich dieser Probleme annimmt. Es existierten erste Vorstellungen bezüglich des Systems und der damit korrespondierenden Workflows. Es sollte ein Prüfungsmanagement entwickelt werden, mit dessen Hilfe, über verschiedene Instanzen, Prüfungsfragen in das System eingelesen werden können. Hier sollte es auch eine automatisierte Mahnfunktion geben, die beispielsweise die Autoren daran erinnert, dass sie noch Fragen erstellen müssen. Zusätzlich sollte ein Reviewsystem realisiert werden, durch das die Fragen nach verschiedenen Qualitätskriterien geprüft und bewertet werden können. Die Bewertung der Prüfungsfragen sollte zum einen intern durch die Fachvertreter der Medizinischen Fakultät und zum anderen extern durch Kooperation mit anderen Medizinischen Fakultäten Nordrhein-Westfalens erfolgen. Auf diese Weise entstände ein Pool hochwertiger Prüfungsfragen.

1.3 Zielsetzung

Die Ziele dieser Arbeit sind:

- Die Ausarbeitung der Workflows für das Prüfungsmanagement
- Die Ausarbeitung der Workflows für das Reviewsystem
- Die Implementierung des Prüfungsmanagements
- Die Implementierung des Reviewsystems

Die Gesamtheit des hier vorgestellten Systems wird als **E**lektronisches **P**rüfungsmanagement in der **M**edizin, kurz EPM, bezeichnet.

1.4 Aufbau der Arbeit

Zunächst werden die Motivation und die Zielsetzungen für diese Arbeit erklärt. Im zweiten Kapitel werden die technischen und fachlichen Grundlagen gelegt. Das dritte Kapitel beschreibt die Anforderungen an das System und die Technologieauswahl, während im vierten Kapitel die Implementierung des Prüfungsmanagements erläutert wird. In der abschließenden Diskussion wird das EPM-System mit ähnlichen Systemen verglichen. Dabei werden Vor- und Nachteile betrachtet und ein möglicher synchroner Einsatz der jeweiligen Systeme diskutiert. Es wird evaluiert, inwiefern die Anforderungen an das System erfüllt wurden, wie hoch sein Nutzen ist und welche Erweiterungsmöglichkeiten und Zukunftsperspektiven es bietet. Hier wird auch auf die bei der Realisierung des Systems aufgetretenen Probleme und Herausforderungen eingegangen.

2 Grundlagen

2.1 Technische Grundlagen

2.1.1 XML

Bei der Extensible Markup Language (XML) handelt es sich um einen vom World Wide Web Consortium (W3C, vgl. [89]) verfassten Standard für maschinen- und menschenlesbare Dokumente. Ein XML-Dokument besitzt einen baumartigen Aufbau, die Struktur und Semantik, jedoch keine Formatierung beschreibt (vgl. [17], S 32).

Abbildung 2.1 beschreibt die Stammdaten einer Person. Die erste Zeile besteht aus der XML-Deklaration, in der in diesem Fall die XML-Version und die Zeichenkodierung angezeigt werden. Die Deklaration ist optional, jedoch empfiehlt es sich, diese anzugeben, damit ein XML-verarbeitender Prozess das Dokument zu verarbeiten weiß. (vgl. [61], S 3). Die Strukturierung der Daten erfolgt durch sogenannte Tags. Ein Tag besteht aus einer Zeichenkette, die durch spitze Klammern begrenzt wird. Dabei wird zwischen Anfangs- und Endtags unterschieden. Zu jedem Anfangstag wie `<Vorname>` existiert genau ein Endtag, der bis auf einen der

```
<?xml version="1.0" encoding="utf-8"?>
<Stammdaten>
        <Vorname>Wolfram</Vorname>
        <Nachname>Urich</Nachname>
</Stammdaten>
```

Abbildung 2.1: Beispiel einer XML-Datei

ersten Klammer folgenden Slash dem Anfangstag entspricht (`</Vorname>`). Die Tags umschließen entweder eine Anzahl von Tags oder die eigentlichen Daten. Die Tag-Bezeichnungen sollten so gewählt sein, dass sich die Bedeutung der enthaltenen Daten aus ihnen erschließt. Einem Paar `<Beispieltag></Beispieltag>` entspricht der Empty-Element-Tag `<Beispieltag />`.

Die Struktur einer XML-Datei kann mittels einer DTD-Datei (Dokumenttype-definition-Datei) oder eines XML-Schemas (XSD) festgelegt werden, wodurch eine maschinelle Syntaxprüfung einer XML-Datei ermöglicht wird. Bei einem XML-Schema handelt es sich um eine vom W3-Konsortium erarbeitete Weiterentwicklung des DTD-Standards, der es unter anderem erlaubt, Daten einen Datentyp zuzuweisen oder die Länge und die Syntax einer Zeichenkette zu bestimmen (vgl. [13], S 81–82).

XML bietet zahlreiche Vorteile gegenüber anderen Formaten. XML-Dokumente können leicht in andere XML-Formate oder Objekte objektorientierter Programmiersprachen transformiert werden. Eine XML-Datei ist vergleichsweise robust gegenüber Beschädigungen, da es sich um eine Textdatei handelt, die auch in Fragmenten noch lesbar ist.

Neben der oben genannten Literatur sei auf [18], [2] und [58] verwiesen, wobei die letztgenannten Bücher auf das Zusammenspiel von XML und Java eingehen.

2.1.2 Datenbanksysteme

Ein Datenbanksystem (DBS) besteht aus der Datenbank, in der die Daten abgelegt werden, und dem Datenbankmanagementsystem (DBMS). Zu den Aufgaben eines DBMS zählt neben der sicheren Speicherung der Daten die Verarbeitung von Kommandos zur Abfrage, Analyse und Sortierung von Daten (vgl. [34], S 34). Im Allgemeinen wird zwischen drei Arten von DBMS unterschieden: Relationale, objektorientierte und objektrelationale Datenbankmanagementsysteme. In relationalen Datenbanken sind die Daten als Tabellen strukturiert. Es sind Verweise von einer Tabelle zu einer anderen möglich. Ein objektorientiertes DBMS ist in der Lage, Objekte direkt zu speichern, ohne dass diese in Tabellen zerlegt werden müssen. Methoden eines Objektes können direkt im DBMS implementiert wer-

den. Das Objektrelationale DBMS stellt eine Mischform aus objektorientiertem und relationalem DBMS dar (vgl. [65], S 11). Die Abgrenzung eines relationalen DBMS gegenüber einem objektrelationalen DBMS fällt in der Praxis oft schwer, da einige Teile objektrelationaler DBMS Einzug in nativ relationale DBMS erhalten haben. Objektrelationale DBMS lassen sich am ehesten so umschreiben, dass sie versuchen, wichtige Prinzipien der Objektorientierung mit den Mitteln relationaler DBMS zu lösen.

Bei MySQL und Postgres handelt es sich um zwei der weitverbreitetsten freien DBMSe. Zu den bekanntesten und größten kommerziellen DBMSen gehören DB2 von IBM und Oracle. Alle vier System wandelten sich von ihrer ursprünglichen relationalen Form hin zu objektrelationalen Datenbanksystemen, die objektorientierte Prinzipien in unterschiedlich starker Form unterstützen. Das Prinzip der objektorientierten DBMS wird beispielsweise in den Produkten ObjektStore, O2 und Caché realisiert. Trotz stetig zunehmender Verbreitung objektorienter Sprachen wie Java fristen objektorientierte DBMS weiterhin ein Nischendasein (vgl. [34], S 35).

2.1.3 Aspektorientierte Programmierung

Bei der objektorientierten Programmierung werden Teile der realen Welt in Form von Klassen, die miteinander in Relation stehen, in einer geeigneten Software-Architektur abgebildet. Entitäten wie Kunde oder Auftrag werden in entsprechende Klassen gekapselt. (vgl. [7], S 16). Dieses Modell der realen Welt wird im Laufe der Implementierung immer weiter verwässert, da Punkte wie Logging, Autorisierung und Sicherheit in viele Klassen Einzug erhalten und sich quer durch die gesamte Systemarchitektur (vgl. Abschnitt 2.1.5) ziehen. Diese Verteilung macht das System anfällig für Fehler. Möchte man z. B. alle Datenbankzugriffe mitprotokollieren, muss in der objektorientierten Programmierung an sämtlichen Stellen, an denen ein Zugriff gemacht wird, eine entsprechende Logging-Methode aufrufen werden. Der Entwickler muss neben der Geschäftslogik viele andere Dinge im Auge behalten, was zu einer signifikanten Verminderung der Produktivität führt. Insgesamt leidet die Code-Qualität und der erstellte Code kann schlechter wiederverwendet werden (vgl. [37]). Hier setzt das Paradigma der aspektorientierten

```
1 public aspect DBLogging {
2     pointcut logInsert():
3         execution(public * CRUDOperations.insert*(..));
4
5     before(): logInsert(){
6         System.out.println("Data inserted into DB.");
7     }
8 }
```

Abbildung 2.2: Beispiel eines Aspektes in AspectJ.

Programmierung (AOP) an. Mit ihrer Hilfe ist es möglich, Gesichtspunkte wie Logging oder Persistenz, sogenannte Crosscutting Concerns, an einer zentralen Stelle zu implementieren und zu verwalten. Zur Implementierung der Logging-Funktionalität bei Datenbankzugriffen muss nicht an vielen Stellen die Logging-Methode aufgerufen werden, es genügt die Definition an einer Stelle (vgl. das weiter unten beschriebene Beispiel in Abbildung 2.2).

Die aspektorienterte Programmierung ist eine Erweiterung des objektorientierten Ansatzes. Der Übergang von der strengen Objektorientierung zur Aspektorientierung wird oft mit dem Wechsel von der prozeduralen zur objektorientierten Programmierung verglichen. Die aspektorientierten Sprachen (wie AspectJ für Java, AspectC++ für C++ oder aoPHP für PHP) gelten als nächste Epoche der Programmiersprachen (vgl. [7], S 3–7).

Ein wichtiger Begriff in der AOP ist der Begriff des Joinpoints. Ein Joinpoint ist ein Punkt im Programm, der durch zusätzlichen Quellcode erweitert oder sublimiert werden kann. Beispiele für Joinpoints sind Aufrufe von Methoden, das Exception-Handling[1] oder der Zugriff auf eine Variable. Mit Hilfe von Pointcuts definiert der Programmierer eine Menge von einem oder mehreren Joinpoints, die er mit einer bestimmten Aktion verknüpfen will. Es ist beispielsweise möglich, nach jedem Aufruf einer Methode einer Meldung in der Konsole auszugeben. Die Aktion, in diesem Fall die Konsolenmeldung, wird in der Aspektorientierung als Advice bezeichnet. Über Introduction ist es möglich, einer Klasse zusätzliche

[1] Bei vielen Programmiersprachen wird in Fällen, in denen ein Fehler auftritt, eine sogenannte Exception geworfen. Diese kann direkt abgefangen oder an einen anderen Programmteil weitergereicht werden.

Attribute und Methoden hinzuzufügen ohne den Quellcode der Klasse an sich zu modifizieren. In der AOP spielt ein Aspekt eine ähnliche Rolle wie eine Klasse in der Objektorientierung (vgl. [14]). Ein Aspekt bildet den Container für Pointcuts, Advices und Introductions. Die Aspekte müssen in den vorhandenen Quellcode „eingewebt“ werden, d.h. der Programmcode wird an den entsprechenden Stellen um die in den Aspekten definierten Advices erweitert. Diese Aufgabe übernimmt der Aspect Weaver.

Abbildung 2.2 enthält ein einfaches, in AspectJ geschriebenes Beispiel eines Aspektes. Der Aspekt `DBLogging` enthält den Pointcut `logInsert()` (Z 2–3), der eine Menge von Joinpoints definiert: Der Pointcut enthält sämtliche öffentliche (`public`) Aufrufe von Methoden der Klasse `CRUDOperations`, die mit `insert` beginnen, beliebig viele Parameter und einen beliebigen Rückgabewert besitzen. Der erste Stern (»`*`«) dient als Platzhalter für den Rückgabewert der Methode, der zweite steht für beliebig viele Zeichen, die auf die Zeichenkette `insert` folgen. Die zwei Punkte (»`..`«) dienen als Platzhalter für keine oder eine endliche Anzahl von Übergabeparametern. In den Zeilen 5-7 ist ein `Before`-Advice definiert, der vor jedem Joinpoint, der durch `logInsert()` erfasst ist, ausgeführt wird und zu einer entsprechenden Konsolenmeldung (Z 6) führt. Diese acht Codezeilen erlauben ein einfaches Protokollieren sämtlicher Einfüge-Operationen der Klasse `CRUDOperations`.

2.1.4 Java

Java ist eine von der Firma SUN Microsystems entwickelte objektorientierte Programmiersprache, die in verschiedenen Bereichen wie Standalone-Applikationen, verteilten Systemen oder Webanwendungen zum Einsatz kommt.

Java existiert in drei unterschiedlichen Varianten: Der Java 2 Platform Standard Edition 5.0 (J2SE), der Java Enterprise Edition (Java EE) und der Java Platform Micro Edition (Java ME). Die J2SE stellt die Hauptversion von Java dar, die vor allem auf Desktop-Rechnern zum Einsatz kommt. Java EE[2] fügt auf der Basis

[2]Vor der Einführung der Java-Version 5.0 trug die Java Enterprise Edition den Namen Java 2 Platform Enterprise Edition (J2EE).

von J2SE Erweiterungen für serverseitige Anwendungen hinzu, die hohe Anforderungen an Skalierbarkeit und Robustheit der Software stellen. Insbesondere können mit Hilfe der Java EE Webapplikationen realisiert werden. Bei der J2ME handelt es sich um eine Version von Java, die für Systeme mit geringen Resourcen konzipiert wurde und beispielsweise auf PDAs, Smartphones oder Mobiltelefonen zum Einsatz kommt (vgl. [62], S 15 und [73]).

Um Java den Zugriff auf Datenbanken zu ermöglichen, existiert das von SUN eingeführte Paket JDBC. JDBC steht für Java Database Connectivity und ist eine Schnittstelle, die auf eine weitgehende Abstraktion vom verwendeten Datenbanksystem abzielt. Durch die Verwendung von JDBC kann der Programmierer auf einheitliche Weise auf die von unterschiedlichen Datenbanksystemen bereitgestellten Funktionalitäten zurückgreifen (vgl. [73]).

Zum Einstieg in die objektorientierte Programmierung mit Java eignen sich neben [62] besonders [12] und [82], welche im Internet frei zugänglich sind. Ausführliche Erklärungen aller Neuerungen der Java-Version 5.0 gegenüber der Vorgängerversion 1.4 finden sich in [43].

Die von SUN erarbeitete Java EE-Spezifikation definiert insbesondere Standards für Java-basierte Webapplikationen. Die Spezifikation gibt einen Rahmen zur Entwicklung verteilter, mehrschichtiger, komponentenbasierter Anwendungen vor (vgl: [41], S. 22). In der Literatur existiert keine einheitliche Definition des Begriffs „Komponente“ (vgl. [95], S 33 und [96], S 21–22). Zimmermann und Beneken liefern in [95] auf S 34 folgende Beschreibung des Komponenten-Begriffs: „Durch eine Komponente werden logisch zusammenhängende Funktionen bereitgestellt, um bestimmte Funktionalitäten oder Dienste zu realisieren. Diese Dienste sind in sich abgeschlossen und haben eine wohldefinierte Schnittstelle, damit sie von anderen Komponenten benutzt werden können“. Eine komponentenbasierte Softwarearchitektur erlaubt es, Komponenten unabhängig voneinander zu entwickeln und gemeinsam in ein Softwaresystem zu integrieren. Das Java EE-Programmiermodell unterscheidet die drei Komponentenarten Clientkomponenten, Web-Komponenten und Enterprise JavaBeans-Komponenten (vgl. [80], S 3).

Die Clientkomponenten stellen primär eine grafische Benutzeroberfläche bereit. Sie sind als eigenständige Applikationen oder als sogenannte Applets realisiert.

Ein Applet ist ein kleines Java-Programm, das in einem Webbrowser läuft und nur eingeschränkte Rechte auf die Systemresourcen besitzt.

Servlets und JavaServer Pages (JSPs) sind die wichtigsten Web-Komponenten. Servlets sind Java-Klassen, die Web-Anfragen entgegennehmen und Antworten generieren. Servlets empfangen beispielsweise Anfragen von Webbrowsern und generieren HTML-Ausgaben (vgl. [94], S 302). Bei JavaServer Pages handelt es sich um Web-Seiten mit eingebettetem Java-Code (vgl. [94], S 312), die intern in Servlets umgewandelt werden. Der Vorteil einer JSP gegenüber einem Servlet liegt darin, dass sie die Verwendung von Tags erlaubt und nicht jede HTML-Ausgabe mittels eines „out.println()"-Befehls erfolgen muss. Dies erleichtert Programmierung und Übersichtlichkeit. Eine weitere Vereinfachung wurde durch die Einführung der JavaServer Pages Standard Tag Library (JSTL) eingeführt. Durch diese Sammlung von Tags entfällt für den Webdesigner die Notwendigkeit, Java-Code in seine JavaServer Pages einbetten zu müssen (vgl. [88], S 42). Es exitieren z. B. Tags für Schleifen, Datenbankzugriffe und Fallunterscheidungen. Durch die Verwendung der JSTL-Tags wird die Übersichtlichkeit erhöht und der Code besser lesbar. In Verbindung mit der Entwicklung der JSTL wurde die Expression Language (EL) entwickelt. Es handelt sich hierbei um eine Sprache, mit deren Hilfe sehr einfach auf Objekte und deren Eigenschaften zugegriffen werden kann. Sie erlaubt überdies einfache arithmetische und logische Operationen (vgl. [80], S 108). Abbildung 2.3 zeigt ein Beispiel, bei dem iterativ die Attribute Vorname und Nachname der verschiedenen Personen einer Liste von Personen (`personList`) durchlaufen werden. Oben ist dabei die Ausgabe mittels herkömmlichen Javacodes, unten die unter Verwendung von JSTL und EL dargestellt. Auf den Nachnamen eines Objekts vom Typ Person wird nicht mittels `person.getNachname()`, sondern mittels `person.nachname` zugegriffen. Dies ist insbesondere bei tief verschachtelten Objekten von Vorteil. Bezüglich Länge und Übersichtlichkeit ist die zweite Möglichkeit der ersten überlegen.

Es existieren drei verschiedene Arten von Enterprise JavaBeans (EJBs)[3]: Entity, Session und Message Driven Beans. Die Verwendung der einzelnen Komponenten lässt sich gut mittels eines Beispiels verdeutlichen (vgl. [52], S 15–16). Ein Enti-

[3]Enterprise JavaBeans haben außer der Namensverwandtschaft nichts mit JavaBeans gemein (vgl. [96], S 33). Bei JavaBeans handelt es sich um Java-Klassen, auf deren Attribute mittels Get- und Set-Methoden zugegriffen wird.

```
<ul>
  <%
  for (Person person: personList){
  %>
    <li>
      <%=person.getVorname()%> <%=person.getNachname()%>
    </li>
    <%
    }
    %>
</ul>

<%"= ------------------------------------------ --%>

<ul>
  <c:forEach var="person" items="${personList}">
    <li>
      ${person.vorname} ${person.nachname}
    </li>
  </c:forEach>
</ul>
```

Abbildung 2.3: Ausgabe von Personendaten in einer JSP. Oben mittels Javacode, unten mittels JSTL und EL.

ty Bean repräsentiert ein Objekt der realen Umwelt. Bei einer Bootsvermietung können z. B. ein Schiff-Bean, ein Kabinen-Bean und ein Kunden-Bean existieren. Technisch gesehen werden die verschiedenen Objekte durch das Entity Bean in einer Datenbank persistiert. Entity Beans bilden eine zusätzliche Schicht vor der Datenbankschicht, über welche die in einem Datenbanksystem gespeicherten Daten manipuliert werden können. Der Vorteil liegt darin, dass direkt auf Objekten gearbeitet wird. Dazu existiert eine an die Standard Query Language (SQL) angelehnte Abfragesprache, die Enterprise JavaBeans Query Language (EJB QL), mit deren Hilfe Anfragen direkt auf Objekten ausgeführt werden. Die Abfragesprache ist nicht an ein spezielles Datenbanksystem gebunden. Die Daten können sowohl in einem relationalen als auch in einem objektorientierten Datenbanksystem gespeichert werden. (vgl. [3], S 815). Session Beans übernehmen oft die Koordination zwischen verschiedenen Objekten. Bei einer Reservierung spielen mehrere Entity Beans wie Schiff-Bean, Kabinen-Bean oder Kunden-Bean eine

Rolle und müssen in Abhängigkeit voneinander angesprochen werden. Diese Abstimmung übernehmen die Session Beans. Die Message Driven Beans stellen eine Erweiterung der Session Beans dar, bei der die beiden Kommunikations-Partner nicht aufeinander warten müssen und stattdessen parallel weiterarbeiten können (vgl. [67], S 212). Durch sie werden EJB-Systeme für asynchrone Kommunikation geöffnet. Über ein Hotel könnten online Bootsreservierungen getätigt werden. Ein Message Driven Bean würde die Reservierung entgegennehmen, diese mit Hilfe von Session und Entity Beans abarbeiten, sobald die entsprechenden Resourcen dies erlauben und anschließend das Reservierungssystem des Hotels über die Reservierung benachrichtigen. Während dieser Zeit kann die Hotel-Applikation weiter genutzt werden, ohne durch das Warten auf eine Bestätigung weiter blockiert zu sein.

Alle Komponenten der Java EE 5-Technologie sind in der Lage, Informationen innerhalb gewisser Kontexte auszutauschen. Es existieren der Application-, der Session-, der Request- und der Page-Kontext. Die Informationen innerhalb des Application-Kontextes stehen allen Komponenten jederzeit zur Verfügung. Hier werden Informationen gespeichert, die für alle Nutzer gleich sind und schnell zur Verfügung stehen sollen. Ein Beispiel sind Artikel innerhalb eines Webshops. Der Session-Kontext ermöglicht es einer Webanwendung, Informationen zwischen ihren Internetseiten auszutauschen. Eine Session beginnt beim Einloggen eines Nutzers und endet nach dem Ausloggen oder wenn der Nutzer eine zeitlang nicht mit der Applikation interagiert hat. Ein Beispiel für Session-Daten sind die Informationen über den gerade eingeloggten Nutzer. Die Daten innerhalb des Request-Kontextes sind nur bis zum Aufruf der nächsten Seite gültig. Die innerhalb des Page-Kontextes gespeicherten Informationen sind ausschließlich innerhalb der entsprechenden Internetseite verfügbar (vgl. [3], S 64f).

Spezifikationen und einleitende Tutorials bezüglich der verschiedenen J2EE-Komponenten finden sich neben den bereits genannten Quellen auf der Homepage von SUN ([73]). Hervorzuheben ist das Buch [10], welches einen sehr guten Einstieg in die Programmierung mit Hilfe der EJB-Technologie ermöglicht.

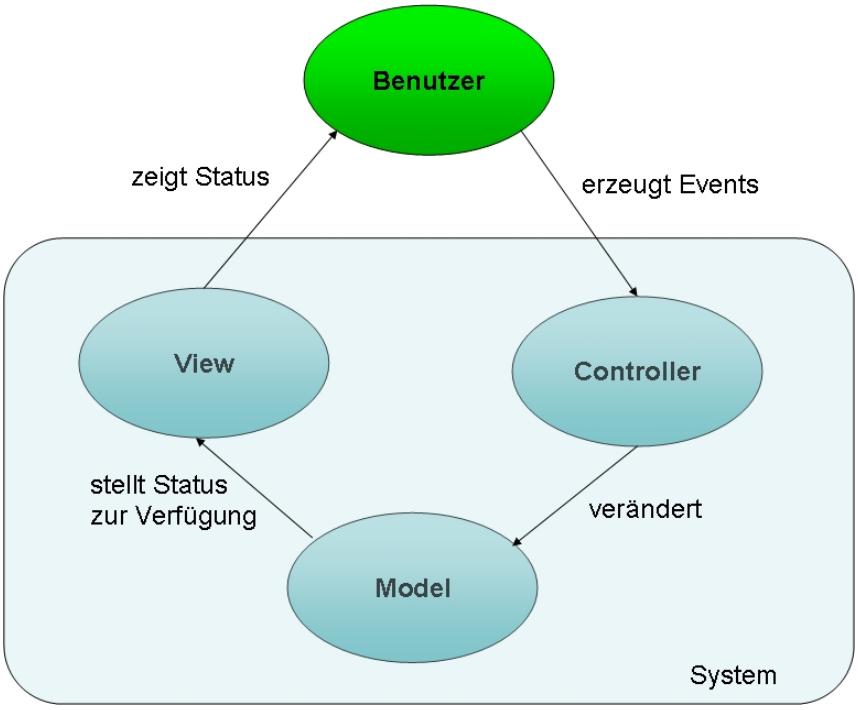


Abbildung 2.4: Funktionsweise der MVC-Architektur (nach [66])

2.1.5 Architektur

Softwareprojekte einer gewissen Größe erfordern eine genaue Planung des Aufbaus, der Architektur, des zu entwickelnden Systems. Dabei bilden spezielle Entwurfsmuster, die sich auf die Systemarchitektur beziehen, die sogenannten Architekturmuster, das Fundament eines guten Softwaresystems. Entwurfsmuster oder Pattern sind „ [...] Beschreibungen zusammenarbeitender Objekte und Klassen, die maßgeschneidert sind, um ein allgemeines Entwurfsproblem in einem bestimmten Kontext zu lösen.“ ([23], S 4).

Das Model-View-Controller-Architekturmuster (MVC) stellt ein wichtiges Architekturmuster dar. Es dient zur Separation von Daten (Model), Darstellung (View) und Kontrollfluss (Controller) (vgl. Abbildung 2.4). So sind einzelne Komponenten austauschbar. Das Model beinhaltet die Daten eines Programms, die in einer oder mehreren Views dargestellt und manipuliert werden können. Eine View kann beispielsweise eine Tabelle, ein Kuchen- oder ein Balkendiagramm sein. Ein Nutzer, der in einer View Daten verändert, erzeugt ein sogenanntes Event. Ein

Event wird z. B. durch das Drücken einer Maustaste oder das Drücken der Enter-Taste ausgelöst. Das Event veranlasst den zur jeweiligen View gehörenden Controller, das Model entsprechend der vom Nutzer vorgenommenen Veränderungen zu manipulieren. Sämtliche Views, welche die modifizierten Daten repräsentieren, werden daraufhin vom Model über die Änderung des Zustands informiert und aktualisieren ihre Darstellung. Damit das Model seine View-Objekte nicht kennen muss, müssen sich diese beim Model-Objekt registrieren. Die angemeldeten View-Objekte werden bei einer Zustandsänderung automatisch benachrichtigt und holen sich eigenständig die relevanten Datenänderungen des Models. Dieser Prozess wird durch das Beobachter-Muster realisiert. Dieses stellt das wichtigste Entwurfsmuster des MVC-Architekturmusters dar, das mehrere Entwurfsmuster in sich vereint (vgl. [90], S 19–25 und [23], S 5–8 und S 287–300).

MVC-2 ist eine Variation des klassischen MVC-Musters für Webanwendungen. Bei Webapplikationen kann das Model nicht von sich aus Änderungen an die Views übermitteln, da ein mit dem HTTP-Protokoll[4] arbeitender Server nur auf eine Anfrage (Request) von Seiten des Nutzers eine Antwort (Request) gibt (vgl. [30], S 43). Das MVC-2-Muster ist auch unter dem Namen Model-2-Architektur bekannt. Der Begriff Model-1-Architektur bezeichnet eine klassische Webapplikation, die ohne einen Controller realisiert ist. Solcherart realisierte Webanwendungen erwiesen sich jedoch bei zunehmender Komplexität als nur schwer wartbar. Aus diesem Grund wurde das MVC-2-Prinzip eingeführt (vgl. [92], S 26). Bei Java-Webapplikationen wird der Controller üblicherweise durch ein Servlet, das sogenannte Controller-Servlet umgesetzt. JavaServer Pages stellen die Views dar und das Model wird durch Objekte oder Enterprise JavaBeans repräsentiert. Der Zustand des Models wird üblicherweise in einer Datenbank persistent gespeichert.

2.1.6 Struts

Struts ist ein weit verbreitetes Open-Source-Framework, das die MVC-2-Architektur umsetzt. Es basiert auf durchdachten und vielfach erprobten Strukturen,

[4] HTTP steht für **H**yper**T**ext **T**ransfer **P**rotocol und ist ein Protokoll zur Übermittlung von Daten über ein Netzwerk. Obwohl auch andere Anwendungen möglich sind, wird es hauptsächlich eingesetzt, um Daten aus dem World Wide Web in einen Webbrowser zu laden. Die HTTP- Spezifikationen der Version 1.1 finden sich im RFC 2616 unter [20].

die Robustheit, Übersichtlichkeit und hohe Effizienz ermöglichen (vgl. [24], S 23). Hierbei spielen Entwurfsmuster eine wichtige Rolle.

Der Großteil der Konfiguration einer Struts-Applikation erfolgt in der XML-Datei `struts-config.xml`. Hier werden Mappings zwischen Aliasnamen und Klassen bzw. JSPs definiert. So können die Komponenten einer Anwendung leichter ausgetauscht werden (vgl. [92], S 22). Die Dateien `validator-rules.xml` und `validation.xml` dienen der Definition automatischer Validierungen. Die Einträge in diesen XML-Dateien veranlassen Struts, die vom Nutzer auf Webseiten eingegebene Formulardaten auf Richtigkeit zu überprüfen, optional auch clientseitig mittels JavaScript (vgl. Abschnitt 2.1.8). Diese grundlegende Arbeitserleichterung funktioniert jedoch nicht in allen Situationen, so dass in einigen Fällen eigene Fehlerroutinen geschrieben werden müssen. Die Textdatei `ApplicationResources.properties` dient zur Definition von Schlüsseln hinter deren Werten sich Texte wie Fehlermeldungen verbergen. Diese Datei kann leicht für neue Sprachen erweitert werden. Struts stellt somit ein effizienten Mechanismus für die Internationalisierung einer Website bereit (vgl. [92], S 22).

Die Verwendung der von Struts zur Verfügung gestellten Fnktionalitäten erfolgt mittels der Struts-Tag-Libraries. Mit ihrer Hilfe kann beispielsweise vereinfacht auf die Attribute einer Klasse zugegriffen werden, HTML-Code erzeugt oder es können logische Bedingungen geprüft werden. Einige dieser Operationen werden bereits durch die JSTL abgedeckt. Um die Kompatibilität zum JSTL-Standard zu gewährleisten, wurde das Struts EL-Projekt gegründet. Im Vergleich zu den ursprünglichen Struts-Tags-Libraries wurden die Tags entfernt, die bereits durch die JSTL abgedeckt werden (vgl. [78]). Kontrollstrukturen und der Zugriff auf Objektattribute erfolgen durch die Expression Language (vgl. Abschnitt 2.1.4). Die Struts EL ermöglicht den Einsatz von Struts-spezifischen Tags auf der einen und JSTL-Tags und EL auf der anderen Seite.

Tiles stellen eine Erweiterung des Struts-Frameworks dar, die auch ohne Struts, unter Einbuße einiger Features verwendet werden kann (vgl. [78]). Mit Hilfe des Tiles-Framework lassen sich Schablonen, sogenannte Templates, erstellen, die mit unterschiedlichen Inhalten gefüllt werden können. Bestehen die Internetseiten einer Webanwendung beispielsweise aus einer Kopfzeile, einem Inhaltsbereich, einer

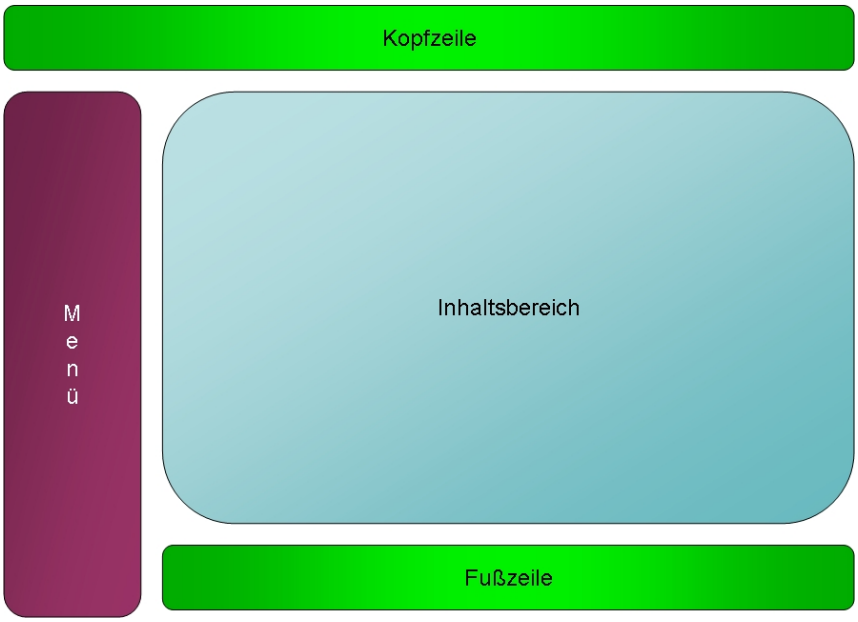


Abbildung 2.5: Mögliche Unterteilung einer Webseite

Fußzeile und einem Menü (vgl. Abbildung 2.5), so ist deren Position in einer zentralen JavaServer Page mit Hilfe von Platzhaltern definiert. Die Elemente der zentralen JSP, wie das Logo, finden sich auf sämtlichen Internetseiten der Applikation. In einer XML-Datei mit Namen `tiles-defs.xml` werden den Platzhaltern dann Zeichenketten (etwa für Kopf- oder Fußzeile) oder JSP-Seiten zugeordnet (z. B. im Inhaltsbereich oder Menü) (vgl. [16], S 303). Erfolgt eine Server-Anfrage, wird mit Hilfe dieser XML-Datei eine einzige JSP zusammengesetzt, die an den Browser zurückgesendet wird.

Neben der im Text genannten Literatur sei insbesondere auf [30] und [90] verwiesen. Die Website [26] bietet sehr gute zum Teil kostenpflichtige Struts-Beispielimplementierungen und -Tutorials. Zur Klärung des Zusammenspiels von Struts und JSTL bzw. EL, ist die Seite [25] sehr empfehlenswert. Sie stellt eine gute Ergänzung der Inhalte der Struts-Homepage ([78]) dar.

2.1.7 JavaServer Faces

Die JavaServer Faces (JSF) wurden entwickelt, um eine leichtere Implementierung anspruchsvoller Weboberflächen zu ermöglichen. Wiederverwendbarkeit, Kompatibilität und Standardisierung sind die Hauptvorteile der JSF. Mit JSF lassen sich Bedienoberflächen für Webanwendungen aus vorgefertigten Komponenten konstruieren. Die Darstellung vieler Komponenten erfolgt unter Verwendung von JavaScript (vgl. 2.1.8). Es besteht die Möglichkeit, eigene Komponenten zu entwickeln und zu benutzen. Neben der Referenzimplementierung von SUN, der Sun JavaServer Faces Implementation (RI) ([73]), existieren freie und kommmerzielle Implementierungen des Standards wie die Apache MyFaces ([79]) oder die Oracle Application Development Framework Faces (ADF) ([59]). Vielfach wird kontrovers diskutiert, ob die JSF Struts als Standard-Framework für Webapplikationen ablösen werden (vgl. z. B. [74] oder [24], S 25-26). Dabei wird außer Acht gelassen, dass Struts und JSF eine gemeinsame Schnittmenge besitzen, aber dennoch andere Schwerpunkte haben und sich gegenseitig ergänzen. Während das Hauptaugenmerk der JSF klar auf der Darstellung der Daten im Browser liegt, wird durch Struts die Architektur der gesamten Webapplikation festgelegt (vgl. [4], S 8-9). Ein tabellarischer Vergleich der beiden Frameworks findet sich unter [21]. Bislang besitzt Struts noch eine klare Vorrangsstellung. Es bleibt abzuwarten, ob sich ein Framework in der Zukunft durchsetzen wird.

2.1.8 JavaScript und JScript

JavaScript ist eine von Netscape entwickelte Skriptsprache, die sowohl clientseitig als auch serverseitig ausgeführt werden kann (vgl. [8] S 15 und [69], S 35). Vornehmlich wird diese Sprache auf Clientseite eingesetzt, um Elemente einer im Browser angezeigten Webseite zu manipulieren. Der Vorteil von JavaScript gegenüber serverseitig ausgeführten Sprachen liegt darin, dass keine erneute Anfrage an den Server geschickt werden muss, wenn der Inhalt einer Webseite vom Nutzer bearbeitet wird. Trotz der Namensähnlichkeit ist JavaScript nicht mit der Programmiersprache Java zu verwechseln. Bei der Namensgebung von JavaScript spielten primär Marketinggründe eine Rolle und nicht die technische Verwandt-

schaft der beiden Programmiersprachen (vgl. [83]). JScript ist das von Microsoft entwickelte Gegenstück zu Netscapes JavaScript. Prinzipiell bieten beide Sprachen die gleichen Möglichkeiten, wobei JScript in einigen Punkten umfassender und ausgereifter als JavaScript ist (vgl. [93]).

Der Einsatz von JavaScript wird in vielen Webapplikationen aus Sicherheitsgründen abgelehnt. Mit JavaScript ist es möglich, das Eingabefenster von vertrauenswürdigen Webseiten zu simulieren, wodurch sensible Daten wie Nutzernamen, Passwörter oder Kreditkarteninformationen abgefangen und an den Angreifer übertragen werden können. Mittels JavaScript kann die Statusleiste des Webbrowsers verändert werden, so dass die dort angezeigte Adresse nicht der tatsächlich verlinkten Webseite entspricht. Unter Umständen erhält der Angreifer vollständigen Zugriff auf einen betroffenen Rechner (vgl. [9]). Grundlegende Informationen und Dokumentationen bezüglich JavaScript und JScript finden sich unter [51], [53] und [54]. Das Buch [68] bietet in FAQ-Form umfassende Lösungen zu Problemen, die mit JavaScript gelöst werden können.

2.1.9 CADS

Der Name CADS steht für Central Authentification and Directory Service und ist ein am Institut für Medizinische Informatik und Biomathematik der WWU Münster entwickeltes Single-Sign-On-System. Mit Hilfe dieses Systems kann ein Nutzer mit nur einer Nutzername/Passwort-Kombination auf verschiedene campusinterne Dienste zugreifen. Jede an den CADS angebundene Applikation kann die zentral gespeicherten Stammdaten der CADS-Nutzer erfragen. Diese werden in Abstimmung mit dem Studierendensekretariat und dem IfAS[5] auf dem aktuellen Stand gehalten.

Die Kerntechnologie des CADS-Systems basiert auf der Verwendung von Java Servlets. Der CADS kommuniziert mit den verschiedenen Anwendungen mittels XML. Die Nutzung des CADS-Dienstes wird durch verschiedene Versionen einer Softwarebibliothek mit Namen CADSLib ermöglicht. Zurzeit existie-

[5] IfAS steht für **I**nstitut **f**ür **A**usbildung und **S**tudienangelegenheiten und ist eine Einrichtung der Medizinischen Fakultät Münster

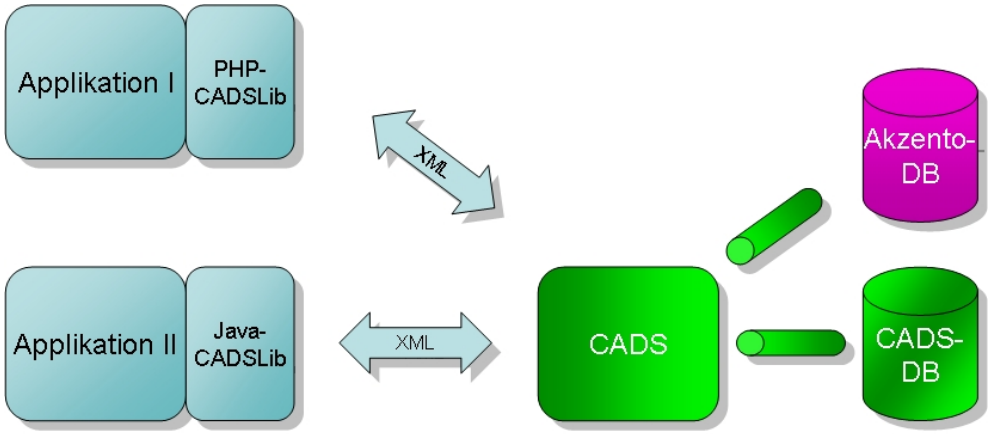


Abbildung 2.6: Funktionsweise des CADS-Systems

ren CADSLibs für PHP- und Java-Applikationen, wobei die Java-CADSLib in jeweils einer Variante für Standalone- und Webanwendungen vorliegt. Prinzipiell kann eine CADSLib für jede beliebige Programmiersprache implementiert werden, was eine größtmögliche Flexibilität und Offenheit des Systems garantiert. Die CADSLib stellt Funktionen/Methoden bereit, die aus dem Programm heraus aufgerufen werden. Diese hüllen die übergebenen Daten in einen XML-String, welchen sie an das entsprechende CADS-Servlet übergeben (vgl. Abbildung 2.6). Das Servlet bearbeitet die Abfrage unter Zugriff auf die eigene Datenbank und schickt die Antwort als XML-Zeichenkette an die Bibliothek zurück. Nach dem Herausfiltern der relevanten Daten aus dem XML-String liefert die Funktion/Methode der CADSLib das entsprechende Ergebnis. Der CADS-Dienst bietet ferner die Möglichkeit, auf die Datenbank des Akzento-Systems zuzugreifen. Akzento ist ein an der Medizinischen Fakultät der WWU Münster eingesetztes Softwaresystem, über die eine Reihe von Daten gepflegt werden. Das Zugriffskonzept mittels CADS bietet einige sicherheitstechnische Vorteile gegenüber der direkten Kommunikation einer Applikation mit der Akzento-Datenbank.

2.2 Fachliche Grundlagen

2.2.1 Prüfungsfragetypen

Es existieren eine Vielzahl von Formaten für Prüfungsfragen. Der bekannteste und in der Medizin am weitesten verbreitetste Typus ist die Multiple-Choice-Frage (MC-Frage). Hierbei muss sich der Prüfling aus einer Auswahl von M Antwortmöglichkeiten für die N Antworten entscheiden, die seiner Meinung nach richtig bzw. falsch sind. Diese Art der Fragen werden im Folgenden als M-aus-N-MC-Fragen bezeichnet. 1-aus-5-MC-Fragen gelten durch die Verwendung in den vom IMPP bereitgestellten Prüfungsfragen als Quasi-Standard. IMPP steht für Institut für medizinische und pharmazeutische Prüfungsfragen (vgl. [31]). Das Institut unterstützt die Landesprüfungsämter bei der Durchführung der bundeseinheitlichen schriftlichen Prüfungen, entsprechend der Approbationsordnungen für Ärzte und Pharmazeuten. Eine zentrale Rolle kommt im Dienstleistungskatalog des IMPP der Erstellung von Prüfungsaufgaben zu. Bei den 1-aus-5-MC-Fragen unterscheidet man zwischen Fragen mit positiver Einfachauswahl und Fragen mit negativer Einfachauswahl. Bei ersteren gibt es unter den fünf Wahlantworten nur eine richtige oder beste. Bei den Fragen mit negativer Einfachauswahl steht der Prüfling vor der Aufgabe, die Antwortmöglichkeit zu wählen, die seiner Meinung nach die Ausnahme darstellt oder am wenigsten zutrifft. An der Medizinischen Fakultät der WWU Münster werden die 1-aus-5-Fragen neben Fragetext und Antwortmöglichkeiten im Bedarfsfall um einen sogenannten Fragestamm erweitert (vgl. [33]). Dieser beinhaltet oft eine Art Präambel mit klinischen Bezug wie beispielsweise „Eine 35jährige Patientin war wiederholt in Ihrer ärztlichen Praxis, um verschiedene Beschwerden im Magen-Darm-Bereich abklären zu lassen. Alle somatischen Befunde einschließlich einer gastroenterologischen Abklärung erbrachten keinen pathologischen Befund.“.

Einen anderen Fragentypus bilden die MEQ-Fragen. MEQ steht für Modified Essay Question und bezeichnet eine schriftliche Prüfung, die meist sechs Patientenfälle umfasst. Die Studierenden bearbeiten Seite für Seite eine klinische Fallgeschichte, zu der Fragen aus verschiedenen medizinischen Fachrichtungen gestellt werden. Die Fragen werden bewusst offen gestellt und müssen frei formu-

liert beantwortet werden. Der wesentliche Vorteil dieses Prüfungsformats ist die Tatsache, dass nicht nur Wissen, sondern auch die Fähigkeit der Studierenden, Probleme zu lösen, evaluiert wird (vgl. [40]).

Neben den hier vorgestellten Typen von Prüfungsfragen existieren noch viele weitere Fragetypen, auf die an dieser Stelle nicht weiter eingegangen wird, da sie für das EPM-System (noch) nicht relevant sind.

2.2.2 Approbationsordnung für Ärzte

Mit der neuen Approbationsordnung für Ärzte vom 27. Juni 2002 ([5]) wurde das Medizinstudium in Deutschland grundlegend reformiert. Zielvorgaben für die ärztliche Ausbildung sind die Verbesserung der praktischen Ausbildung, die Entwicklung klinischer Kompetenz, die Stärkung der allgemeinmedizinischen Fähigkeiten und die „[...] vertikale und horizontale Vernetzung der Fächer in Vorklinik und Klinik." ([81]) Weitere tiefgreifende Änderungen gegenüber dem bis dato gültigen Lehrplan sind themenbezogener Unterricht an Stelle der bisherigen Abhandlung einzelner Fächer und interdisziplinäre Seminare. Durch eine Ausweitung des Unterrichts am Krankenbett soll der Praxisbezug verstärkt werden (vgl. [86]).

Die neue Approbationsordnung wurde in Münster mit Beginn des Wintersemesters 2003/04 erstmals umgesetzt. Im Gegensatz zur alten Ordnung werden naturwissenschaftliche Grundlagen wie Physik, Chemie und Biologie bereits im ersten Semester unterrichtet (vgl. Abbildung 2.7).

Durch integrierte Seminare wird eine stärkere Verzahnung der Disziplinen Anatomie, Physiologie und Biochemie angestrebt und es werden erste klinische Bezüge hergestellt. Die Restrukturierung des Medizinstudiums betrifft vor allem den klinischen Teil. Anstelle klassischer Fächer wie Innerer Medizin oder Chirurgie werden Lerninhalte nun zum größten Teil themenzentriert in sogenannten Modulen vermittelt (vgl. [85]). Der themenbezogene Unterricht erfolgt durch Vertreter differierender Fachdisziplinen, die ein Thema, wie Atmung, Herz-Kreislauf oder die Medizin des Magen-Darm-Traktes jeweils aus Sicht ihrer Fachdisziplin beleuchten (vgl. Abbildung 2.8). Dies führt zu einem umfassenden, kompakten Wissen bezüglich eines Themas.

Studienplan für das 1. vorklinische Semester

Fach	Vorlesungen	Prakt. Übungen	Seminare	Integr. Seminare	Sem. mit klin. Bez.	Wahlfach	Gesamt
Chemie	4	3					7
Physik	4	3					7
Biologie	4	3					7
Anatomie	4	-					4
Med. Terminologie		2					2
EKM	1	2					3
BfE		1					1
	17	**14**					**31**

Abbildung 2.7: Studienplan des 1. vorklinischen Semesters

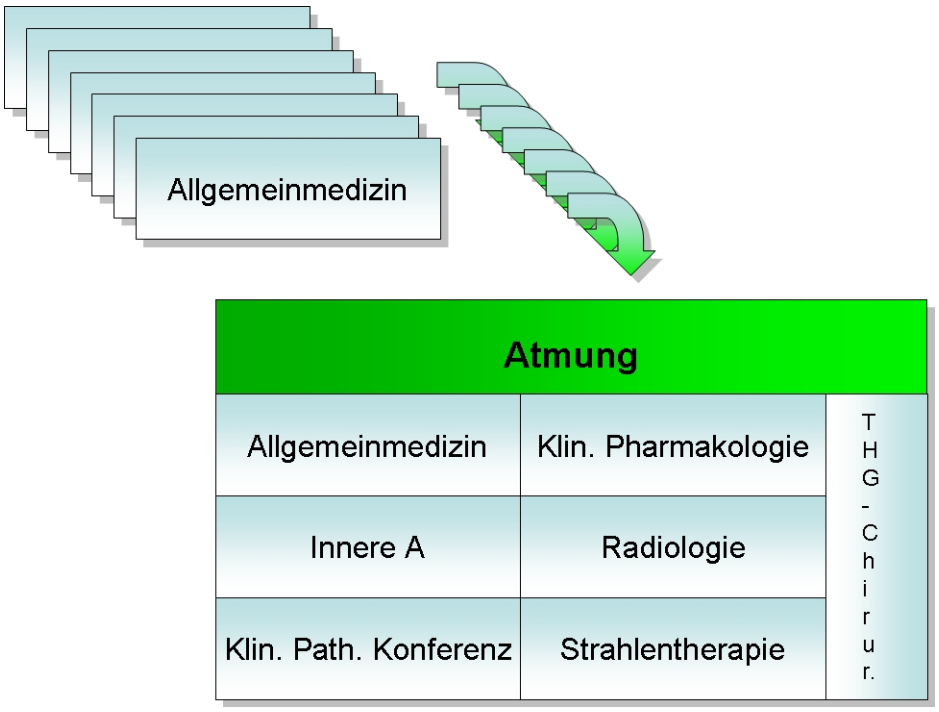


Abbildung 2.8: Aufbau des Moduls Atmung im Sommersemester 2006

Eine weitere Neuerung besteht in der Einführung eines zweiwöchigen Blockpraktikums in Krankenhäusern oder Arztpraxen, das die Studierenden am Ende des klinischen Abschnitts ableisten müssen.

Neben den Modifikationen bezüglich des inhaltlichen Aufbaus des Medizinstudiums wurden starke Eingriffe im Prüfungswesen vorgenommen. Die Anzahl der Examina wurde von vier auf zwei reduziert. Der erste Abschnitt der ärtzlichen Prüfung erfolgt nach dem vierten Semester, der zweite Teil durch das Staatsexamen nach dem Praktischen Jahr am Ende der Ausbildung. Gegenüber der alten Approbationsordnung fließt das Ergebnis der ersten Prüfung zu einem Drittel in die Abschlussnote ein. Eine weitere Modifikation besteht in der Einführung benoteter Leistungsnachweise. Für alle 39 erforderlichen Leistungsnachweise werden den Studierenden Noten ausgestellt, aus denen Zensuren für einzelne Fächer berechnet werden. Diese Noten werden später auf dem Examenszeugnis aufgeführt (vgl. [86]).

Die neue Approbationsordnung erforderte die Überarbeitung und Reorganisation des an der Medizinischen Fakultät der WWU Münster etablierten Prüfungsverfahrens (vgl. [85]). Zeitgleich wurde eine neue Studienordnung erarbeitet ([49]). Die wichtigste Neuerung des neuen Prüfungsverfahrens besteht in der Einführung einer gesonderten Abschlussklausur für jedes klinische Semester. In dieser werden den Studierenden Prüfungsfragen zu sämtlichen Modulen und Fächern aus jeweils einem klinischen Semester gestellt. Die Fragen sind 1-aus-5-MC-Fragen mit positiver oder negativer Einfachauswahl. Neben den Klausuren für Erstschreiber existieren Wiederholungsprüfungen, die in der Regel Inhalte aus verschiedenen klinischen Semestern beinhalten.

2.2.3 Bestehensgrenze, Ankerklausel und Nachteilsausgleich

Ein Fach[6] ist mit Umsetzung der neuen Approbationsordnung meist auf verschiedene Module verteilt. Zur Ermittlung der Note eines Leistungsnachweises werden die Prüfungsergebnisse sämtlicher Module, in denen das Fach aufgeführt ist, summiert. Um einen Leistungsnachweis zu erhalten, müssen mindestens 60 Prozent aller gewerteten Prüfungsfragen richtig beantwortet sein. Diese Grenze wird als Bestehensgrenze bezeichnet. Die Noten eines Leistungsnachweises ergeben sich aus dem Prozentsatz derjenigen Fragen, die über die Bestehensgrenze hinaus zutreffend beantwortet wurden. Ein Ergebnis im Intervall $[50, 75[$ führt beispielsweise zur Note „gut“. Ob und in welchem Umfang weitere Leistungen wie mündliche Prüfungen zur Erlangung eines Leistungsnachweises erbracht werden müssen, ist von den zuständigen Fachvertretern festgelegt (vgl. [45]).

Die Bestehensgrenze von 60 Prozent ist nicht statisch, sondern kann nach der sogenannten Ankerklausel heruntergesetzt werden. Die Ankerklausel an der Medizinischen Fakultät der WWU Münster ist angelehnt an entsprechende Regelungen von §14 Abs. 6 der neuen Approbationsordnung für Ärzte. Demnach wird die schriftliche Prüfungsleistung eines Studierenden als ausreichend bewertet, wenn der Prüfling mindestens 60 Prozent der gestellten Prüfungsfragen zutreffend beantwortet hat oder wenn die Zahl der vom Prüfling korrekt beantworteten Fragen um nicht mehr als 22 Prozent von den durchschnittlichen Prüfungsleistungen der Prüflinge abweicht, die zum ersten Mal an dieser Prüfung teilgenommen haben. Wurden in einem Semester beispielsweise 15 Fragen in der Semesterabschlussprüfung des 2. klin. Semesters (130 Studierende) und 15 Fragen in der Semesterabschlussprüfung des 3. klin. Semesters (135 Studierende) gestellt, so ergeben sich insgesamt 3975 gestellte Fragen ($130 * 15 + 135 * 15$). Im 2. Semester wurden 1495 richtige Antworten und im 3. klin. Semester 1685 richtige Antworten gegeben. Insgesamt wurden 3180 richtige Antworten erzielt, was einem Anteil von 80 Prozent entspricht ($3180/3975 * 100$). Gemäß der Ankerklausel können nun 22

[6] Es sei darauf hingewiesen, dass der Begriff „Fach“ im Kontext der Approbationsordnung für Ärzte eine andere Bedeutung hat. Die dort aufgeführten Fächer bilden zusammen mit den Querschnittsbereichen die Leistungsnachweise der Medizinischen Fakultät der WWU Münster. An der Medizinischen Fakultät der WWU Münster wird der Begriff Fach als Abkürzung für Fachdisziplin verwendet.

Prozent dieser durchschnittlichen Prüfungsleistung subtrahiert werden. Es ergibt sich eine Bestehensgrenze von 58 Prozent $(80 - 22)$ (vgl. [44]).

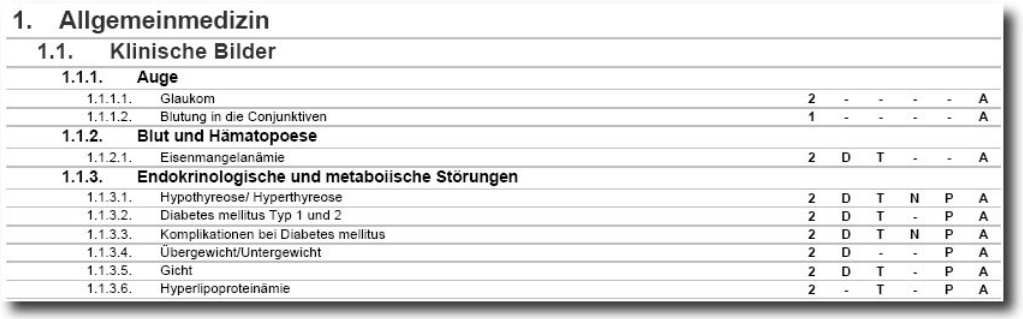
Damit Wiederholungsprüfungen nicht nur innerhalb der Grenzen eines kompletten Leistungsnachweises möglich sind, wurden Leistungsnachweise in verschiedene Teilleistungsnachweise aufgeteilt. Dieses Vorgehen ermöglicht eine portionsweise Wiederholung einzelner Teilgebiete (vgl. [46]).

Durch statistische Verfahren wird nach Absolvierung und Auswertung einer Klausur die Eignung von Fragen bestimmt. Auf dieser Grundlage und aufgrund anderer Unregelmäßigkeiten können Fragen aus der Wertung genommen werden. Damit die Studierenden, die diese Fragen richtig beantwortet haben, keine Verschlechterung ihres Ergebnisses erfahren, gibt es den sogenannten Nachteilsausgleich. Für diese Studierenden werden die Fragen weiterhin gewertet, so dass sie im Extremfall die Maximalpunktzahl übertreffen können.

2.2.4 Lernzielkatalog

Im Rahmen der neuen Approbationsordnung wurde an vielen Universitäten ein Lernzielkatalog erarbeitet und eingeführt. Dieser Katalog soll den Lehrenden als Orientierung für ihre jeweilige Fachdisziplin dienen. Die in den Lernzielkatalogen festgehaltenen Anforderungen stellen ein Rahmenprogramm dar, das als Grundlage für die Prüfung an den jeweiligen Hochschulen fungieren soll. Den Lehrenden wird dennoch die Freiheit gelassen, eigene Schwerpunkte und Gewichtungen zu setzen. Für die Studierenden beschreibt der Lernzielkatalog die zum Abschluss des Medizinstudiums erforderlichen ärztlichen Fähigkeiten (vgl. [39]).

Der in Anlehnung an eine Vorlage des Fachbereichs Medizin der Universität Hamburg entstandene Lernzielkatalog der Medizinischen Fakultät der WWU Münster mit Stand vom 01.02.2004 ([47]) gliedert sich wie folgt. Der Lernzielkatalog besteht aus vier hierarchisch angeordneten Ebenen. Auf der obersten Ebene befinden sich die Fächer. Sämtlichen Fächern sind die drei Abschnitte klinische Bilder, erweiterte Kenntnisse und Fertigkeiten untergeordnet. Innerhalb der Abschnitte finden sich verschiedene Themen, unter welchen sich schließlich die Lernziele befinden. Wie aus Abbildung 2.9 hervorgeht, ist das Lernziel Hyperlipoproteinämie



1.	**Allgemeinmedizin**						
1.1.	**Klinische Bilder**						
1.1.1.	**Auge**						
1.1.1.1.	Glaukom	2	-	-	-	-	A
1.1.1.2.	Blutung in die Conjunktiven	1	-	-	-	-	A
1.1.2.	**Blut und Hämatopoese**						
1.1.2.1.	Eisenmangelanämie	2	D	T	-	-	A
1.1.3.	**Endokrinologische und metaboiische Störungen**						
1.1.3.1.	Hypothyreose/ Hyperthyreose	2	D	T	N	P	A
1.1.3.2.	Diabetes mellitus Typ 1 und 2	2	D	T	-	P	A
1.1.3.3.	Komplikationen bei Diabetes mellitus	2	D	T	N	P	A
1.1.3.4.	Übergewicht/Untergewicht	2	D	-	-	P	A
1.1.3.5.	Gicht	2	D	T	-	P	A
1.1.3.6.	Hyperlipoproteinämie	2	-	T	-	P	A

Abbildung 2.9: Ausschnitt aus dem Lernzielkatalog der Medizinischen Fakultät der WWU Münster mit Stand vom 01.02.2004

dem Thema endokrinologische und metaboische Störungen untergeordnet. Dieses Thema fällt unter den Abschnitt klinische Bilder und gehört zum Fach Allgemeinmedizin. Eine Erklärung für die im Lernzielkatalog aufgeführten Kürzel findet sich unter [48].

3 Methoden und Konzepte

Das Kürzel EPM steht für **E**lektronisches **P**rüfungsmanagement in der **M**edizin. Das EPM-System lässt sich in drei Teilbereiche gliedern. Der erste Teilbereich befasst sich mit dem Prüfungssystem, welches zur Speicherung, Modifikation und Auswertung von Prüfungsdaten dient. Der zweite Teilbereich umfasst das Prüfungsmanagement. Dieses Modul beinhaltet einen Workflow, der es dem Studiendekanat erlaubt, über mehrere zwischengeschaltete Instanzen Prüfungsfragen erstellen zu lassen und im System abzuspeichern. Der dritte Teil umfasst das Reviewsystem, dessen Ziel es ist, durch eine Begutachtung der Prüfungsfragen eine signifikante Qualitätssteigerung zu erreichen.

3.1 Teilbereich I: Prüfungssystem

Das Prüfungssystem basiert auf der Prüfungsdatenbank[1], deren Struktur maßgeblich von einer Reihe von Anforderungen bestimmt wird, die sich aus der Umsetzung der neuen Approbationsordnung für Ärzte ergeben (vgl. Abschnitt 2.2.2). Die Datenbank wurde entworfen, um allen drei Teilbereichen des EPM-Systems gerecht zu werden. Eine Gesamtübersicht über die Datenbank-Struktur findet sich im Anhang. Im Folgenden werden nur die Tabellen und Relationen beschrieben, die für das Prüfungssystem von Belang sind. Auf eine Beschreibung der auf diese Datenbank aufsetzenden Programme wird verzichtet, da dies den Rahmen dieser Arbeit sprengen würde. Die Programme wurden größtenteils unter Nutzung der Programmiersprache PHP und mit Hilfe des Content-Management-Systems TYPO3 realisiert.

[1] Im Folgenden werden die Anforderungen an die Datenbank beschrieben wie sie mit Stand zum 12.10.2006 vorlagen und größtenteils vom Autor dieser Dissertation umgesetzt wurden.

Die Datenbank kann die Datenbestände verschiedener Universitäten aufnehmen. Sie ermöglicht es, dass verschiedene Universitäten auf dieselben Module, Fächer, Leistungsnachweise, Teilleistungsnachweise und Semester zurückgreifen, aber auch eigene Einträge hinzufügen können.

Sie ist in der Lage, Prüfungsfragen unterschiedlichen Typs zu verwalten und kann leicht um neue Fragetypen erweitert werden. Bisher werden lediglich 1-aus-5-MC-Fragen mit positiver und negativer Einfachauswahl (vgl. Abschnitt 2.2.1) gespeichert.

Eine Frage kann Klausuren von verschiedenen Universitäten, mehreren Semestern (z. B. WS 06/07) und mehreren Modul/Fach-Kombinationen zugeordnet sein. Ein Modul ist einem Fachsemester (z. B. 1. klinisches Semester) zugeordnet. Diese Zuweisung ist einerseits für die Prüfungsanmeldung von Belang, andererseits können so semesterweise Auswertungen erstellt werden. Ein Fach innerhalb eines Moduls ist ebenfalls einem Fachsemester zugeordnet. Diese Notwendigkeit ergibt sich aus der Anforderung von Wiederholungsprüfungen, die als Modul realisiert sind. Innerhalb eines solchen Wiederholungsmoduls werden oft Prüfungsinhalte eines einzigen Faches abgefragt. Die Inhalte stammen thematisch jedoch aus verschiedenen Fachsemestern, weshalb den Fächern innerhalb eines Moduls ebenfalls eine Semesterangabe zugewiesen ist. Eine Frage ist demnach einer oder mehreren Universität/Semester/Modul (Fachsemester)/Fach (Fachsemester)-Kombination(en) oder kurz U/S/M/F-Kombination(en) zugeordnet. Der eben beschriebene Zusammenhang bezieht sich auf die Zuweisung einer Frage zu einer Klausur und wird in einer entsprechenden Datenbanktabelle abgelegt. Dadurch kann eine Frage in verschiedenen Klausuren aus verschiedenen Univerisitäts/Modul/Fach/Semester-Kombinationen verwendet werden. Eine Frage wurde jedoch an genau einer Universität, in genau einem Semester, für genau eine Modul (Fachsemester)/Fach (Fachsemester)-Kombination erstellt. Die Angabe dieser U/S/M/F-Kombination findet sich direkt innerhalb des Datenbankeintrages einer Frage.

Es existiert eine weitere Tabelle, die der Tabelle für die Klausurerstellung entspricht. Diese wird dazu verwendet, eine Musterklausur zusammenzustellen. Erst nachdem die Klausurzusammenstellung genehmigt wurde, werden die Daten in die entsprechende Tabelle des Klausurkontextes verschoben. Auf diese Weise kön-

nen Autoren eine Frage so lange editieren bis die Klausur in ihrem finalen Zustand vorliegt (vgl. Abschnitt 3.2.7 d)).

Ein Leistungsnachweis setzt sich aus einem oder mehreren Teilleistungsnachweisen zusammen. Beispielsweise besteht der Leistungsnachweis Infektiologie/Immunologie aus den Teilleistungsnachweisen Transfusionsmedizin und Infektiologie. Die Teilleistungsnachweise sind wiederum an eine oder mehrere U/S/M/F-Kombinationen gekoppelt, da sich ihre Zusammensetzung von Universität zu Universität und von Semester zu Semester unterscheiden kann.

Eine weitere Anforderung an die Prüfungsdatenbank besteht darin, dass sich die Studierenden gezielt zu Prüfungen anmelden können. Die Anmeldung erfolgt auf Ebene der U/S/M/F-Kombinationen.

Um die Erweiterungsmöglichkeit für Fragen unterschiedlichen Typs zu gewährleisten, sind die allen Fragen inhärenten Attribute in der Tabelle Frage gekapselt. Die Attribute umfassen Autor, Erstellungsdatum, Fragetext und Vaterfrage. Eine Vaterfrage ist eine der Ursprungsfrage inhaltlich verwandte Frage. Beispielsweise können Frage und Vaterfrage nur hinsichtlich des Fragetextes voneinander abweichen. Durch die Vater-Kind-Relation ist diese Verwandtschaft auf einen Blick ersichtlich. In der Frage-Tabelle verweist ein Fremdschlüssel auf die U/S/M/F-Kombination im Rahmen derer die Frage erstellt wurde. Es existiert eine Kennzeichnung, ob die Frage auf Anregung von Seiten des Studiendekanats oder autark erstellt wurde. Eine Frage kann maximal einen Fragestamm besitzen. Dieser Fragestamm kann auch anderen Fragen zugewiesen sein, so dass sich mehrere Fragen einen Fragestamm teilen. Ihm können ein oder mehrere Medien zugeordnet sein. Ein Medium besteht zunächst aus Grafiken im png-, jpg-, gif- oder bmp-Format, kann aber um Dateitypen beliebigen Typs erweitert werden. Neben den MIME-Typen[2] ist in der Datenbank ebenfalls die Dateiendung des Mediums festgehalten. Auf diese Weise ist es z. B. möglich, einem Medium, welches dem MIME-Typ „img/jpeg“ zugewiesen ist, die Endung „.jpg“ zuzuordnen. Dies erleichtert die korrekte Darstellung eines Mediums in Browser oder Standalone-Applikation. Jedes Medium enthält eine Beschriftung, die nähere Informationen bezüglich des

[2]MIME steht für **M**ultipurpose **I**nternet **M**ail **E**xtensions und dient zur Deklaration von Inhalten in Internetprotokollen und E-Mails. Die genauen Spezifikationen sind im RFC 2045 unter [22] vermerkt.

jeweiligen Mediums beinhaltet. Bei Bildern ist dies üblicherweise eine Bildunterschrift. Neben dem Fragestamm können auch einer Frage beliebig viele Medien zugewiesen werden.

M-aus-N-MC-Fragen (vgl. Abschnitt 2.2.1) besitzen neben diesen allgemeingültigen Attributen N Antwortmöglichkeiten. Bei diesen ist jeweils vermerkt, ob sie eine der M Lösungen darstellen oder nicht. Neben dieser Angabe findet sich ein Feld, in dem bei Bedarf die Lösungsbegründung vermerkt ist. Die Lösungsbegründung kann durch Anhängen eines oder mehrerer Medien veranschaulicht werden.

Mehrere MEQ-Fragen sind einem Fall zugeordnet, da ein Fall im Allgemeinen aus mehreren Fragen besteht. Für jeden Fall ist vermerkt, auf welcher Seite und an welcher Position auf dieser Seite eine Frage zu finden ist. Für die Antwort existiert ein Textfeld.

Einer Frage können mehrere Lernziele zugewiesen werden, welche wiederum, wie in Abschnitt 2.2.4 beschrieben, einem Thema, einem Abschnitt und einem Fach zugeordnet sind. Die entsprechenden Datenbanktabellen unterliegen einer Versionierung, die sich an der Zeitdauer eines Semesters orientiert.

Zu jeder in einer Klausur verwendeten Prüfungsfrage können die Studierenden der Medizinischen Fakultät der WWU Münster einen Kommentar abgeben. Dieser wird in einer gesonderten Tabelle gespeichert.

Um den in Abschnitt 2.2.3 beschriebenen Anforderungen bezüglich Bestehensgrenze, Ankerklausel und Nachteilsausgleich gerecht werden zu können, existieren diverse Felder in verschiedenen Teilen der Datenbank. Desweiteren müssen Fragen gekennzeichnet sein, die zwar einer Klausur zugeordnet sind, aber von vornherein nicht in die Wertung miteinbezogen werden. Somit haben die derart markierten Fragen keinerlei Einfluss auf Bestehensgrenze, Anwendung der Ankerklausel und Nachteilsausgleich.

Einem Nutzer können innerhalb der Datenbankstruktur eine oder mehrere Rollen zugewiesen sein.

Die Datenbank ermöglicht die Implementierung einer Logging-Funktionalität.

Hierzu existiert eine im EPM-System angelegte Tabelle, in der die UserID, der Timestamp und der Loggingtyp gespeichert werden. Der Loggingtyp ist ein Fremdschlüssel auf eine weitere Tabelle und kann einen der Werte `INSERT`, `UPDATE`, `SELECT`, `VALIDATE` oder `INVALIDATE` annehmen. Während die ersten drei Werte die entsprechenden Datenbankoperationen kennzeichnen, beziehen sich die Einträge `VALIDATE` und `INVALIDATE` auf ein spezielles Feld, das jede Tabelle besitzt. Dieses „Gültig"-Flag kennzeichnet einen Eintrag als gültig oder nicht gültig. Einträge werden nicht gelöscht, sondern auf ungültig gesetzt. Dies erlaubt in Verbindung mit den MySQL-Logfiles die Wiederherstellung eines Systemzustandes nach einer Fehlbedienung oder Fehlfunktion der auf der Datenbank operierenden Software. Sowohl in der Datenbank als auch in den MySQL-Logfiles sind die Zeitpunkte der Datenbankoperationen vermerkt. Während die Logfiles die SQL-Anweisungen enthalten, ist in der Logging-Tabelle vermerkt, welcher Nutzer die entsprechende Operation auf Programmebene ausgeführt hat. Um die Einträge in der Logging-Tabelle auf ein Minimum zu reduzieren, werden nicht sämtliche SELECT-Anweisungen mitgeloggt, sondern nur jene, welche von besonderer Relevanz sind.

3.2 Teilbereich II: Prüfungsmanagement

Das Prüfungsmanagement bildet diejenigen Arbeitsabläufe ab, die nötig sind, um Prüfungsfragen durch Autoren erstellen zu lassen und in der Datenbank abzuspeichern. Im Folgenden werden die Anforderungen beschrieben, die an das Prüfungsmanagement gestellt wurden[3].

3.2.1 Rollensystem

Es wird zwischen fünf verschiedenen hierarchisch geordneten Rollen unterschieden. Es existiert die Rolle

[3] Zu Beginn des Projekts lagen nur grobe Vorstellungen hinsichtlich der Anforderungen und des benötigten Workflows vor. Die differenzierte Ausarbeitung oblag dem Autor dieser Dissertation.

- des Prüfungskoordinators,
- des Modulkoordinators,
- des Fachkoordinators,
- des an eine konkrete Modul/Fach-Kombination (genauer: U/S/M/F-Kombination, vgl. Abschnitt 3.1) gebundenen Autors und
- des an ein Fach gebundenen Autors.

3.2.2 Hierarchische Struktur der Rollen

Der Prüfungskoordinator ist verantwortlich für die Organisation der Prüfungen. Diese Rolle wird üblicherweise den Mitarbeitern des Studiendekanats zugewiesen. Der Prüfungskoordinator wählt das Modul bzw. Fach, für das er einen Modulkoordinator bzw. einen Fachkoordinator bestimmen möchte. Einem Fach wird automatisch ein Fachkoordinator zugewiesen, einem Modul wird jedoch nicht zwingend ein Modulkoordinator zugeordnet. Besteht ein Modul in dem ausgewählten Semester nur aus einem Fach, entfällt die Notwendigkeit der Instantiierung eines Modulkoordinators und der Prüfungskoordinator ernennt direkt einen Fachkoordinator. Gehört einem Modul in dem betreffenden Semester mehr als ein Fach an, wird ein Modulkoordinator bestimmt.

Ein Modulkoordinator kann für jedes der Fächer, aus denen sein Modul besteht, einen Fachkoordinator benennen. Die Aufgabe eines Modulkoordinators besteht darin, dafür zu sorgen, dass für sein Modul die vom Prüfungskoordinator geforderte Anzahl Fragen fristgerecht erstellt wird. Er ist bei Terminverzögerungen der erste Ansprechpartner des Prüfungskoordinators.

Ein Fachkoordinator wird entweder direkt vom Prüfungskoordinator eingesetzt oder von einem Modulkoordinator instantiiert. Die Aufgabe des Fachkoordinators ist es, die Autoren zum Erstellen von Prüfungsfragen auszuwählen und für die rechtzeitige Erstellung der Fragen in seiner U/S/M/F-Kombination Rechnung zu tragen. Werden die Prüfungsfragen nicht termingerecht erstellt, dient der Fachkoordinator als Kontaktperson zwischen Prüfungskoordinator bzw. Mo-

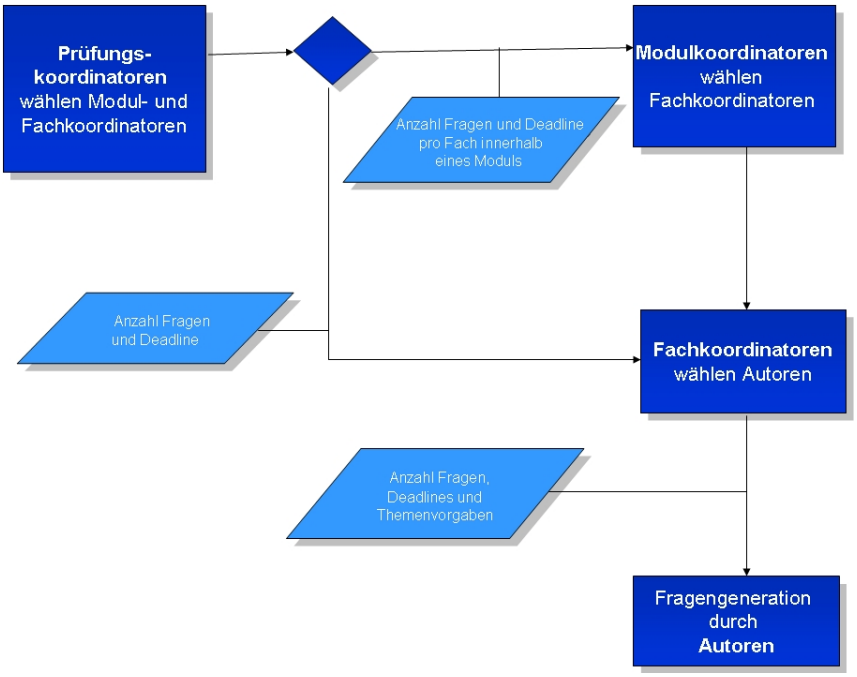


Abbildung 3.1: Prinzipieller Workflow des Prüfungsmanagements

dulkoordinator und Autoren.

Der an eine konkrete U/S/M/F-Kombination gebundene Autor hat die von ihm geforderte Anzahl an Prüfungsfragen fristgerecht zu erstellen.

Die an ein Fach gebundene Autor kann jederzeit beliebig viele Prüfungsfragen für das ihm zugewiesene Fach erstellen.

3.2.3 Workflow im Überblick

Das Diagramm in Abbildung 3.1 verdeutlicht den prinzipiellen Workflow zur Erstellung von Prüfungsfragen. Der Prüfungskoordinator wählt die Modulkoordinatoren bzw. direkt die Fachkoordinatoren. Wird ein Modulkoordinator bestimmt, wählt dieser die Fachkoordinatoren, welche ihrerseits die Autoren auswählen. Die Details werden in den folgenden Abschnitten beschrieben.

3.2.4 Funktionen für den Prüfungskoordinator

(a) Der Prüfungskoordinator soll in der Lage sein, Modul- bzw. Fachkoordinatoren auszuwählen. Bei der Ernennung eines Modulkoordinators teilt der Prüfungskoordinator diesem für jedes dem Modul inhärente Fach mit, wie viele Fragen bis zu welchem Zeitpunkt unter seiner Aufsicht zu erstellen sind. Einem vom Prüfungskoordinator gewählten Fachkoordinator werden ebenfalls Deadline und Anzahl der benötigten Fragen mitgeteilt.

(b) Es wird eine Statusübersicht für den Prüfungskoordinator benötigt, aus welcher hervorgeht, wie viele Prüfungsfragen für ein Fach bzw. Modul in einem Semester zu erstellen sind und wie viele bereits erstellt wurden. Bei allen Fächern und allen Modulen, die lediglich aus einem Fach bestehen, muss die Frist vermerkt sein, innerhalb derer die Prüfungsfragen vom Fachkoordinator geliefert werden müssen. Der Name des Fachkoordinators ist ebenfalls aufzuführen. Handelt es sich um ein Modul, das aus mehreren Fächern besteht, müssen die einzelnen Deadlines, der Name des Modulkoordinators und die Anzahl zu erstellender bzw. bereits erstellter Fragen angegeben sein.

(c) Der Prüfungskoordinator benötigt eine differenzierte Übersicht über sämtliche in den entsprechenden U/S/M/F-Kombinationen erstellten Fragen. Dem Prüfungskoordinator soll direkt einsichtig sein, welche Fragen autark bzw. nach Aufforderung erstellt wurden.

(d) Der Nutzer in der Rolle des Prüfungskoordinators muss in der Lage sein, 1-aus-5-MC-Prüfungsfragen zu erstellen. Es gibt zwei Methoden zur Erstellung von Fragen. Zum einen soll der Prüfungskoordinator in einem Online-Formular Prüfungsfragen erstellen. Zum anderen soll ein Import via Excel-Datei möglich sein. Eine Frage soll von einer Vaterfrage ableitbar sein, um die inhaltliche Verwandtschaft zu kennzeichnen. Eine Vaterfrage muss nicht zwingend der gleichen U/S/M/F-Kombination angehören. Als Vaterfrage sollen auch sämtliche Fragen zur Auswahl stehen, die in einem vorangegangenen Semester für die gleiche Universität/Modul/Fach-Zusammensetzung erstellt wurden. Der Prüfungskoordinator kann einer Frage einen Autor und

ein oder mehrere Lernziele zuordnen. Die Lernziele für die Medizinische Fakultät der WWU Münster sind aus dem Akzento-Informationssystem zu übernehmen. Hier werden die verschiedenen Versionen der Lernzielkataloge gepflegt. Im Allgemeinen ist eine Lernzielkatalogversion für genau ein Semester gültig. Existieren mehr als eine gültige Lernzielkatalogversion soll der Nutzer zwischen diesen wählen können. Für jede Frage kann ein eigener Fragestamm erstellt werden. Verschiedene Fragen sollen den gleichen Fragestamm besitzen können. Der Nutzer kann Fragestämme wählen, die der gleichen Universität/Modul/Fach-Kombination entstammen. Die Wahl ist nicht auf Fragestämme des aktuellen Semesters beschränkt. Entscheidet sich der Nutzer für die Erstellung eines eigenen Fragestammes, kann er diesem Fragestamm ein Medium zuweisen. Entweder lädt er ein eigenes Medium hoch oder er bedient sich eines bereits gespeicherten Mediums. Der Nutzer hat die Wahl zwischen sämtlichen Medien, die der gleichen Universität/Modul/Fach-Zusammensetzung angehören wie die zu erstellende Frage. Entscheidet sich der Nutzer zu einem eigens erstellten Fragestamm ein Medium hochzuladen, kann er dieses beschriften. Der Nutzer muss zu jeder 1-aus-5-MC-Frage einen Fragetext angeben. Einer Frage können bis zu fünf Bilder zugewiesen werden, die entweder vom Nutzer hochzuladen und zu beschriften sind oder aus der Datenbank gewählt werden können. Neben den Texten für die fünf Antwortmöglichkeiten muss der Prüfungskoordinator die richtige Lösung angeben. Optional soll er die Möglichkeit haben, einen erklärenden Text zur Lösung zu schreiben und diesen durch ein hochgeladenes oder aus der Datenbank gewähltes Medium zu illustrieren. Bei Fragestamm, Fragetext, Antwortmöglichkeiten, Begründung der Lösung und Beschriftungen der verschiedenen Medien soll der Nutzer in der Lage sein, eine Formatierung mittels HTML-Tags vorzunehmen und Sonderzeichen in den Text zu integrieren. Die Formatierungsoptionen sind auf die Attribute fett, kursiv, unterstrichen, durchgestrichen, höher- und tiefergestellt und Zeilenumbruch eingegrenzt. Als Sonderzeichen kann der Nutzer sämtliche in HTML verfügbaren Sonderzeichen verwenden. Zur Erleichterung der Eingabe erhält der Nutzer eine Kurzanleitung zur Verwendung der zulässigen Tags und Sonderzeichen. Dem Nutzer soll bei aktivierten JavaScript die Möglichkeit geboten werden, sich einer Buttonleiste zu bedienen, die nach

Markierung der entsprechenden Textpassagen die Tags für die Formatierung oder das gewünschte Sonderzeichen einfügt. Der Prüfungskoordinator soll seine durch HTML-Tags und Sonderzeichen erweiterten Texte betrachten und Veränderungen vornehmen können. Die hochgeladenen oder aus der Datenbank gewählten Bilder sollen im Eingabedialog als Thumbnails[4] dargestellt werden. Der Nutzer soll die Möglichkeit haben, das Bild in Originalgröße zu betrachten.

(e) Um eine bereits erstellte Prüfungsfrage modifizieren zu können, benötigt der Prüfungskoordinator ein Werkzeug, mit dessen Hilfe er die gewünschte Frage auswählen und sämtliche Attribute der Frage modifizieren kann. Es soll angezeigt werden, ob, wann und in welcher Modul/Fach-Zusammensetzung, die zu editierende Frage in einer Klausur Verwendung fand.

3.2.5 Funktionen für den Modulkoordinator

(a) Der Modulkoordinator soll für jedes der seinem Modul angehörenden Fächer einen Fachkoordinator selektieren können. Diesen kann der Modulkoordinator eine Deadline vorgeben, die der ihm zugewiesenen Deadline entspricht oder früher angesetzt ist. Auf diese Weise kann sich der Modulkoordinator einen zeitlichen Puffer schaffen, um Rücksprache mit den Fachkoordinatoren zu nehmen, die die ihnen zugewiesene Deadline überschritten haben und so für ein verspätetes, aber rechtzeitiges Einlesen der Fragen sorgen. Der Modulkoordinator besitzt außerdem die Möglichkeit, vom Fachkoordinator mehr Fragen zu verlangen als ihm selbst vom Prüfungskoordinator aufgetragen wurde, erstellen zu lassen. Dadurch können bei der Zusammenstellung der Klausur aus einer Auswahl von Fragen die geeignetsten selektiert werden.

(b) Der Modulkoordinator benötigt eine Übersicht, aus der hervorgeht, wie viele Fragen für jedes der seinem Modul inhärenten Fächer bereits erstellt wurden. Es muss weiterhin aufgeführt sein, wie viele Fragen vom Prüfungsko-

[4] Ein Thumbnail ist die verkleinerte Version eines Bildes, die oft als Vorschau für die große Version dient.

ordinator für jedes Fach verlangt wurden und wie viele Fragen der Modulkoordinator den einzelnen Fachkoordinatoren aufgetragen hat, zu erstellen. In dem Überblick muss festgehalten sein, welche Deadline vom Prüfungskoordinator vorgegeben wurde und welche Deadlines der Modulkoordinator für die einzelnen Fachkoordinatoren festgesetzt hat.

(c) Dem Modulkoordinator ist für jedes der seinem Modul inhärenten Fächer die Rolle des an ein Fach gebundenen Autors zuzuweisen. Somit stehen ihm zusätzlich sämtliche unter 3.2.8 beschriebenen Möglichkeiten zur Verfügung.

3.2.6 Funktionen für den Fachkoordinator

(a) Der Fachkoordinator muss beliebig viele Autoren auffordern können, Prüfungsfragen zu erstellen. Jedem Autor teilt der Fachkoordinator mit, wie viele Fragen er bis zu welchem Zeitpunkt zu welchem Themengebiet zu erstellen hat. Die einem Autor zugewiesene Deadline kann der dem Fachkoordinator vorgegebenen Deadline entsprechen oder davor liegen, um dem Fachkoordinator bei der Organisation der Prüfungsfragen einen zeitlichen Puffer einzuräumen. Die Größer des Puffers bemisst der Fachkoordinator anhand der Anzahl der Fragen und der Persönlichkeit des Autors.

(b) Der Fachkoordinator benötigt eine detaillierte Übersicht über die unter Aufsicht bereits erstellten Fragen. Aus dieser soll hervorgehen, wie viele Prüfungsfragen bis zu welchem Zeitpunkt der übergeordnete Koordinator angefordert hat. Es soll aufgeführt sein, wie viele Fragen für das Fach bereits erstellt wurden und wie viele Fragen der Fachkoordinator den einzelnen Autoren aufgetragen hat, zu erstellen. Neben dem Namen des Autors muss auch vermerkt sein, wie viele Fragen der Autor bisher erstellt hat und welche Deadline einem Autor vom Fachkoordinator vorgegeben wurde.

(c) Der Fachkoordinator soll in einer Übersicht sämtliche autark und nach Aufforderung erstellten Fragen einer U/S/M/F-Kombination betrachten können.

(d) Alle von einem Autor nach Aufforderung erstellten Fragen sollen vom Fach-

koordinator einzusehen sein.

(e) Der Fachkoordinator erhält für sein Fach die Rolle des an ein Fach gebundenen Autors.

3.2.7 Funktionen für den an eine konkrete Modul/Fach-Kombination gebundenen Autor

(a) Der Nutzer in dieser Autorrolle benötigt einen Überblick, um festzustellen, wie viele der verlangten Fragen er bereits erstellt hat. Insbesondere muss die Deadline aufgeführt sein, bis zu der die Fragen erstellt sein müssen.

(b) Der Autor benötigt einer Übersicht über alle von ihm erstellten Fragen.

(c) Der Nutzer, der diese Form der Autor-Rolle inne hat, soll mindestens die Anzahl Prüfungsfragen in der ihm zugewiesenen U/S/M/F-Kombination erstellen, die ihm vom Fachkoordinator aufgetragen wurde. Es gelten für diesen Dialog die gleichen Anforderungen wie für den Fragen-Erstellen-Dialog des Prüfungskoordinators (vgl. Abschnitt 3.2.4). Im Gegensatz zum Erstellungsdialog des Prüfungskoordinators müssen in diesem Dialog die Lernziele angegeben werden. Nach dem Speichern einer Prüfungsfrage müssen die Statusübersichten für die beteiligten Koordinatoren aktualisiert werden.

(d) Der Autor soll nur Prüfungsfragen aus dem aktuellen oder aus zukünftigen Semestern editieren können. So wird verhindert, dass bereits in Klausuren verwendete Fragen nachträglich modifiziert werden. Bereits einer Klausur zugeordnete Fragen dürfen von den Autoren nicht mehr bearbeitet werden. Sie können sich an einen Prüfungskoordinator wenden, um Veränderungen an alten Prüfungsfragen vornehmen zu lassen. Alternativ können sie die Frage als Vaterfrage verwenden und an der Kindfrage die gewünschten Modifikationen vornehmen.

(e) Der Autor erhält die Autorenrechte für das innerhalb seiner U/S/M/F-Kombination verborgene Fach.

3.2.8 Funktionen für den an ein Fach gebundenen Autor

(a) Ein Nutzer mit dieser Form der Autor-Rolle ist jederzeit in der Lage, Prüfungsfragen zu erstellen. Die Anforderungen an das Erstellen einer Prüfungsfrage entsprechen den Anforderungen aus Abschnitt 3.2.7. Darüber hinaus kann der Autor nicht nur in der ihm zugewiesenen U/S/M/F-Kombination Prüfungsfragen erstellen, sondern in jeder Modul/Fach-Kombination des aktuellen oder zukünftiger Semesters seiner Universität, in der das ihm zugewiesene Fach enthalten ist. Diese Rolle ergab sich aus der Anforderung, dass sämtliche am System beteiligte Personen jederzeit Prüfungsfragen erstellen können. Ein Modulkoordinator bekommt die Autorenrechte für jedes der in seinem Modul vertretenen Fächer, ein Fachkoordinator für sein Fach und ein Autor für das seiner U/S/M/F-Kombination angehörende Fach. Da an den Fragen-Erstellungsdialog des Prüfungskoordinators erweiterte Anforderungen, wie etwa das Zuweisen eines Autors zu einer Frage geknüpft sind, reicht es nicht aus, einem Prüfungskoordinator die Autor-Rechte für sämtliche Fächer zu verleihen. Im Gegensatz zu allen bisher beschriebenen Rollen, mit Ausnahme der Rolle des Prüfungskoordinators, behält ein Nutzer die hier geschilderten Autorenrechte über Semestergrenzen hinweg. Alle anderen Rechte verfallen nach Ablauf eines Semesters.

(b) Der an ein Fach gebundene Autor muss in die Lage versetzt werden, von ihm erstellte Fragen zu editieren. Hier gelten die gleichen Beschränkungen wie in Abschnitt 3.2.7 d beschrieben.

3.2.9 Weitere Funktionalitäten und Anforderungen

(a) Der Sicherheit des Systems ist höchste Priorität einzuräumen.

(b) Es ist großer Wert auf eine intuitive und übersichtliche Bedienung des Programms zu legen.

(c) Das gesamte System muss ohne JavaScript lauffähig sein. JavaScript darf nur optional zum Einsatz kommen, um Arbeitserleichterungen zu ermöglichen.

(d) Die Benutzerauthentifizierung und der Zugriff auf die Stammdaten soll mittels des CADS-Systems (vgl. Abschnitt 2.1.9) erfolgen.

(e) Nach Vorgabe des Behindertengleichstellungsgesetzes vom 16. Dezember 2003 ([38]) muss das EPM-System den Regeln der Barrierefreiheit genügen.

(f) Sind bis zum Ablauf einer Deadline noch nicht genügend Prüfungsfragen erstellt, so sind die involvierten Nutzer automatisiert per Mail zu informieren.

(g) Jeder Nutzer soll auf einer Statusseite eine Übersicht über die ihm zugewiesenen Rollen erhalten. Es ist möglich, dass ein Nutzer mehrfacher Modulkoordinator und Fachkoordinator ist und diverse Autor-Rollen inne hat.

(h) Zu jeder relevanten Seite soll eine Hilfefunktion existieren, die Antworten auf möglicherweise auftretende technische oder inhaltliche Fragen gibt.

(i) Es soll eine Logging-Funktionalität implementiert werden, die wichtige Datenbankoperationen protokolliert. Aus dem Protokoll soll hervorgehen, wer wann welche relevante Datenbankoperation ausgeführt hat. Zusätzlich sollen aufgetretene Fehler in einem Logfile gespeichert werden.

3.3 Teilbereich III: Reviewsystem

Ziel des Reviewsystems ist eine Verbesserung der Qualität von Prüfungsfragen. Durch die Beteiligung verschiedener Universitäten soll ein NRW-weiter Pool hochqualitativer Prüfungsfragen aufgebaut werden.

Ein Nutzer in der Rolle des Prüfungskoordinators soll in der Lage sein, einen Reviewprozess für im System gespeicherte Prüfungsfragen anzustoßen.

Es existieren drei verschiedene Reviewverfahren (vgl. Abbildung 3.2). Beim internen, fachspezifischen Reviewverfahren wird die zu begutachtende Prüfungsfrage verschiedenen Dozenten der Fachdisziplin vorgelegt, der die zu begutachtende Frage zugeordnet ist. Beim internen, interdisziplinären Reviewverfahren wird die Frage einer handverlesenen Auswahl von Dozenten verschiedener Fachdiszipli-

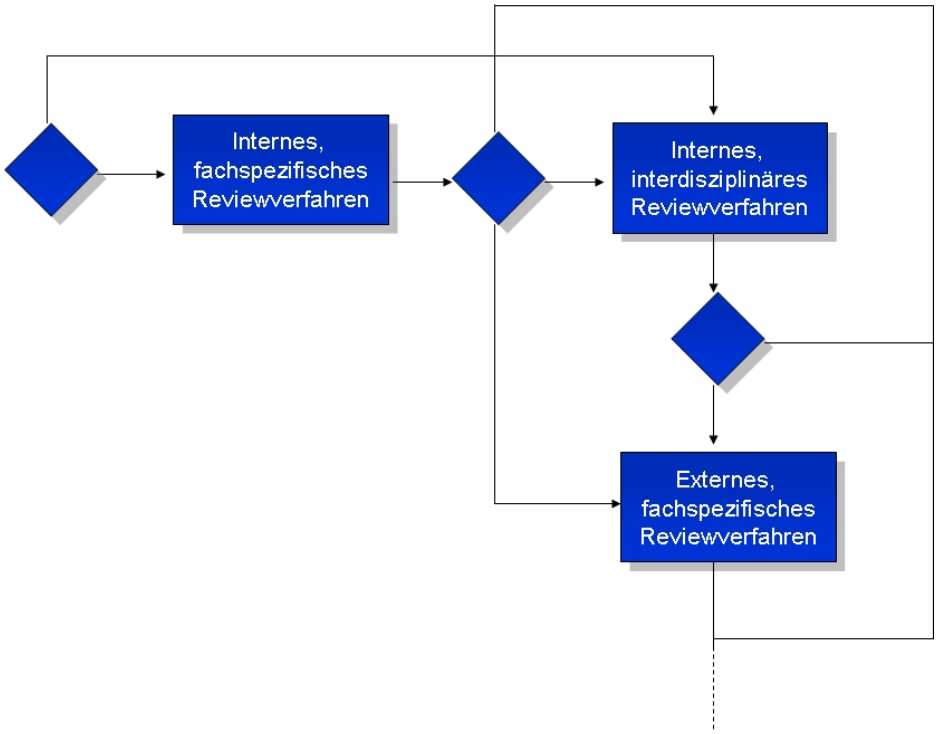


Abbildung 3.2: Review-Workflow

nen zur Begutachtung übergeben. Durch die interdisziplinäre Zusammensetzung des Reviewkomitees soll vermieden werden, dass bei der Bewertung einer Prüfungsfrage medizinisches Wissen als bekannt vorausgesetzt wird, das außerhalb einer Fachdisziplin wenig bekannt ist. Das externe, fachspezifische Reviewverfahren setzt auf eine Kooperation verschiedener Medizinischer Fakultäten innerhalb Nordrhein-Westfalens. Die Frage wird von Dozenten verschiedener Universitäten begutachtet. Alle Dozenten gehören der Fachdisziplin an, die der zu beurteilenden Frage zugewiesen wurde.

Der Prüfungskoordinator kann sich für ein singuläres, internes, fachspezifisches oder ein internes, interdisziplinäres Reviewverfahren entscheiden. Ihm steht auch die Möglichkeit zur Vefügung, diese beiden Verfahren miteinander zu kombinieren, wobei zunächst das fachspezifische und dann das interdisziplinäre Reviewverfahren durchgeführt wird. Wurde ein internes Reviewverfahren nach einer der drei genannten Kombinationen durchgeführt, kann sich der Prüfungskoordinator entscheiden, ob er die Prüfungsfragen zusätzlich dem externen Reviewverfahren

unterziehen will, um so ein Maximum an Qualität zu erzielen.

Die Fragen sind bei der Begutachtung nach den Kriterien inhaltliche Relevanz, orthographische und grammatikalische Qualität und didaktische Qualität zu beurteilen. Die Beurteilung nach den Kennzeichen inhaltiche Relevanz und grammatikalische Qualität erfolgt mittels einer Skalierung.

Abhängig vom Ergebnis einer Begutachtung wird eine Frage entweder

- direkt in den Fragenpool übernommen,
- nach kleineren Korrekturen in den Pool eingespeist oder
- sie durchläuft nach ausführlicher Überarbeitung erneut den Reviewprozess.

Ein wichtiges Kriterium beim Entwurf des Review-Workflows besteht in dem Prinzip der Anonymität. Ein Autor kennt den Begutachter nicht. Diesem ist wiederum nicht bekannt, wer die vorgelegte Frage erstellt hat. Die Auswahl der externen Begutachter soll randomisiert erfolgen.

3.4 Technologieauswahl

Es musste die Entscheidung bezüglich des beim Prüfungsmanagement zu verwendenden Datenbanksystems (vgl. Abschnitt 2.1.2) gefällt werden. Zur Diskussion standen die Datenbanksysteme Oracle, PostgreSQL und MySQL. Für Oracle sprachen Stabilität, Leistungsstärke und Mächtigkeit. Gegen Oracle sprachen die Kosten für Lizenz[5] und Administration (vgl. [11]) und die Tatsache, dass in der zuständigen Abteilung keinerlei Erfahrungen bezüglich Administration und Nutzung von Oracle-Datenbanken bestanden. Stärken von PostgreSQL liegen in Robustheit, Leistungsumfang und freier Zugänglichkeit aufgrund von Open-Source-Lizenzen. Gegenüber MySQL ist ein einfacherer Umstieg auf Oracle möglich. Obgleich der Leistungsumfang im Vergleich zu MySQL größer ist, ist er geringer als der des Konkurrenten Oracle. Der lesende Zugriff ist gegenüber MySQL

[5] Die mit Einschränkungen freie (vgl. [60]) Oracle-Version mit Namen Oracle Database 10g Express Edition (Oracle Database XE) wurde erst im März 2006 von Oracle freigegeben (vgl. [75]).

langsamer. Für Mysql sprechen neben freier Zugänglichkeit, Stabilität und Leistungsstärke vor allem der in der Abteilung bestehende Erfahrungsschatz bezüglich Administration und Nutzung von MySQL-Datenbanksystemen und die gute Dokumentation (vgl. [29]). Der im Vergleich zu den Kontrahenten geringere Funktionsumfang ist ein Nachteil dieses Datenbanksystems. Eine genaue Analyse bezüglich des zu erwartenden Datenaufkommens und bezüglich der Anforderungen an den Funktionsumfang ergab, dass alle drei Kontrahenten den Anforderungen gewachsen sein würden. Die Entscheidung bezüglich des einzusetzenden Datenbanksystems fiel auf MySQL. Hauptgründe waren das Lizenzmodell und die in der Abteilung vorhandenen Erfahrungen bezüglich dieses Datenbanksystems.

Aus lizenztechnischen und institutsinternen Gründen sollte das EPM-System mit Hilfe des vollen Funktionsumfang der J2EE-Spezifikation entwickelt werden. Hierbei sollten als Framework JavaServer Faces oder Struts und zur Realisierung der Persistenzschicht Entitiy- und Session-Beans zum Einsatz kommen (vgl. Abschnitt 2.1.4). Da die JavaServer Faces-Technologie zum damaligen Zeitpunkt lediglich in der ersten Version (Version 1.1) zur Verfügung standen, war eine geringe Stabilität zu befürchten. Diese Bedenken wurden durch Erfahrungen in einem anderen Projekt untermauert. Daher wurde von dem Einsatz der JSF oder einer Mischform aus JSF und Struts Abstand genommen und der Verwendung von Struts als Framework der Vorzug gegeben. Insbesondere sprach gegen den symbiotischen Gebrauch von Struts und JSF der gegenüber einem einzelnen Framework erhöhte Einarbeitungsaufwand.

Zur Evaluierung der Praxistauglichkeit bezüglich der Kombination von Struts und EJBs wurde ein Programm zur Prüfungsanmeldung unter Anwendung dieser Technologien umgesetzt. Die Implementierung dieser Anwendung zeigte, dass der Einsatz von EJBs die Entwicklungszeit unnötig erhöhte. Um kurzfristig auf Änderungen der Anforderungen reagieren zu können, ist die Verwendung von EJBs gegenüber dem direkten Datenbankzugriff zu unflexibel. Insbesondere erwies sich die Einarbeitung in die EJB-Technologie als extrem aufwendig und schwierig. Bei der weiteren Entwicklung wurde auf den Einsatz von EJBs verzichtet. Der Datenbankzugriff erfolgt direkt mittels JDBC (vgl. Abschnitt 2.1.4), ohne von einer zusätzichen Persistenzschicht Gebrauch zu machen. Zur Steigerung der Lesbarkeit wurden JSTL-Tags, die Expression Language und Struts EL eingesetzt. Die

Tiles-Technologie ermöglicht einen einfachen Umstieg auf ein neues Layout und erleichtert es, die Anforderung der Barrierefreiheit umzusetzen. Sofern möglich wurde die in Struts ingegrierte vereinfachte Validierung der Nutzereingaben genutzt.

4 Ergebnisse

Dieses Kapitel beschreibt die Realisierung des Prüfungsmanagements (vgl. Abschnitt 3.2), das den Schwerpunkt dieser Dissertation darstellt.

4.1 Universitätszugehörigkeit

Ein Nutzer kann mehrere Rollen innerhalb verschiedener Universitäten inne haben. In diesem Fall muss sich der Nutzer beim Start des Programms für eine Universität entscheiden. Dies verhindert, dass dieser bei jedem Aufruf eines Menüpunktes zunächst eine Universität wählen muss.

4.2 Layout

Das Layout des EPM-Systems wurde unter Nutzung von Tiles (vgl. Abschnitt 2.1.6) realisiert. Der prinzipielle Aufbau entspricht der Struktur in Abbildung 2.5, wobei der Menübereich aus einem Hauptmenü und bei Bedarf aus einem zusätzlichen Untermenü besteht. Jeder Menüpunkt ist mit einem eigenen Icon versehen, das sich nach der Auswahl eines Menüpunktes auch in der Überschriftzeile des Content-Bereichs wiederfindet (vgl. z. B. Abbildung 4.3). In der Fußzeile einer jeden Seite findet sich eine Mailadresse, über die der Nutzer bei Bedarf den direkten Kontakt zum EPM-Supportteam herstellen kann. Die wichtigsten Punkte des Hauptmenüs stellen die Einträge Statusübersicht und Fragengeneration dar, wobei das Anklicken des letzteren zum Öffnen eines Submenüs führt. Zusätzlich erhält der Nutzer im Contentbereich eine genaue Beschreibung der für

ihn zugänglichen Menüpunkte. Alle Seiten sind unter Einhaltung der Regeln zur Barrierefreiheit umgesetzt (vgl. Anforderung 3.2.9 e). Die einzelnen Menüeinträge werden im Folgenden genauer beschrieben.

4.3 Statusübersicht

Um der Anforderung nach einer möglichst guten Übersichtlichkeit und Bedienbarkeit des Programms gerecht zu werden, (vgl. 3.2.9 b) wurden die Anforderungen 3.2.4 b, 3.2.4 c, 3.2.5 b, 3.2.6 b, 3.2.6 c, 3.2.6 d, 3.2.7 a, 3.2.7 b und 3.2.9 g unter dem Hauptmenüpunkt Statusübersicht umgesetzt. Hier erfährt der Nutzer, welche Rollen er besitzt und wie es um die Erstellung der Prüfungsfragen bestellt ist. Meldet sich ein Nutzer im System an, wird er initial auf die erste Seite dieses Wizards[1] geleitet, um auf einen Blick über den Fortschritt der Fragenerstellung unterrichtet zu sein und sich bei Bedarf über die Details zu informieren.

Der Prüfungskoordinator sieht unter der Überschrift „Prüfungskoordinatorkonto: Fächer und Module, die nur aus einem Fach bestehen“ eine Tabelle, welche den Modulnamen, den Fachnamen, das Semester, den gewählten Fachkoordinator und die Deadline enthält (vgl. Abbildug 4.1 links). Die Gesamtzahl der erstellten und für diese U/S/M/F-Kombination zu erstellenden Fragen ist in einer Legende unter der Bezeichnung „Fragenratio“ abgebildet. Die Anzahl der bereits erstellten Fragen wird durch einen blauen Punkt repräsentiert, während die Menge der noch zu erstellenden Prüfungsfragen durch einen roten Punkt vertreten ist. Da im System die Möglichkeit besteht, mehr Prüfungsfragen als vom Prüfungskoordinator gefordert erstellen zu lassen, werden die „zu viel erstellten Fragen“ durch einen grünen Punkt angezeigt.

Über einen Link erhält der Prüfungskoordinator Zugriff auf eine Übersicht über sämtliche für die gewählte U/S/M/F-Kombination erstellten Prüfungsfragen. Hier finden sich sowohl die autark als auch die nach Aufforderung erstellten Fragen.

[1] Der Begriff Wizard oder Assistent bezeichnet in der Datenverarbeitung eine Aneinanderreihung von Dialogen, die der ergonomischen Eingabe von Daten dient. Die Wahl des nächsten Dialogs erfolgt abhängig von der Auswahl des Nutzers und dem Zustands des zugrunde liegenden Datenmodels. Die verschiedenen Abfolgen lassen sich als Wege innerhalb eines gerichteten Graphen illustrieren.

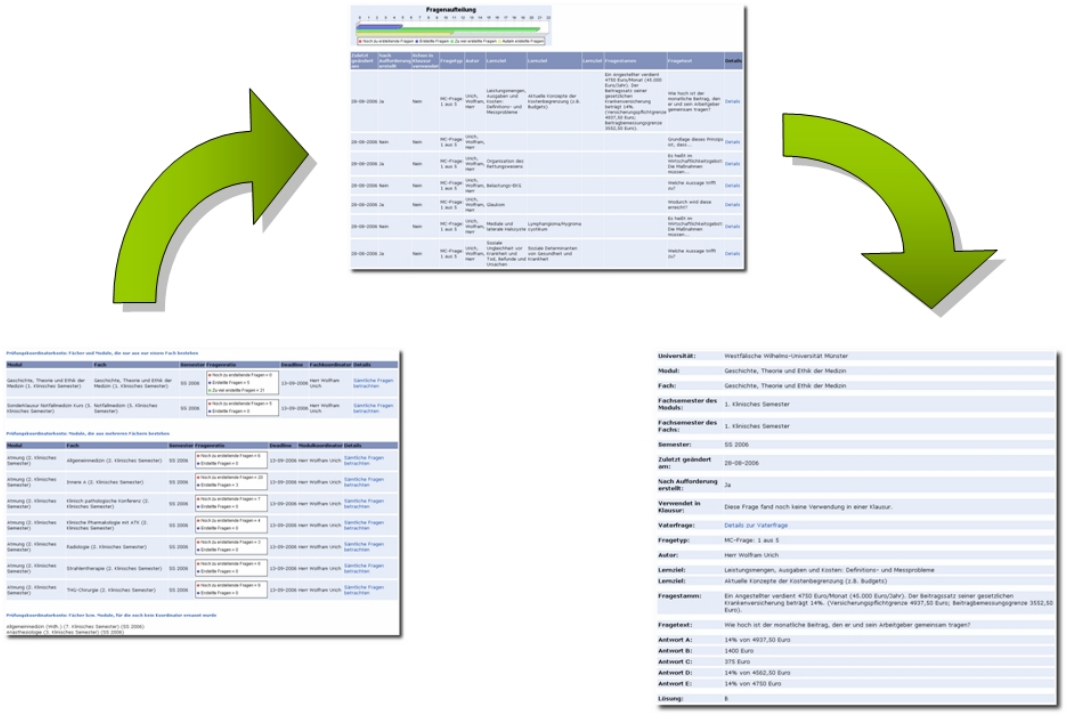


Abbildung 4.1: Status- und Fragenübersicht für den Prüfungskoordinator

An dieser Stelle kann der Nutzer die Fragen nach mehreren Kriterien gleichzeitig ordnen und über das Anklicken eines Hyperlinks sämtliche Attribute einer Frage betrachten (vgl. Abbildug 4.1 mittig und rechts).

Unter der Kopfzeile „Prüfungskoordinatorkonto: Module, die aus mehreren Fächern bestehen“ finden sich die gleichen Angaben wie unter der gerade beschriebenen Überschrift. Auch hier hat der Nutzer die Möglichkeit, die Prüfungsfragen im Detail zu betrachten. Anstelle des Fachkoordinators ist der Name des Modulkoordinators aufgeführt.

Unterhalb der Überschrift „Prüfungskoordinatorkonto: Fächer bzw. Module, für die noch kein Koordinator ernannt wurde“ findet der Nutzer eine Übersicht über sämtliche Module bzw. Fächer des momentanen bzw. zukünftiger Semester, für die noch kein Modul- bzw. Fachkoordinator gewählt wurde.

Der Nutzer in der Rolle eines Modulkoordinators erhält unter der Überschrift „Modulkoordinatorkonto“ eine Übersicht über alle wichtigen Details bezüglich

Abbildung 4.2: Statusübersicht für den Modulkoordinator

der Prüfungsfragenerstellung. In Form einer Tabelle bezieht der Nutzer eine Aufstellung aller für ihn relevanten U/S/M/F-Kompositionen (vgl. Abbildug 4.2). Hier finden sich neben der Semesterangabe, die für jedes Fach vom Prüfungskoordinator und vom Modulkoordinator selbst vorgegebenen Deadlines, inklusive zweier Kuchendiagramme, welche die Fragenratios bezüglich der Vorgaben des Prüfungskoordinators und bezüglich der Richtschnur des Modulkoordinators repräsentieren. Der Name des Fachkoordinators ist aufgeführt, sofern der Modulkoordinator ihn bereits ernannt hat. Die Kuchendiagramme wurden unter Nutzung des Cewolf-Projektes ([70]) implementiert. Dieses erlaubt eine dynamische, serverseitige Generierung von Diagrammen, die an den Browser als Bilddatei geliefert werden.

Ein Fachkoordinator findet unter der Überschrift „Fachkoordinatorkonto“ eine Auflistung sämtlicher für ihn wichtigen U/S/M/F-Kombinationen. Modulname, Fachname, Semester und Deadline werden ergänzt durch die Fragenratio bezüglich des übergeordneten Koordinators in Form eines Kuchendiagramms. Durch

Anklicken eines Links gelangt der Fachkoordinator zu einer Ansicht, in der im oberen Bereich eine wiederholte Darstellung der entsprechenden Zeile der Tabelle der vorhergehenden Seite erscheint. Darunter befindet sich eine Tabelle, die die Aktivitäten der einzelnen Autoren detailliert wiedergibt. Neben Modul-, Fach- und Semesterangaben enthält diese Tabelle den Namen des Autors, die Deadline und die Fragenratio hinsichtlich der Vorgaben des Fachkoordinators als Tortendiagramm. Diese Seite bietet dem Fachkoordinator die Möglichkeit, sämtliche autark und nach Aufforderung erstellten Fragen für die gewählte U/S/M/F-Zusammenstellung zu betrachten. Alternativ kann er alle von einem einzelnen Autor nach Aufforderung generierten Fragen einsehen.

Ein an eine konkrete Modul-Fach-Kombination gebundener Autor findet unter dem Punkt „Autorkonto“ ein Auflistung von Modulname, Fachname, Semester und Deadline. Die sich auf die Vorgaben des Fachkoordinators beziehende Fragenratio ist in Form eines Kreisdiagramms dargestellt. Über einen Link gelangt der Autor zu einer Übersicht über sämtliche von ihm erstellten Fragen der gewählten U/S/M/F-Kombination.

Unter der Überschrift „Rollenübersicht“ findet der Nutzer eine Aufzählung seiner Rollen.

4.4 Ernennung von Modul- und Fachkoordinatoren

Der Wizard für die Ernennung von Modul- und Fachkoordinatoren (vgl. Anforderung 3.2.4 a) kann von Nutzern in der Rolle des Prüfungskoordinators über den Menüeintrag „Koordinator ernennen“ innerhalb des „Fragengeneration“-Submenüs gestartet werden. Das Programm überprüft zunächst, ob es mehr als ein Semester gibt, das noch nicht abgelaufen ist. Dieser Fall tritt z. B. ein, wenn ein Semester noch gültig, aber das nächste Semester samt zugehöriger Modul/Fach-Kombinationen bereits in der Datenbank angelegt ist. Existiert nur ein gültiges Semester, wird direkt zum nächsten Dialog gesprungen, gibt es mehr als eines, kann der Nutzer ein Semester selektieren. Im nächsten Dialog kann der Prü-

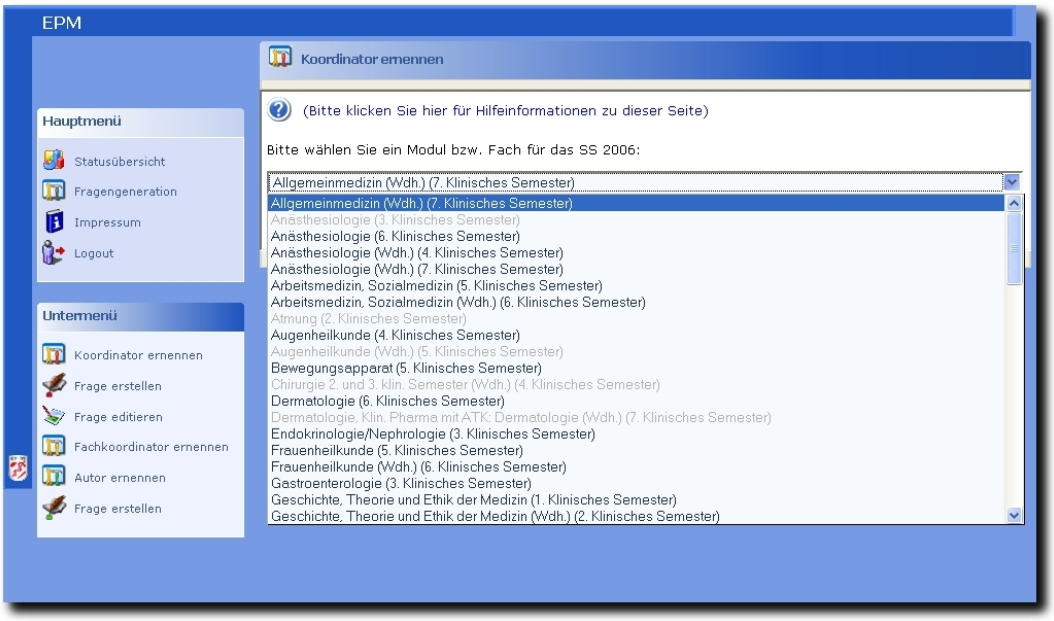


Abbildung 4.3: Auswahl eines Fachs bzw. Moduls.

fungskoordinator sich für ein Modul oder Fach des davor selektierten Semester entscheiden. Die Einträge, für die bereits ein Modul- bzw. Fachkoordinator ernannt wurde, sind ausgegraut dargestellt (vgl. Abbildug 4.3). Entscheidet sich der Nutzer dennoch für ein solches Modul bzw. Fach, erscheint eine entsprechende Fehlermeldung. Abhängig davon, ob der Nutzer ein Fach oder ein Modul gewählt hat, erfolgt der Fortgang innerhalb des Eingabeassistenten:

- Hat sich der Prüfungskoordinator für eine Fach entschieden, kann er im nachfolgenden Dialog einen Fachkoordinator wählen. Um Nutzer mit gleichem Vor- und Zunamen unterscheiden zu können, ist neben der Anrede und dem Titel das Institut aufgeführt. Der Prüfungskoordinator legt Deadline und Anzahl der zu erstellenden Fragen fest. Klickt der Nutzer bei der Eingabe auf ein Kalendersymbol, das sich neben dem Eingabefeld für die Deadline befindet, öffnet sich ein weiteres Fenster. Dieses beinhaltet einen Kalender mit dessen Hilfe der Nutzer ein Datum für die Deadline wählt. Bei deaktiviertem JavaScript (vgl. Anforderung 3.2.9 c) muss der Prüfungskoordinator die Deadline per Hand eingeben. Das Programm überprüft, ob die Angabe für das Datum im gültigen Format vorliegt und ob der gewählte

Termin nicht vor dem aktuellen Tag liegt. Für den Wert der Fragenanzahl wird überprüft, ob eine positive Ganzzahl angegeben wurde. Der nächste Dialog beinhaltet ein Mail-Formular. Der Prüfungskoordinator legt hier die Betreff-Zeile und den Text der E-Mail fest, die dem potentiellen Fachkoordinator nach dem Drücken des Abschicken-Buttons gemailt wird. Die Absenderadresse ist die Mailadresse des Prüfungskoordinators, die Empfängeradresse ist die des zukünftigen Fachkoordinators. Sämtliche im System für die entsprechende Universität registrierten Prüfungskoordinatoren erhalten eine Kopie der Mail. Dies dient der Erleichterung der Abstimmung und dem gezielten Informationsfluss zwischen den Prüfungskoordinatoren. Sowohl Betreff als auch Text der Mail sind standardmäßig vorbelegt. Sie können jedoch beliebig modifiziert werden. Im folgenden Dialog wird der Nutzer informiert, ob das Versenden der Mail erfolgreich verlaufen ist. Konnte eine Mail nicht ordnungsgemäß zugestellt werden, geht die Mail zurück an die Adresse des Prüfungskoordinators. Alle wichtigen Informationen werden in der Datenbank gespeichert. Es wurden Sicherheitsmaßnahmen getroffen, um einer sogenannten SQL-Injeciton vorzubeugen. Bei einer SQL-Injection fügt der Nutzer einer SQL-Anfrage zusätzliche SQL-Befehle hinzu. Auf diese Weise können sensible Information erlangt oder ganze Datenbankinhalte gelöscht werden (vgl. [27]). Der Fachkoordinator erhält neben der Rolle des Fachkoordinators die Autorrolle für das entsprechende Fach.

- Fiel die Entscheidung des Prüfungskoordinators auf ein Modul, so kann er im nachfolgenden Dialog den Modulkoordinator ernennen. Für jedes dem Modul inhärente Fach werden Deadline und Fragenanzahl festgelegt. Bei der Angabe der Deadline kann der Nutzer auf einen Popup-Kalender zurückgreifen. Nach dem Drücken des Abschicken-Buttons und dem Prüfen der Daten gelangt der Nutzer zu einem Mail-Dialog, in welchem er einen vorgefertigten Betreff-Text und einen vorbelegten Mailtext findet. Dem Nutzer steht es frei, diese zu modifizieren. Beim Abschicken der Mail erhalten sämtliche Prüfungskoordinatoren eine Kopie. In der Datenbank werden alle relevanten Daten gespeichert. Insbesondere werden dem gewählten Modulkoordinator sämtliche Autorenrechte für die innerhalb seines Moduls existenten Fächer zugewiesen (Anforerung 3.2.5 c).

4.5 Ernennung eines Fachkoordinators

Ein Nutzer in der Rolle eines Modulkoordinators startet durch den Menüpunkt „Fachkoordinator ernennen“ innerhalb des „Fragengeneration“-Untermenüs einen Assistenten, der ihm bei der Auswahl und Ernennung eines geeigneten Fachkoordinators behilflich ist (Implementierung der Anforderung 3.2.5 a). Zunächst überprüft das Programm, ob der Nutzer Modulkoordinator für Module aus einem oder mehreren gültigen Semestern ist. Besitzt der Nutzer Modulkoordinatorrechte für Module aus unterschiedlichen Semestern, wird er zunächst aufgefordert, ein Semester auszuwählen. Ist er Modulkoordinator für ein oder mehrere Module aus nur einem gültigen Semester, wird dieses automatisch ausgewählt und der Nutzer gelangt geradewegs zum zweiten Dialog. Nun wird überprüft, ob der Nutzer Modulkoordinator für ein oder mehrere Module des vorher selektierten Semesters ist. Ist er Modulkoordinator für nur ein Modul, wird der Nutzer direkt zum nächsten Dialog weitergeleitet. Im anderen Fall muss er sich für ein Modul entscheiden. Im nun folgenden Abschnitt des Wizards entscheidet sich der Nutzer für ein dem Modul zugeordnetes Fach. Auf dieser Seite wird der Name des automatisch oder von ihm gewählten Moduls angezeigt. Neben dem Fachnamen ist auch das Fachsemester aufgeführt, das einem Fach zugewiesen ist. So kann der Nutzer bei Wiederholungsmodulen die einzelnen Einträge unterscheiden. Ist der Nutzer beispielsweise Modulkoordinator des Moduls „Allgemeinmedizin (Wdh.)“ kann er vier verschiedene Fachkoordinatoren für das Fach Allgemeinmedizin bestimmen, da dieses dem 2., 3., 5. und 6. klinischem Semester zugeordnet ist (vgl. Abschnitt 2.2.2 unten). Hat der Modulkoordinator für das gewählte Fach bereits einen Fachkoordinator bestimmt, ist der entsprechende Eintrag ausgegraut. Selektiert der Nutzer diesen Eintrag dennoch und drückt anschließend auf den Abschicken-Button, kommt es zu einer Fehlermeldung. Im nächsten Dialog wählt der Modulkoordinator den Fachkoordinator und legt die für ihn gültige Deadline und die Anzahl der zu erstellenden Fragen fest (vgl. Abbildug 4.4). Standardmäßig sind hier die vom Prüfungskoordinator festgelegten Werte vorgegeben. Der Modulkoordinator erhält vom Programm den durch eine gelbe Glühbirne signalisierten Ratschlag, für den Fachkoordinator eine frühere Deadline als die ihm selbst vom Prüfungskoordinator vorgegebene Frist zu wählen. So kann er sich einen zeitlichen Puffer verschaffen. Oben auf der Seite findet der Nutzer die blau abgesetzten

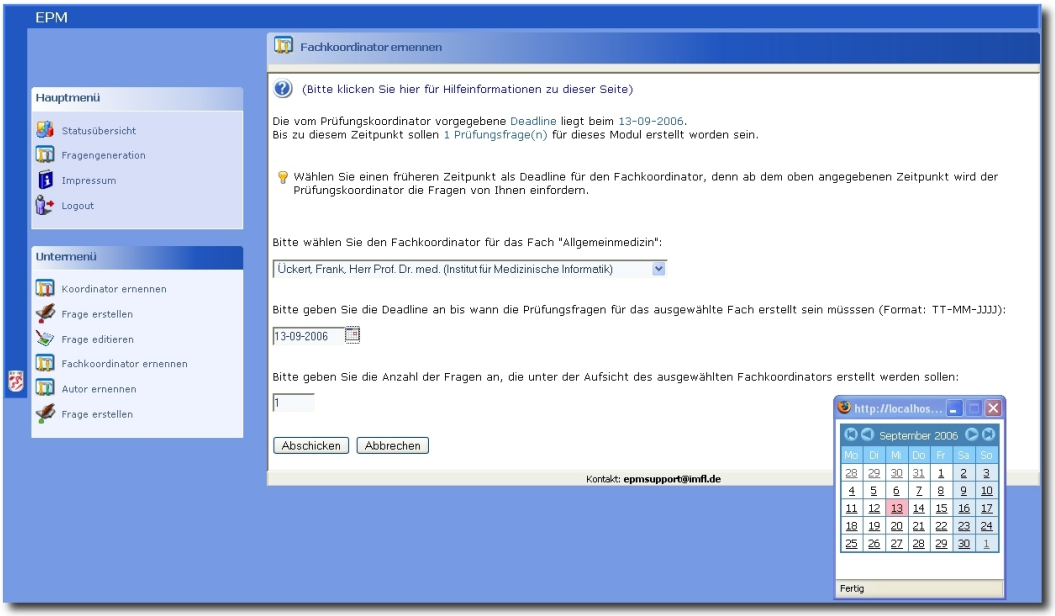


Abbildung 4.4: Auswahl eines Fachkoordinators.

Angaben bezüglich der vom Prüfungskoordinator vorgegebenen Deadline und der von ihm geforderten Anzahl der zu erstellenden Fragen. Dem Modulkoordinator steht es frei, beim Fachkoordinator mehr Fragen als von ihm selbst gefordert zu verlangen. Das System verhindert die Bitte nach einer zu geringen Anzahl von Prüfungsfragen. Es wird überprüft, ob es sich bei der Fragenanzahl um einen positiven Ganzzahlwert handelt und ob die Deadline nicht vor dem momentanen Datum liegt. Im nachfolgenden Dialog kann der Modulkoordinator den vom System vorgegebenen Einladungstext an den potentiellen Fachkoordinator und die Betreffzeile modifizieren. Der Modulkoordinator erhält eine Kopie der Mail. Nach dem erfolgreichen Verschicken der Mail wird der Modulkoordinator informiert und die Datenbankeinträge werden vorgenommen. Der Fachkoordinator erhält die Autorrechte für sein Fach (Anforderung 3.2.6 e).

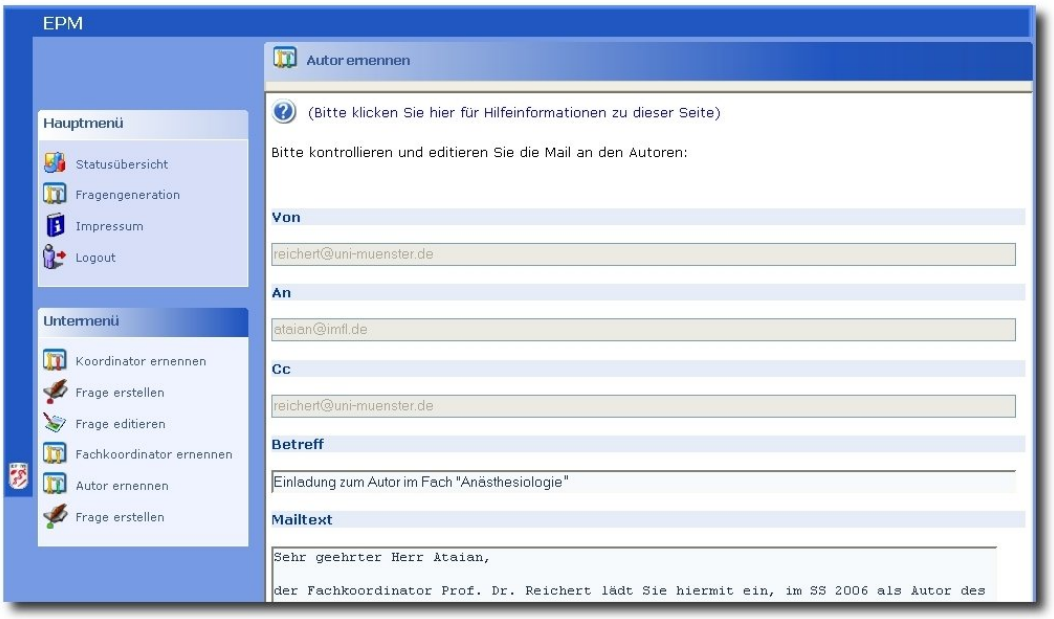


Abbildung 4.5: Mail-Dialog bei der Ernennung eines Autors.

4.6 Ernennung eines Autors

Ein Nutzer in der Rolle eines Fachkoordinators startet mittels des Menüeintrages „Autor ernennen“ innerhalb des „Fragengeneration“-Submenüs den Assistenten zum Selektieren eines an eine konkrete U/S/M/F-Komposition gebundenen Autors (Anforderung 3.2.6 a). Zunächst wird überprüft, ob der Nutzer Fachkoordinator für Fächer ist, die mehreren gültigen Semestern zugeordnet sind. Ist dies der Fall, gelangt der Autor zu einem Dialog, in dem er das entsprechende Semester wählt, ansonsten wird das einzige zur Verfügung stehende Semester selektiert und der Nutzer wird automatisch zur nächsten Seite weitergeleitet. Ist der Nutzer in dem gesetzten Semester für mehr als ein Fach als Fachkoordinator registriert, gelangt er nun zu einem Eingabedialog, in welchem er sich für ein Fach entscheiden muss. Existiert nur ein Fach, wird er direkt weitergeleitet. Im oberen Feld des Folgedialogs findet der Fachkoordinator eine Aufschlüsselung sämtlicher für ihn relevanten Daten. Hier ist die vom Prüfungs- oder Modulkoordinator vorgegebene Deadline vermerkt. Neben der von ihm verlangten Anzahl der zu erstellenden Prüfungsfragen ist aufgeführt, wie viele Prüfungsfragen der Fachkoordinator bis-

her von seinen Autoren angefordert hat. Unter diesen Angaben findet sich ein mit einem gelben Glühbirnen-Symbol versehener Text, der den Fachkoordinator darauf hinweist, die dem Autor zuzuweisende Frist nach Möglichkeit vor die dem Fachkoordinator selbst zugewiesene Deadline zu setzen. In den Eingabefeldern kann der Fachkoordinator aus einer Liste den Autor wählen und die Deadline (eventuell mittels Popup-Kalenders) und die Anzahl der vom Autor zu erstellenden Prüfungsfragen festlegen. Beim Abschicken des Formulars wird überprüft, ob die Daten korrekt angegeben wurden und ob insbesondere das gewählte Datum nicht vor dem aktuellen Datum und nach der vom übergeordneten Koordinator festgesetzten Deadline liegt. Im nächsten Teil des Assistenten kann der Nutzer Mailtext und Betreff der an den Autor zu schickenden Mail editieren, um diesem beispielsweise aufzutragen, eine Frage zu einem bestimmten Thema zu realisieren (vgl. Abbildung 4.5). Nach dem Abschicken erhält der zukünftige Autor die Mail, der Fachkoordinator eine Kopie. In der Datenbank werden für den vom Fachkoordinator gewählten Nutzer nicht nur die Autorenrechte für eine konkrete Modul/Fach-Kombination zugewiesen, er erhält zusätzlich die Rolle eines Autors für das konkrete Fach (Implementierung von Anforderung 3.2.7 e).

4.7 Fragenerstellung für den Prüfungskoordinator

Unter dem Menüpunkt „Frage erstellen“ im „Fragengeneration“-Untermenü kann der Prüfungskoordinator eine Frage erstellen und im System abspeichern (Anforderung 3.2.4 d). Im ersten Teil des Assistenten wird überprüft, ob für die gewählte Universität mehr als ein gültiges, d.h. noch nicht abgelaufenes Semester besteht. Gibt es nur ein valides Semester, wird direkt zum nächsten Dialog gesprungen. Andernfalls erhält der Nutzer die Möglichkeit der Semesterauswahl. Im zweiten Dialog wird der Nutzer aufgefordert, ein Modul bzw. Fach zu wählen. Entscheidet er sich für ein Modul, kann er im nächsten Dialog ein Fach aussuchen, das in diesem Modul vertreten ist. Entscheidet er sich für ein Fach oder ein Modul, das nur aus einem Fach besteht, wird dieser Dialog übersprungen. Im nächsten Teil des Wizards muss sich der Nutzer entscheiden, ob er die Prüfungsfrage direkt on-

line erstellen oder via Excel- bzw. XML-Datei (vgl. Abschnitt 2.1.1) importieren möchte.

4.7.1 Online-Fragenerstellung

Wählt der Nutzer die direkte Erstellung, gelangt er auf eine Seite, die die Eingabe sämtlicher für eine 1-aus-5-Prüfungsfrage relevanter Attribute erlaubt. Inhaltlich zusammengehörige Attribute, wie der Verbund der verschiedenen Antwortmöglichkeiten, sind blau hinterlegt und durch eine weiße Trennlinie von den übrigen Blöcken abgesetzt. Im oberen Bereich der Seite findet der Nutzer die Angaben zu Universität, Modul, Fach, Fachsemester des Moduls, Fachsemester des Fachs und selektiertem Semester. Durch das Drücken eines Knopfes kann der Nutzer eine Vaterfrage wählen. Hierdurch gelangt er zu einem weiteren Dialog, der in tabellarischer Form sämtliche jemals für diese Modul/Fach-Kombination erstellten Fragen dieser Universität enthält. Existiert für die Kombination noch keine Frage, erhält der Nutzer eine entsprechende Mitteilung. Die Tabelle enthält nur die wichtigsten Attribute, um eine Frage zu identifizieren. Der Nutzer kann die Tabelle nach sämtlichen Kriterien mittels Mausklicks auf die entsprechende Tabellenüberschrift auf- oder absteigend sortieren. In der ersten Spalte befindet sich der Radiobutton zum Selektieren der Frage als Vaterfrage. Es folgen Erstellungsdatum und die Angabe, in welchem Semester die Prüfungsfrage erstellt wurde. Nach der Information, ob die Fragen bereits in einer Klausur verwendet wurde, sind Fragetyp, Autor, Lernziele, Fragestamm und Fragetext aufgeführt. Die letzte Spalte enthält für jede Frage einen Link, um Details zur Vaterfrage zu erhalten. Nach dem Klicken dieses Links gelangt der Nutzer zur einer weiteren Seite, die alle mit der Frage assoziierten relevanten Daten enthält.

Nach der Wahl einer Vaterfrage werden sämtliche Textfelder initial mit den Werten der Vaterfrage gefüllt. Auch die Bilder werden übernommen. Der Nutzer kann die von der Vaterfrage übernommenen Daten beliebig modifizieren. Durch einen neu erscheinenden Link kann der Nutzer Details zur Vaterfrage abrufen und mit Hilfe des „Vaterfrage löschen“-Buttons den Vaterfrage-Kontext aufheben. In diesem Fall bleiben die gegebenenfalls schon modifizierten Textfelder und Bilder bestehen. Die nächste Spalte des Fragenerstellungsdialogs enthält den Fragetyp,

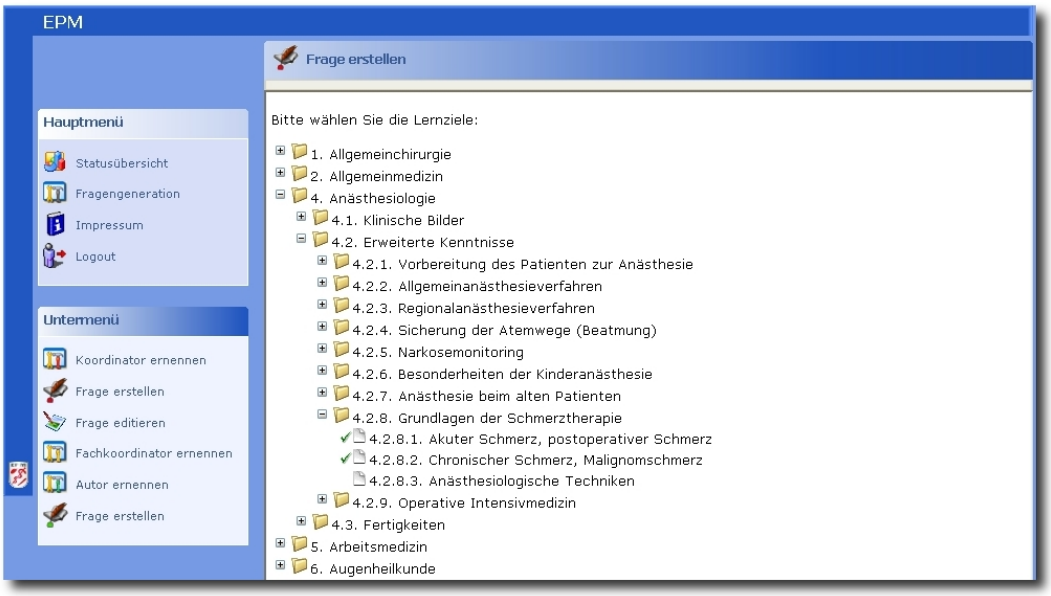


Abbildung 4.6: Auswahl der Lernziele.

mittels der darauf folgenden Auswahlliste kann ein Autor gewählt werden. In der nächsten Zeile können der Frage Lernziele zugeordnet werden. Durch das Drücken des entsprechenden Knopfes wird der Nutzer zum nächsten Dialog geführt, mit dessen Hilfe eine gültige Version des importierten Lernzielkatalogs gewählt werden kann. Ist zurzeit nur eine Version des Lernzielkatalogs gültig, wird direkt zum nächsten Dialog weitergeleitet. Hier kann der Nutzer die verschiedenen Ebenen des Lernzielkatalogs mittels einer auf- und zuklappbaren Baumstruktur durchforsten und die gewünschten Lernziele auswählen (vgl. Abbildug 4.6). Die Klasse zur Anzeige der Baumstruktur wurde rekursiv umgesetzt. Dadurch ist es möglich, Bäume beliebiger Tiefe darzustellen. Der gesamte Lernzielkatalog ist im Application-Kontext abgelegt (vgl. Abschnitt 2.1.4 unten). Gegenüber einer Ablage innerhalb des Session-Kontextes bietet dies die Vorteile des verminderten Speicherplatzes und des schnelleren Zugriffs. Die aktuelle Version des Lernzielkatalogs muss nur einmal aus der Datenbank gelesen und innerhalb des Application-Kontextes gespeichert werden und steht danach allen Nutzern ohne erneuten Datenbankzugriff zur Verfügung. Eine weitere Alternative hätte darin bestanden, für jedes Öffnen und Schließen des Baums eine gesonderte Datenbankanfrage durchzuführen. Dies hätte zu einer großen Anzahl von Datenbankzugriffen ge-

führt. Nach dem Drücken des „Wählen“-Knopfes gelangt der Nutzer wieder zur Haupteingabemaske, in der die selektierten Lernziele nun aufgeführt sind. Durch erneutes Drücken des Knopfes können weitere Lernziele hinzugefügt bzw. bereits gewählte gelöscht werden.

Der nächste farblich abgesetzte Block des Frage Erstellen-Dialogs enthält die Eingabemöglichkeiten für den Fragestamm (vgl. Abbildug 4.7). Zu Beginn des Blockes befindet sich ein Eingabefeld, mit dessen Hilfe der Fragestamm eingegeben werden kann. Mittels HTML-Tags und HTML-Sonderzeichen kann der Nutzer den eingegebenen Text formatieren. In der Hilfefunktion zu dieser Seite (vgl. Anforderung 3.2.9 h) findet sich eine entsprechende Anleitung. Ist im Browser des Nutzers JavaScript aktiviert (vgl. Anforderung 3.2.9 c), müssen die Tags nicht manuell eingetragen werden. Es genügt, die entsprechende Stelle des Textes zu markieren und einen der unter dem Textfeld angebrachten Buttons zu drücken (vgl. Abbildug 4.7 oben). Die entsprechenden Tags bzw. Sonderzeichen erscheinen daraufhin im Text. Die Buttons werden nur eingeblendet, wenn der Browser JavaScript unterstützt. Mit Hilfe der Knöpfe kann der Nutzer einen Teil des Textes als fett (`<b>` und `</b>`), kursiv (`<i>` und `</i>`), unterstrichen (`<u>` und `</u>`), durchgestrichen (`<strike>` und `</strike>`), höhergestellt (`<sup>` und `</sup>`) oder tiefergestellt (`<sub>` und `</sub>`) markieren. Der Button „Zeilenumbr.“ fügt den Tag `<br />` und damit einen Zeilenumbruch ein. Es stehen Buttons für die Sonderzeichen <, >, ©, ®, ‰, μ, α, β und γ zur Verfügung, der Nutzer kann jedoch beliebige andere HTML-Sonderzeichen in den Text einfügen. Auf der Hilfeseite findet er einen Link, der ihn auf eine Seite mit einer Auflistung sämtlicher in HTML zulässiger Sonderzeichen verweist. Hinter den Sonderzeichen < und > verstecken sich die HTML-Zeichen `&lt;` und `&gt;`. Der Nutzer darf für das Kleiner- bzw. Größerzeichen nicht einfach die entsprechenden Zeichen der Tastatur verwenden, da vor der Weiterverarbeitung des Textfeldes geprüft wird, ob der Text die jeweils gleiche Anzahl der Zeichen < und > enthält. Anschließend wird getestet, ob zwischen jeweils einem Klammerpaar nur einer der zulässigen Taginhalte (`b`, `/b`, `i`, `/i`, `u`, `/u`, `strike`, `/strike`, `sup`, `/sup`, `sub`, `/sub`, `B`, `/B`, `I`, `/I`, `U`, `/U`, `STRIKE`, `/STRIKE`, `SUP`, `/SUP`, `SUB` oder `/SUB`) auftritt. Auf diese Weise wird der Nutzer zum einen gezwungen, sich auf die vorgegebenen Formatierungsmöglichkeiten zu beschränken. Zum anderen wird auf diese Weise ein mögliches Cross-

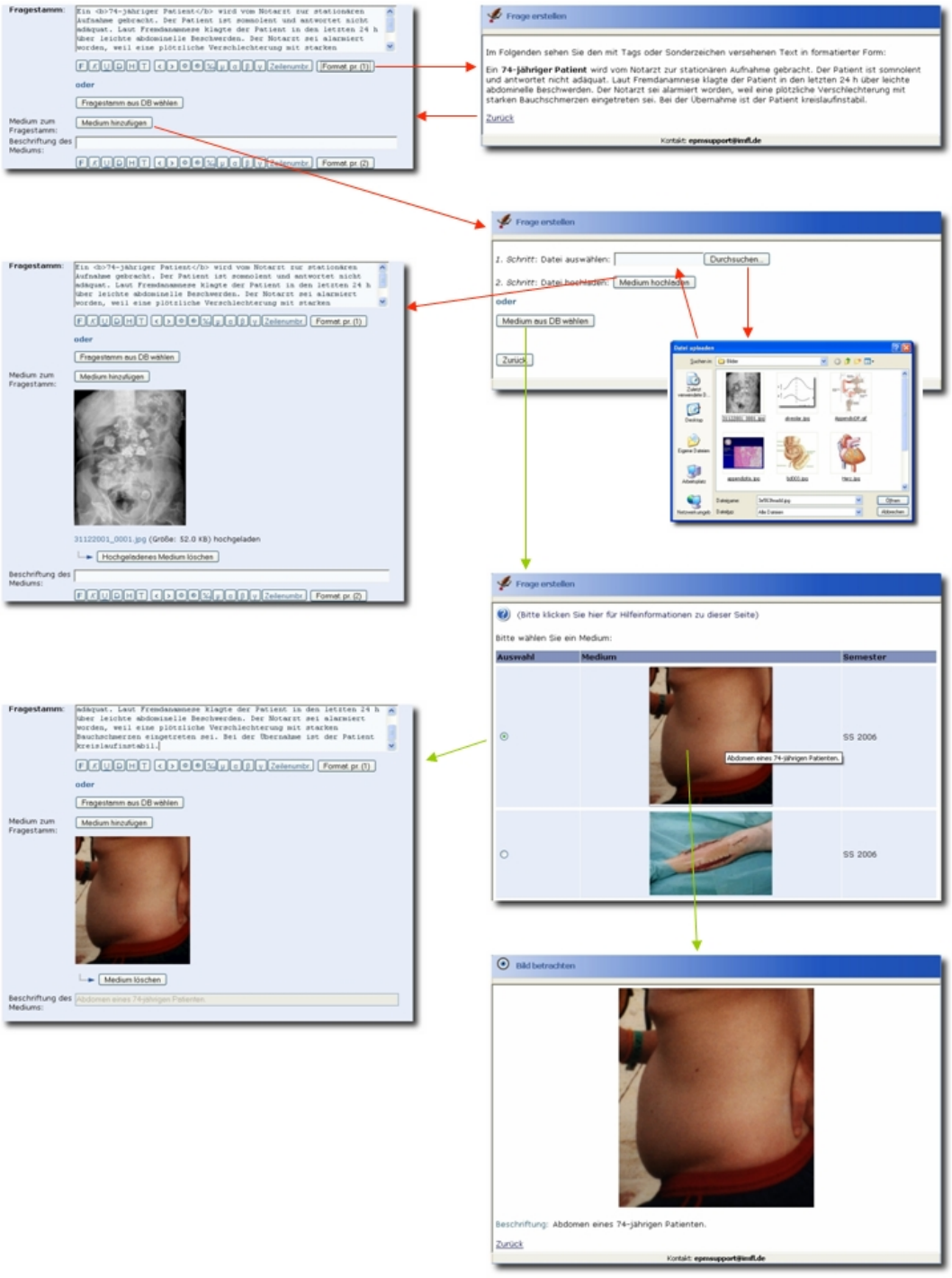


Abbildung 4.7: Möglichkeiten beim Erstellen eines eigenen Fragestamms.

Site-Scripting [2] etwa mittels `<script>`-Tags verhindert (vgl. Anforderung 3.2.9 a). Die Hilfeseite enthält einen extra markierten Bereich, in dem vor der Verwendung der <- und >-Zeichen als Kleiner- oder Größerzeichen gewarnt wird.

Mit Hilfe des Buttons „Format. pr." gelangt der Nutzer zu einer Seite, auf der er das Ergebnis der von ihm vorgenommenen Textformatierungen überprüfen kann (vgl. Abbildung 4.7 oben). Durch das Drücken des Buttons mit der Beschriftung „Medium hinzufügen" gelangt er zu einer weiteren Seite. Hier kann der Nutzer entscheiden, ob er ein eigenes Bild hochladen oder dem Fragestamm ein in der Datenbank hinterlegtes Medium zuweisen möchte (Abbildug 4.7 Mitte rechts):

- Im ersten Fall erlangt der Nutzer durch Drücken des „Durchsuchen"-Buttons Zugriff auf sein lokales Dateisystem. Durch einen Doppelklick entscheidet er sich für ein Medium. Anschließend erscheint die Pfadangabe im Textfeld neben dem „Durchsuchen"-Knopf. Nach dem Anklicken des „Medium hochladen"-Knopfes wird zunächst überprüft, ob eine Datei gewählt wurde und ob diese eine Größe von 4 Megabyte nicht überschreitet. Ist dies der Fall, wird getestet, ob die hochzuladende Datei eine Bilddatei im jpeg-, gif- oder bmp-Format darstellt. Dabei wird nicht nur die Endung, sondern der komplette Dateiheader geprüft. Hierbei kommt die Javaklasse „ImageInfo" ([63]) zum Einsatz. Bei gif-Dateien besteht prinzipiell die Möglichkeit, dass diese aus mehreren „Slices" bestehen, um so eine kleine Animation zu realisieren. Solche animierten gif-Dateien werden vom System erkannt und mit einer entsprechenden Fehlermeldung quittiert. Dadurch wird sichergestellt, dass die Prüfungsfragen auch bei Klausuren in Papierform eingesetzt werden können. Standardmäßig kann Java keine Dateien im bmp-Format verarbeiten. Aufgrund der Verbreitung dieses Bildformats musste dieses Format vom EPM-System ebenfalls unterstützt werden. Zu diesem Zweck wurde die Bibliothek ImageJ ([55]) eingebunden. Nach erfolgreichem Test der Bilddatei gelangt der Nutzer wieder zur Haupteingabeseite, die eine verkleinerte Darstellung der hochgeladenen Bilddatei enthält (Abbildug 4.7 Mitte links). Neben dem ursprünglichen Namen der hochgeladenen Datei findet der Nutzer unter dem Bild die Größe der Datei in Kilobyte und einen Knopf, mit

[2] Bei einem Angriff mittels Cross-Site-Scripting (XSS) wird schädlicher Code in eine dynamisch generierte Webseite eingeschleust, um wichtige Daten der Webseiten-Nutzer, wie deren Zugriffsrechte, zu erlangen (vgl. [64]).

dessen Hilfe das hochgeladene Medium gelöscht werden kann. Durch Klick auf den Thumb der Grafik gelangt der Nutzer zu einer Seite, auf der er das Medium in Originalgröße betrachten kann. Er kann in der nächsten Zeile des Erstellungsdialogs die Beschriftung des Mediums angeben. Bei aktiviertem JavaScript steht ihm die gleiche Buttonleiste wie beim Fragestamm zur Verfügung.

- Entscheidet sich der Nutzer für die Zuweisung eines bereits gespeicherten Mediums, gelangt er durch Drücken des entsprechenden Knopfes zu einer Seite, auf der sämtliche für die gewählte Modul/Fach-Kombination vorhandenen Medien, nach Semester sortiert, als Thumbs aufgelistet sind. Bei der Generierung der Thumbs wurde eine feste Breite vorgegeben, um die Darstellung möglichst übersichtlich zu gestalten. Sofern das Bild mit einer Beschriftung versehen wurde, erscheint diese als Bildbeschreibung, wenn der Nutzer mit dem Mauszeiger über das Bild fährt (Abbildug 4.7 Mitte rechts). Um das Bild in Originalgröße zu betrachten, reicht ein Klick auf die verkleinerte Version der Grafik. Hat der Nutzer das gewünschte Bild selektiert, gelangt er nach Drücken des „Wählen“-Knopfes zurück zum Hauptdialog. Hier findet er neben der Thumb-Version des gewählten Bildes einen Knopf mit der Aufschrift „Medium löschen“ (Abbildung 4.7 unten links). Das Texteingabefeld für die Beschriftung des Mediums ist mit dem in der Datenbank gespeicherten Beschriftung gefüllt und kann vom Nutzer nicht modifiziert werden, das Textfeld ist auf „disabled“ gestellt. Die Button-Leiste zur Vereinfachung der Formatierung der Medium-Kennzeichnung erscheint nicht mehr.

Möchte der Nutzer seiner Frage einen bereits in der Datenbank vorhandenen Fragestamm zuweisen, betätigt er im Haupteingabedialog der Fragenerstellung den Button „Fragestamm aus DB wählen“. Er gelangt dann zu einem Dialog, in welchem er eine Tabelle vorfindet, in der sämtliche für diese Universität/-Modul/Fach-Kombination jemals erstellten Fragestämme samt eventueller Bilder in Thumbnailgröße aufgeführt sind (vgl. Abbildung 4.8 oben). Der Nutzer erhält eine Meldung, wenn für die Universität/Modul/Fach-Kombination noch kein Fragestamm existiert. Die Fragestämme sind primär nach Semester und sekundär nach Fragestammtext sortiert. Gleitet der Nutzer mit dem Mauszeiger

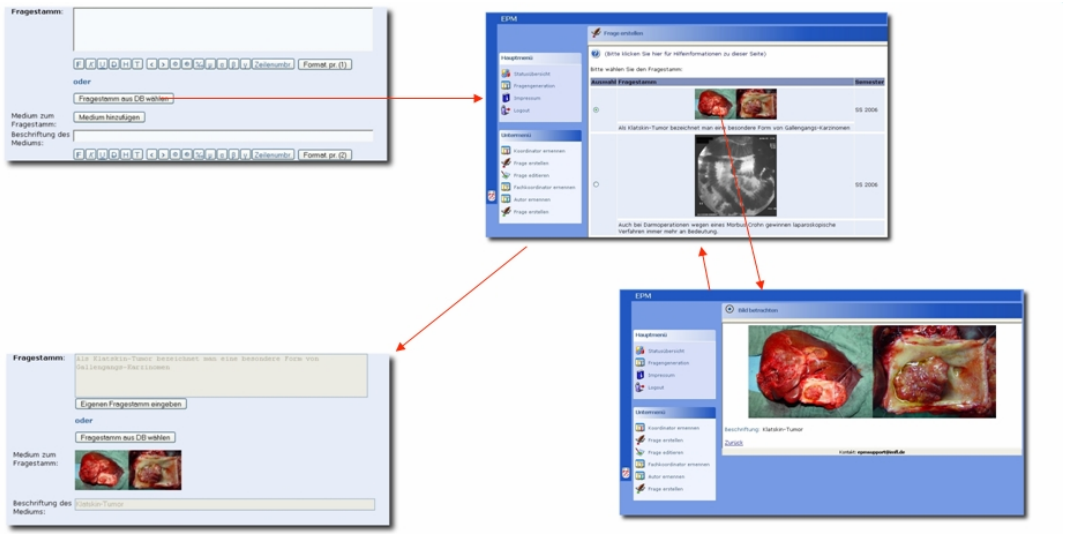


Abbildung 4.8: Wahl eines gemeinsamen Fragestammes.

über ein Thumb, erscheint eine Beschreibung des Mediums, sofern dieses in der Datenbank gespeichert wurde. Durch Klick auf ein Bild gelangt der Nutzer zu einer weiteren Seite, in der das Bild in Vollgröße angezeigt wird. Unter dem Bild findet er die Beschriftung des Mediums (Abbildug 4.8 unten rechts). Drückt der Nutzer im Dialogfenster der Fragestammauswahl den „Wählen“-Button, wird er zurück auf die Hauptseite der Fragenerstellung geleitet (Abbildug 4.8 unten links). Der ausgewählte Fragestamm findet sich in dem entsprechendem Textfeld. Das gegebenenfalls mit dem gewählten Fragestamm ausgewählte Medium ist in Miniaturansicht samt Beschriftung unter dem Fragestamm positioniert. Durch Hinübergleiten über das Medium erhält der Nutzer den Beschriftungstext und durch Anklicken eine Ansicht des Bildes in Originalgröße. Sowohl Textfeld des Fragestammes als auch Textfeld der Medienbeschriftung sind deaktiviert, so dass der Nutzer keine Modifikationen vornehmen kann. Auch die Buttonleisten entfallen. Der Nutzer ist weiterhin in der Lage, durch Betätigung des Knopfes „Fragestamm aus DB wählen“, einen in der Datenbank gespeicherten Fragestamm zu wählen. Durch Drücken des Knopfes „Eigenen Fragestamm eingeben“ wird die Auswahl des Fragestammes rückgängig gemacht und der Eingabedialog in seiner ursprünglichen Form (Abbildung 4.8 oben links) wieder hergestellt.

Im nächsten separat gekennzeichneten Block des Frage Erstellen-Dialogs kann

der Nutzer den Fragetext angeben und der Frage bis zu fünf Medien samt Beschriftung zuweisen. Dieses geschieht analog zur Medienzuweisung beim Fragestamm durch Auswahl eines in der Datenbank gespeicherten Bildes oder durch Hochladen. Unter sämtlichen Text-Eingabefeldern findet sich bei aktiviertem JavaScript die Button-Leiste mit der Tag- bzw. Sonderzeichen-Funktionalität. Im folgenden Block können die verschiedenen Antwortmöglichkeiten eingegeben werden (Antwort A – Antwort E) und im letzten Abschnitt wird angegeben, welche Lösung die richtige ist. Zusätzlich können ein Text zur Lösungsbegründung und ein Lösungsmedium eingefügt werden. Der Prüfungskoordinator muss die durch ein rotes Sternchen gekennzeichneten Attribute Fragetyp, Autor, Fragetext, Antwort A – Antwort E und Lösung angeben. Die restlichen Angaben sind optional. Nachdem der Prüfungskoordinator der Frage alle gewünschten Werte zugewiesen hat, drückt er den „Abschicken“-Knopf. Das Programm überpüft zunächst, ob alle Muss-Felder ausgefüllt sind und gibt im Fehlerfall eine entsprechende Meldung aus. Für jedes Textfeld wird die Tag-Integrität kontrolliert und nach erfolgreicher Prüfung lädt das Programm die nächste Seite des Erstellungsassistenten.

Der nächste Dialog dient dem Nutzer zur Kontrolle seiner Angaben. Sämtliche zuvor gewählten Fragen-Attribute werden auf dieser Seite dargestellt. Zur Überprüfung der Vaterfrage dient ein Link, der auf eine Seite verweist, welche sämtliche relevanten Daten der Vaterfrage auflistet. Im Gegensatz zur vorhergehenden Seite werden die gewählten Grafiken in voller Größe dargestellt, um dem Nutzer ihre wahre Größe zu verdeutlichen und dazu anzuregen, diese unter Umständen zu modifizieren. Wurde ein Medium für den Fragestamm gewählt, aber kein Fragestamm eingegeben, wird das Medium im Kontrolldialog nicht angezeigt und nach dem Abschicken auch nicht gespeichert. Ebenso verhält es sich mit Beschriftungen zu denen kein Medium hochgeladen wurde. Beim Lösungsmedium liegt ein anderer Sachverhalt vor. Es ist möglich, die Lösung nur durch ein Medium, etwa ein entsprechendes Diagramm, zu verdeutlichen. Durch das Drücken des „Ändern“-Buttons gelangt der Nutzer wieder zum vorhergehenden Eingabedialog, das Betätigen des „Speichern“-Knopfes führt zum Speichern der Frage. Beim Sichern der einer Frage zugewiesenen Bilder wird kontrolliert, ob der Nutzer einer Frage dasselbe Medium mehrfach zugewiesen hat. In diesem Fall wird das Medium nur einmal gespeichert. Am Ende des Wizards erhält der Nutzer die Bestätigung über die erfolgreiche Sicherung seiner Frage.

4.7.2 Import einer XML-Datei

Beim Import der Excel-Prüfungsfragen wurden verschiedene Alternativen in Betracht gezogen:

- Der erste Lösungsansatz bestand im Einlesen der nativen Excel-Datei mit der Endung xls. Es wurde nach Java-Bibliotheken gesucht, die sich dieses komplexen Problems annahmen. Die Suche führte zu den Bibliotheken POI ([79]) und Java Excel API ([72]). Der Leistungsumfang beider Bibliotheken stellte sich als unzureichend heraus. Beide Librarys sind nicht in der Lage, Zellinhalte, die mit einer Textformatierung wie Fett- oder Kursivschreibung markiert sind, korrekt auszulesen.

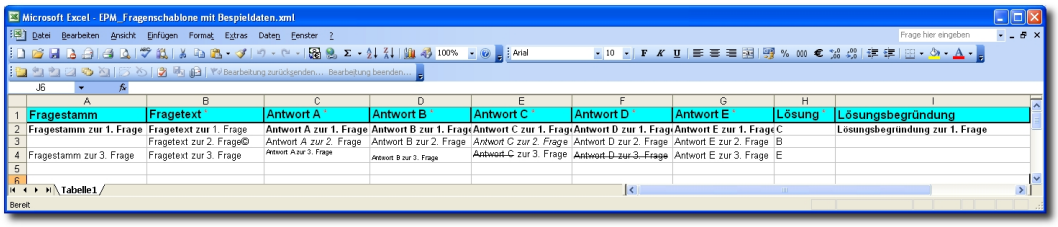
- Die zweite Idee bestand in dem Abspeichern einer Excel-Datei im CSV-Format[3]. Großer Nachteil dieses Ansatz ist, dass in Excel vorgenommene Formatierungen beim Abspeichern als CSV-Datei verloren gehen.

- Um die gewünschte Funktionalität zu liefern, musste eine eigene Lösung implementiert werden. Umfang und Qualität der nicht sehr detailreichen Spezifikation des proprietären Excel-Dateiformats und das partielle Scheitern groß angelegter Projekte, wie des POI-Projekts, sprachen gegen den Einsatz dieses Formats.

- Als Lösung bot sich das seit Excel XP vorhande Excel-eigene XML-Format an. Die Alternativen, das am 1. Mai 2005 veröffentlichte OASIS Open Document Format for Office Applications (Kurzform: OpenDocument-Format) ([57]), ein quelloffener ISO-Standard für die Speicherung von Bürodokumenten, und das von Microsoft entwickelte Office Open XML ([15]) werden von aktuellen Excel-Versionen nicht direkt unterstützt. Der große Vorteil des XML-Ansatzes ist die Möglichkeit, ein solches Dokument mittels eines XML-Prozessors[4] verarbeiten zu können. Zudem existiert für dieses Format eine Dokumentation ([50]).

[3] CSV steht für **C**omma **S**eparated **V**alues. Die Datenblöcke werden durch Kommata oder Semikoli voneinander getrennt.

[4] Ein XML-Prozessor ist ein Programm zum Einlesen und Bearbeiten von XML-Dokumenten.




```
1   <?xml version="1.0"?>
2   <?mso-application progid="Excel.Sheet"?>
3   <Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
4    xmlns:o="urn:schemas-microsoft-com:office:office"
5    xmlns:x="urn:schemas-microsoft-com:office:excel"
6    xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
7    xmlns:html="http://www.w3.org/TR/REC-html40">

23    <Styles>

42     <Style ss:ID="s22">
43      <Font x:Family="Swiss" ss:Bold="1"/>
44     </Style>

57    </Styles>
58    <Worksheet ss:Name="Tabelle1">
59     <Table ss:ExpandedColumnCount="9" ss:ExpandedRowCount="4" x:FullColumns="1" x:FullRows="1"
           ss:DefaultColumnWidth="60">

87      <Row ss:StyleID="s22">
88       <Cell><Data ss:Type="String">Fragestamm zur 1. Frage</Data></Cell>
89       <Cell><ss:Data ss:Type="String" xmlns="http://www.w3.org/TR/REC-html40"><B>Fragetext zur
             </B><Font>1. Frage</Font></ss:Data></Cell>
90       <Cell><Data ss:Type="String">Antwort A zur 1. Frage</Data></Cell>
91       <Cell><Data ss:Type="String">Antwort B zur 1. Frage</Data></Cell>
92       <Cell><Data ss:Type="String">Antwort C zur 1. Frage</Data></Cell>
93       <Cell><Data ss:Type="String">Antwort D zur 1. Frage</Data></Cell>
94       <Cell><Data ss:Type="String">Antwort E zur 1. Frage</Data></Cell>
95       <Cell><Data ss:Type="String">C</Data></Cell>
96       <Cell><Data ss:Type="String">LÃ¶sungsbegrÃ¼ndung zur 1. Frage</Data></Cell>
97      </Row>

119    </Table>

144   </Worksheet>
145   </Workbook>
```

Abbildung 4.9: Eine Excel-XML-Datei. Oben geöffnet in Excel, unten Teile des XML-Quelltextes.

Bei der Implementierung mussten einige Probleme überwunden werden. Es musste herausgefunden werden, welcher Kodierung das XML-Dokument unterlag. Entgegen der Vorgaben des W3C (vgl. Abschnitt 2.1.1 oben) ist in der XML-Datei die Art der Kodierung nicht angegeben (vgl. Abbildung 2.1 Z 1). Während die Datei ohne Probleme mit der unter Windows laufenden Testumgebung eingelesen werden konnte, zeigten sich bei der Linux-Version Schwierigkeiten mit Sonderzeichen. Es stellte sich heraus, dass der Kodierungsstandard „CP1252“, auch bekannt unter der Bezeichnung „Windows-1252“, verwendet wurde. Es wurde ergründet, an welcher von mehreren potentiellen Stellen im Quellcode die Kodierung angegeben werden musste oder ob dies an mehreren Positionen zu erfolgen hatte. Schwierigkeiten gab es auch bei der Verwendung von XPath, einer vom W3-Konsortium entwickelten Abfragesprache, um auf Teile von XML-Dokumenten zuzugreifen. Die Java-Implementierung arbeitete zum Teil entgegen der Spezifikation. Die im CP1252-Standard dargestellten Sonderzeichen mussten in die entsprechenden HTML-Sonderzeichen konvertiert (vgl. z. B. Z 96 in Abbildung 4.9) und der Text auf unzulässige Zeichen untersucht werden. Fehler, die beim Hochladen der Datei unterlaufen können, waren abzufangen. Es wird überprüft, ob ein gültiger Lösungsbuchstabe angegeben ist, wobei von etwaigen Formatierungseigenschaften abstrahiert wird. Es trat das Problem auf, dass eine Excel-Datei aus mehreren Arbeitsblättern bestand, in der Anzeige mittels Excel jedoch nur eines direkt sichtbar war. Um diesem Hindernis in Zukunft aus dem Weg zu räumen, wird bei der Existenz von mehreren Arbeitsblättern eine entsprechende Fehlermeldung ausgegeben. Die Speicherung leerer Zellen erfolgt in Excel nicht einheitlich. Besaß eine Zelle zuvor einen Inhalt, werden nicht zwingend die zugehörigen `<Cell>`-Tags gelöscht. Diese Artefakte führten zu Fehlern beim Einlesen einer Datei. Dieses Problem wurde durch ein Ignorieren von Zeilen und Spalten, die nur aus leeren Zellen bestehen, behoben.

Zur Formatierung des Textes, etwa als dick oder kursiv, existieren verschiedene Möglichkeiten. Es kann entweder

- nur ein Teil einer Zelle,
- eine ganze Zeile oder
- eine komplette Spalte

formatiert sein. Die Zeilen 87–97 in Abbildung 4.9 unten repräsentieren die zweite Zeile der im oberen Teil der Abbildung dargestellten Excel-Datei. Die Zeile ist als fett markiert. Dieses zeigt sich durch die Attributierung `ss:StyleID="s22"`, die auf die Zeilen 42–44 verweist. In Zeile 43 ist die Reihe durch `ss:Bold="1"` als fett definiert. Die Zelle B2 bzw. die Zeile 89 im Quelltext stellen einen Sonderfall dar. Der Text „Fragetext zur“ ist durch die Tags `<B>` und `</B>` zusätzlich zur ganzen Zeile als fett markiert. In diesem Fall wird nur die Markierung des Zellabschnittes berücksichtigt und die Formatierung der Zeile ignoriert. In analoger Weise werden Sonderfälle wie die gleichzeitige Formatierung einer Zelle durch eine Zeilen- und ein Spaltenattribut behandelt.

Hat sich der Nutzer dazu entschlossen, in Excel erstellte Fragen zu importieren bzw. einen Import via XML vorzunehmen, gelangt er nach Auswahl dieser Option zu der entsprechenden Seite. Im oberen Bereich der Seite findet sich ein Link, durch den der Nutzer sich eine Excel-Vorlage herunterladen kann. Die Datei wurde im Excel-eigenen XML-Format abgespeichert. Die Vorlage enthält in der ersten Zeile die Spaltenüberschriften „Fragestamm“, „Fragetext“, „Antwort A“, „Antwort B“, „Antwort C“, „Antwort D“, „Antwort E“, „Lösung“ und „Lösungsbegründung“. Unterhalb der Überschriften kann der Nutzer die Fragen in jeweils einer Zeile eintragen (vgl. Abbildug 4.9 oben). Die Muss-Felder entsprechen den Feldern im oben beschriebenen Hauptdialog der Fragenerstellung und sind durch ein rotes Sternchen markiert. Durch Drücken des „Durchsuchen“-Buttons öffnet sich ein Fenster, mit dessen Hilfe der Nutzer die entsprechende Excel-Datei bzw. XML-Datei aussuchen kann. Der gewählte Pfad erscheint links neben dem „Durchsuchen“-Knopf. Nach Betätigen des „Datei hochladen“-Knopfes wird überprüft, ob eine Datei ausgewählt wurde, ob diese das geforderte Format besitzt und ob die Muss-Felder ausgefüllt wurden. Nach erfolgreicher Prüfung gelangt der Nutzer zu einer Seite, auf der er mittels Checkboxen wählen kann, welche Prüfungsfragen er editieren möchte. Auf diese Weise muss der Nutzer nicht die XML-Datei editieren, um nach einem unterbrochenen Einlesen der Fragen an der entsprechenden Stelle fortfahren zu können. Hat der Nutzer eine Auswahl vorgenommen, gelangt er zum unter 4.7.1 beschriebenen Eingabedialog. Die Textfelder sind mit den adäquaten Einträgen der XML-Datei initialisiert. Die oben erklärten Formatierungsmerkmale wie Kursiv-Schreibung oder Unterstreichung und die dort beschriebenen Sonderzeichen werden in die entsprechenden HTML-Befehle transformiert. Oben

auf der Seite erhält der Nutzer die Angabe, die wie vielte der ausgewählten Fragen er momentan bearbeitet. Nach dem Speichern einer Prüfungsfrage hat der Nutzer die Wahl, den Importvorgang abzubrechen oder mit der nächsten Frage fortzufahren.

4.8 Frageneditierung für den Prüfungskoordinator

Der Menüpunkt „Frage editieren“ innerhalb des „Fragengeneration“-Submenüs dient dem Prüfungskoordinator dazu, eine bereits erstellte Prüfungsfrage zu editieren (Implementierung von Anforderung 3.2.4 e). Zunächst wird überprüft, ob es zurzeit mehr als ein Semester gibt für das bereits Module und Fächer angelegt wurden. Ist dies der Fall, gelangt der Nutzer zu einem Eingabedialog, in dem er sich für ein Semester entscheiden muss. Andernfalls wird der Nutzer direkt zur nächsten Seite weitergeleitet, auf der er ein Modul bzw. Fach auswählen kann. Wählt der Nutzer ein Fach bzw. ein Modul, das im selektierten Semester nur aus einem Fach besteht, wird der nachfolgende Dialog übersprungen, in dem er sich für ein dem Modul zugehöriges Fach entscheiden muss. Anschließend wird der Nutzer auf eine Seite geleitet, die eine tabellarische Übersicht über sämtliche gespeicherten Fragen der gewählten U/S/M/F-Kombination enthält. Analog zum Auswahldialog bei der Zuweisung einer Vaterfrage (vgl. Abschitt 4.7.1) kann der Nutzer die Tabellendaten sortieren und erhält über einen Link Zugriff auf die Details einer zu wählenden Frage. Anders als bei der Wahl der Vaterfrage ist aufgelistet, ob eine Frage nach Aufforderung oder autark erstellt wurde. Existiert für die gewählte Kombination noch keine Prüfungsfrage, erscheint eine entsprechende Meldung. Nach Auswahl einer Frage wird zunächst überprüft, ob die zu bearbeitende Prüfungsfrage einen Fragestamm besitzt. Ist dies der Fall, wird ermittelt, ob der Fragestamm nur von dieser Frage oder auch von anderen Fragen verwendet wird. Im ersten Fall gelangt der Nutzer direkt zur nächsten Seite, in der er sämtliche Attribute der Frage modifizieren kann. Wird der Fragestamm von mehr als einer Frage verwendet, erfolgt eine weitere Fallunterscheidung:

- Wird die Frage nur von Fragen des gleichen Semesters verwendet, wird der

Prüfungskoordinator zu einem Dialog weitergeleitet, in dem er entscheiden kann, ob die Veränderungen am Fragestamm und den mit ihm assoziierten Grafiken für

- sämtliche Fragen gespeichert werden sollen oder
- nur für die selektierte Frage.

Im letzteren Fall wird beim Speichern eine Kopie der entsprechenden Datenbankeinträge angelegt. Hat sich der Nutzer für eine der beiden Möglichkeiten entschieden, wird er zum Editierungsdialog weitergeleitet. Auf dem Kopf der Seite erscheint ein durch ein Warndreieck markierter Hinweis an den Nutzer, dass der Fragestamm von mehreren Fragen eines Semesters verwendet wird und etwaige Änderungen (keine) Auswirkungen auf die übrigen Prüfungsfragen haben werden.

- Es kann der Fall auftreten, dass der Fragestamm in mehreren Fragen aus verschiedenen Semestern Verwendung findet. Trifft dieser Fall zu, wird der Nutzer auf eine weitere Seite geleitet, auf welcher er sich entscheiden muss,
  - ob die Veränderungen nur für den Fragestamm der selektierten Frage Gültigkeit besitzen sollen oder
  - ob die Modifikationen Auswirkungen auf sämtliche Fragen des gewählten Semesters haben.

  In beiden Fällen können die Fragestämme von Fragen zurückliegender Semester nicht modifiziert werden. Hat sich der Prüfungskoordinator für eine Möglichkeit entschieden, wird er zur Editierungsseite weitergeleitet, auf welcher in beiden Fällen eine entsprechende Warnmeldung erscheint.

Der Editierungsdialog gestaltet sich ähnlich dem oben beschriebenen Dialog zur Erstellung einer Prüfungsfrage. Zusätzlich enthält er die Angabe, wann eine Prüfungsfrage erstellt wurde und ob und wenn ja, in welchen Klausuren sie bereits Verwendung fand. Wurde die Frage bereits in einer Klausur eingesetzt, erscheint oben auf der Seite ein Hinweis, um den Nutzer vor der nachträglichen Editierung der Frage zu warnen.

Die Implementierung des Editierungsdialogs stellte sich als sehr komplex heraus. Die große Zahl der Fallunterscheidungen führte dazu, dass an sehr vielen Stellen Session-Attribute (vgl. Abschnitt 2.1.4 unten) gesetzt und wieder gelöscht werden mussten. Beim Speichern der modifizierten Fragedaten fielen zunächst 176 Fallunterscheidungen an. Diese Zahl konnte auf 90 herab gesenkt werden. Es wird nicht mehr an allen Stellen kontrolliert, ob Veränderungen vorgenommen wurden. Die alten Relationen und Datenbankeinträge werden gelöscht (bzw. auf ungültig gesetzt) und die neuen eingefügt.

4.9 Fragenerstellung für den Autor

Unter dem Menüeintrag „Frage erstellen“ des Hauptmenüpunktes „Fragengeneration“ verbirgt sich die Implementierung der Anforderungen 3.2.7 c und 3.2.8 a. Um die Menüpunkte zur Fragenerstellung von Prüfungskoordinator und Autor auch optisch unterscheidbar zu machen, wurden zwei unterschiedliche Icons gewählt, die neben dem entsprechenden Menüeintrag erscheinen.

- Besitzt der Nutzer beide Autorrollen (vgl. Abschnitt 3.2.1 bzw. 3.2.2) wird er zunächst zu einem Dialog geleitet, in dem er sich entscheiden muss,
  - ob er eine vom Fachkoordinator angeforderte Frage erstellen oder
  - ob er autark eine Prüfungsfrage im System abspeichern möchte.
- Besitzt ein Nutzer nur die Rolle des an ein Fach gebundenen Autors wird dieser Auswahldialog übersprungen. Der Nutzer kann eine Prüfungsfrage nur autark erstellen.

Besitzt ein Nutzer die Rolle des an eine konkrete U/S/M/F-Komposition gebundenen Autors, verfügt er gleichzeitig über die Rolle eines an ein Fach gebundenen Autors (vgl. Abschnitt 4.6 unten). Der Auswahldialog erscheint grundsätzlich bei jedem Autor, der beauftragt wurde, Prüfungsfragen zu erstellen.

- Entscheidet sich der Nutzer, eine Prüfungsfrage autark zu erstellen, gelangt er zunächst zu einem Dialog, in dem er sich für ein gültiges, d.h. aktuel-

les oder zukünftiges Semester mit entsprechenden Modulen und Fächern entscheiden muss. Existiert nur ein valides Semester, wird dieses automatisch gesetzt und dieser Dialog übersprungen. Im folgenden Dialog werden sämtliche Module und Fächer des gewählten Semesters aufgelistet, in denen diejenigen Fächer enthalten sind, für die der Nutzer die Rolle des an ein Fach gebundenen Autors inne hat. Der Nutzer muss sich für ein Modul bzw. Fach entscheiden. Existiert nur ein einziges Modul bzw. Fach, das diesen Anforderungen genügt, wird dieses gewählt und der Nutzer wird direkt zur nächsten Seite geführt. Hat der Nutzer ein Modul bestimmt, das aus mehr als einem Fach besteht, gelangt er zur einer Auswahlseite, auf der er die Wahl zwischen allen dem Modul inhärenten Fächern treffen muss. Entschied sich der Nutzer für ein Fach oder ein Modul, das nur aus einem Fach besteht, wird dieser Auswahldialog übersprungen. Auf der nächsten Seite kann der Nutzer wählen, ob er die Prüfungsfrage online erstellen oder via Excel-XML-Datei importieren möchte. Sowohl Online-Erstellung als auch Import via Excel-Datei laufen für den Nutzer analog zur Fragenerstellung für den Prüfungskoordinator ab (vgl. Abschnitt 4.7). Im Gegensatz zum Prüfungskoordinator kann der Autor seiner Prüfungsfrage keinen fremden Autor zuweisen. Er hat die Verpflichtung, der Frage ein oder mehrere Lernziele zuzuweisen.

- Hat sich der Nutzer dazu entschlossen, eine vom Fachkoordinator angeforderte Prüfungsfrage zu erstellen, gelangt er zu einer Seite, auf der sich eine tabellarische Darstellung aller U/S/M/F-Kombinationen findet, zu denen er die Rolle eines an eine konkrete U/S/M/F-Kombinationen gebundenen Autors besitzt. Die Tabelle enthält eine Auflistung mit den Angaben zu Modul, Fach, Semester, Deadline und Fragenratio in Form eines Kuchendiagramms. Dies erleichtert die Bedienung des Programms und erspart dem Autor das Aufrufen der entsprechenden Statusübersicht (vgl. Anforderung 3.2.9 b). Nach der Auswahl einer U/S/M/F-Komposition gelangt der Nutzer zu der oben beschriebenen Seite, auf der er die Wahl trifft, eine Prüfungsfrage online zu erstellen oder via Excel-Datei zu importieren. Der weitere Vorgang gleicht für den Nutzer exakt dem Vorgehen beim autarken Erstellen einer Prüfungsfrage. Intern werden zusätzliche Datenbankeinträge vorgenommen, die die Anzahl der erstellten Prüfungsfragen bei Prüfungskoordinator und

Fachkoordinator um den Faktor eins erhöhen. Handelt es sich um ein Modul, das aus mehreren Fächern besteht, wird auch hier ein entsprechendes Update vorgenommen. In der Datenbank wird zusätzlich gespeichert, dass die Frage nach Aufforderung erstellt wurde.

4.10 Frageneditierung für den Autor

Der Menüpunkt „Frage editieren“ innerhalb des „Fragengeneration“-Untermenüs realisiert die Anforderungen 3.2.7 d und 3.2.8 b. Zur Unterscheidung diese Menüpunktes vom entsprechenden Menüeintrag für den Prüfungskoordinator sind beide Einträge mit unterschiedlichen Icons versehen.

Nach dem Anklicken dieses Menüeintrags gelangt der Nutzer zu einer Seite, auf der er sich für ein gültiges Semester entscheiden muss. Existiert nur ein gültiges Semester, wird dieser Dialog übersprungen und das Semester automatisch gesetzt. Hat der Nutzer noch keine Prüfungsfrage für eines der gültigen Semester erstellt, erscheint eine adäquate Meldung. Auf diese Weise wird vermieden, dass sich der Nutzer durch eine Vielzahl weiterer Dialoge arbeiten muss, um diese Information zu erlangen (vgl. Anforderung 3.2.9 b). Die nächsten beiden Seiten dienen der Auswahl von Modul bzw. Fach. Die Programmlogik folgt der in Abschnitt 4.9 beschriebenen Vorgehensweise. Unter Umständen wird der Dialog zur Wahl eines einem Modul inhärenten Faches ausgespart. Nach der Festlegung der gewünschten U/S/M/F-Kombination gelangt der Nutzer zu einer Übersichtsseite, die eine tabellarische Auflistung seiner für diese Kombination erstellten Fragen erhält. Hier findet der Nutzer sowohl seine autark als auch die von ihm nach Aufforderung erstellten Prüfungsfragen. Die Information, ob die Frage bereits einer Klausur zugeordnet ist, ist aufgeführt. Der Nutzer hat über einen Link Zugang zu sämtlichen Details einer Frage. Wählt der Nutzer eine bereits einer Klausur zugeordnete Prüfungsfrage, wird er darüber informiert, dass er die Frage aufgrund dieses Umstandes nicht mehr verändern darf. In der Praxis tritt dieser Fall nur selten auf, da die Klausuren üblicherweise am Ende eines jeden Semesters stattfinden. Nach der Wahl einer Prüfungsfrage existieren vier verschiedene Möglichkeiten hinsichtlich des weiteren Verlaufs der Frageneditierung:

- Der Standard-Fall tritt ein, wenn der Nutzer der Frage zuvor keinen Fragestamm zugewiesen hat oder wenn der Fragestamm nur in dieser Frage verwendet wurde. Der Nutzer wird direkt auf die Seite zum Editieren einer Prüfungsfrage geleitet und darf den Fragestamm beliebig modifizieren. Der Editierungsdialog ähnelt dem entsprechenden Dialog für den Prüfungskoordinator (vgl. Abschnitt 4.8).

- Wird der Fragestamm von mehreren Fragen desselben Semesters verwendet und wurden die Fragen vom selben Autor erstellt, gelangt der Nutzer zu einer Seite, auf der er sich entscheidet, ob die Veränderungen am Fragestamm entweder

  - nur für die gewählte Frage oder

  - für sämtliche betroffenen Fragen des gewählten Semesters vorgenommen werden sollen.

  In beiden Fällen erscheint zu Beginn des nachfolgenden Dialogs ein entsprechender Warnhinweis.

- Ist der Fragestamm mehreren Fragen des angemeldeten Nutzers aus verschiedenen Semestern zugeordent, hat er die Wahl, die Veränderungen

  - nur für die gewählte Frage oder

  - für alle betroffenen Fragen des aktuellen Semesters anzuwenden.

  Eine Veränderung des Fragestamms zurückliegender Semester ist nicht möglich. Nach seiner Entscheidung gelangt der Nutzer auf die Seite des eigentlichen Editierungsdialogs, auf der ein adäquater Warnhinweis erscheint.

- Der vierte Fall tritt ein, wenn der Fragestamm auch einer Frage eines anderen Autors zugeordnet ist. In diesem Fall darf der Autor den Fragestamm nicht verändern. In einer betreffenden Nachricht wird er darüber in Kenntnis gesetzt, mit dem Hinweis, sich in dringenden Fällen an einen der zuständigen Prüfungskoordinatoren zu wenden, der die Veränderungen vornehmen kann. Entscheidet sich der Autor in diesem Fall dafür, seine Frage von einer Vaterfrage abzuleiten, wird der Fragestamm der Vaterfrage nicht mit

übernommen. Im Dialog zur Auswahl einer Vaterfrage wird der Nutzer über diesen Sachverhalt vorab informiert.

Beim Speichern der Prüfungsfrage werden die verschiedenen Fälle unterschieden. Ein zusätzlicher Fall tritt ein, wenn ein anderer Autor einer seiner Fragen denselben Fragestamm zugewiesen hat, während der angemeldete Autor gerade dabei war, seine Prüfungsfrage zu editieren. In diesem Fall werden die von ihm am Fragestamm vorgenommenen Änderungen nicht gespeichert und der Nutzer erhält auf der finalen Seite des Editierungsdialogs eine entsprechende Meldung. Eine ähnliche Nachricht erhält der Nutzer, wenn der vom Autor verwendete Fragestamm von einem Prüfungskoordinator inzwischen einer weiteren Frage ebendieses Autors zugewiesen wurde. Auch in diesem Fall werden die am Fragestamm vorgenommenen Änderungen nicht gespeichert.

4.11 CADS-Anbindung

Um das EPM-Systems an den CADS (vgl. Abschnitt 2.1.9) anbinden zu können (Anforderung 3.2.9 d), mussten entsprechende Java-Bibliotheken entwickelt werden. Es entstand eine Version für Webanwendungen, genannt `CADSLib` und eine Version für Standalone-Applikationen. Diese trägt den Namen `CADSLibSA`. Die Standalone-Version wurde entwickelt, um das Mahnsystem, das beim Überschreiten einer Deadline in Aktion tritt, implementieren zu können (vgl. Abschnitt 4.12). Zur Realisierung wurde auf die Java-Bibliotheken HttpClient ([76]) und dom4J ([71]) zurückgegriffen. Der HttpClient wird zu der auf HTTP beruhenden Kommunikation der CADSLib-Versionen mit den CADS-Servlets verwendet. Die XML-Anfragen und -Antworten werden mit Hilfe vom dom4J erzeugt bzw. gelesen. Auf Details zur CADSLib-Implementierung muss an dieser Stelle aus Sicherheitsgründen verzichtet werden.

4.12 Mahnsystem

Das Mahnsystem (Anforderung 3.2.9 f) wurde in Form einer eigenständigen Java-Applikation realisiert. Da sie im gleichen Projekt wie die EPM-Webapplikation angesiedelt ist, werden größtenteils dieselben Klassen, z. B. für Datenbankzugriffe und Mailfunktion, genutzt. Eine Woche vor Ablauf einer Deadline wird eine Erinnerungsmail versendet. Drei Tage zuvor eine weitere Erinnerung. Am Tag des Erreichens der Deadline wird die erste Mahnung verschickt, die restlichen Mahnungen erfolgen drei bzw. sieben Tage nach Überschreiten der Deadline.

4.13 Exception-Handling

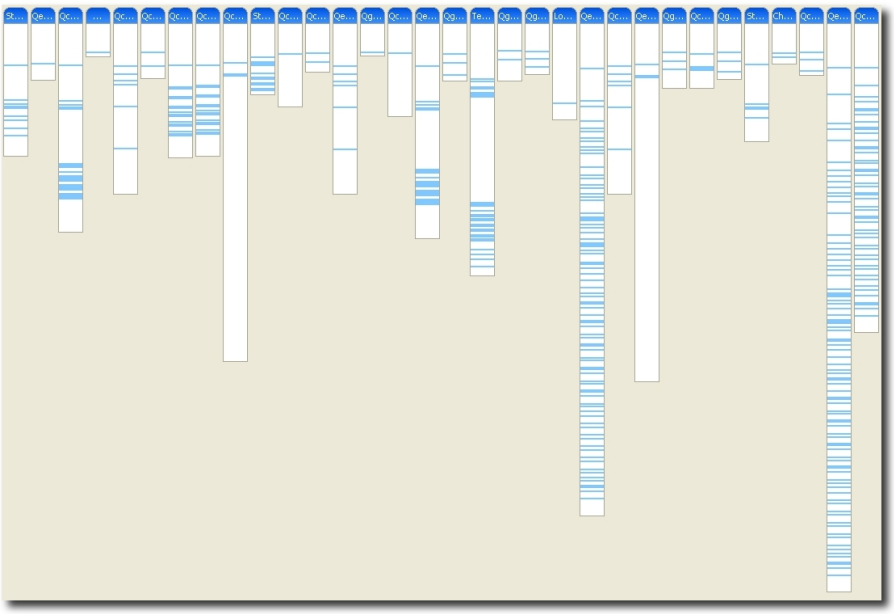


Abbildung 4.10: Jeder Balken stellt eine Klasse des EPM-Systems dar. Die Länge eines Balkens repräsentiert den Umfang des Quellcodes. An den blau markierten Stellen können Exceptions geworfen werden, die mittels AOP-Technologie behandelt werden.

Das Exception-Handling (vgl. die Fußnote im Abschnitt 2.1.3) ist im EPM-System derart realisiert, dass Exceptions bis an die oberste Ebene weitergereicht werden. Dort wird in vielen Fällen eine entsprechende Fehlermeldung an den Nutzer

ausgegeben. Dies geschieht bei allen abzusehenden Fehlermeldungen. Beispiele für solche Exceptions sind der Ablauf der Sitzung, Probleme bei der CADS-Kommunikation oder ein Fehler beim Zugriff auf die Datenbank. Insgesamt sind im EPM-eigenen Exception-Paket fünfzehn Klassen definiert. Zur Ausgabe der Fehlerkette auf Bildschirm bzw. in die `EPM.log` genannte EPM-spezifische Protokolldatei (vgl. Anforderung 3.2.9 i) und in die generelle Tomcat-Logdatei `catalina.out` dient die Klasse `ExceptionChainPrinter`. Die Ausgabe erfolgt unter Angabe einer von fünf Eskaltationsstufen. Der Fehlerkette kann als debug, info, warn, error oder fatal deklariert werden. Intern greift die `ExceptionChainPrinter`-Klasse auf die Log4j-API ([1]) zurück. Hierbei handelt es sich um eine etablierte Logging-Bibliothek. Diese kann mit Hilfe der Datei `log4j.properties` konfiguriert werden.

Der Aufruf der Print-Methode der `ExceptionChainPrinter`-Klasse ist als Aspekt (vgl. Abschnitt 2.1.3) umgesetzt. Auf diese Weise wird vermieden, dass diese Methode an jeder Stelle, an der potentiell eine Exception geworfen werden kann, aufgerufen werden muss. Stattdessen erfolgt die Konfiguration der Behandlung der unterschiedlichen Exceptions und der Aufruf nur an einer einzigen Stelle. Abbildung 4.10 veranschaulicht dies anhand einer Auswahl an Klassen, die jeweils durch einen Balken repräsentiert werden. Bei allen blau markierten Streifen greift der Logging-Aspekt. An all diesen Stellen hätte alternativ die Print-Methode mit entsprechenden Übergabeparamtern aufgerufen werden müssen. Änderungen im gewünschten Logging-Verhalten lassen sich so wesentlich effizienter, stringenter und weniger fehlerträchtig realisieren.

4.14 Rechtesystem

Auf jeder Seite innerhalb des EPM-Systems muss kontrolliert werden, ob die Sitzung des Nutzers noch gültig ist und ob er die zum Aufruf einer Seite nötigen Rechte besitzt. Abbildung 4.11 veranschaulicht die Klassenstruktur, auf die hierfür zurückgegriffen wird. In Struts steht hinter jeder Seite eine Klasse, die von einer `Action` genannten Klasse abgeleitet ist. Die `Action`-Klasse ist eine Erweiterung eines Servlets (vgl. Abschnitt 2.1.4). Sämtliche in Servlets in-

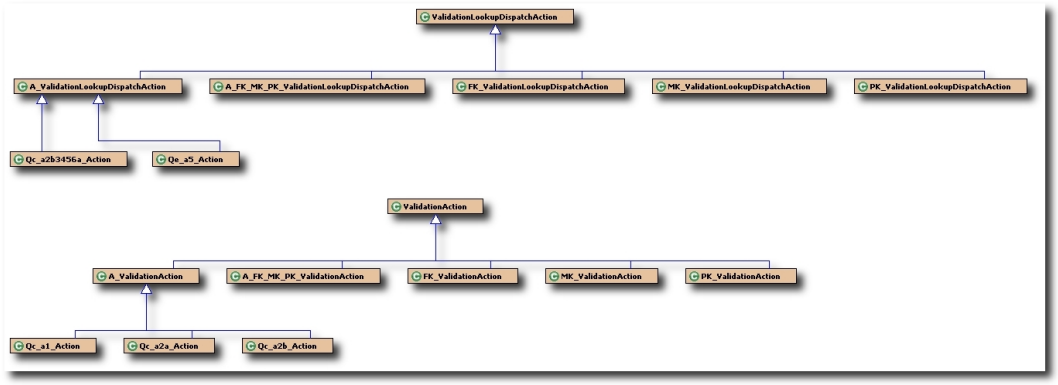


Abbildung 4.11: Übersicht über wichtige Klassen des Rechtesystems

tegrierten Funktionalitäten und Befehle stehen hier zur Verfügung. Die Klasse `ValidationAction` (vgl. Abbildung 4.11 unten) ist von der Struts-`Action`-Klasse abgeleitet. Die Klasse überprüft, ob der Nutzer noch im System angemeldet ist. Von dieser Klasse sind die Klassen `A_ValidationAction`, `FK_ValidationAction`, `MK_ValidationAction`, `PK_ValidationAction` und `A_FK_MK_PK_ValidationAction` abgeleitet. Die `A_ValidationAction` überprüft, ob ein Nutzer die Rolle eines an ein Fach gebundenen Autors besitzt und wirft im Fehlerfall eine entsprechende Exception. Da der an eine U/S/M/F-Kombination gebundene Autor zugleich auch die Rechte für die entsprechenden Fächer besitzt, musste für diese Rolle keine eigene Klasse implementiert werden. Die Klassen `FK_ValidationAction`, `MK_ValidationAction` und `PK_ValidationAction` kontrollieren entsprechend die Rollen Fachkoordinator (FK), Modulkoordinator (MK) und Prüfungskoordinator (PK). Die `A_FK_MK_PK_ValidationAction` prüft, ob der Nutzer mindestens eine der Rollen inne hat. Von diesen Klassen werden die Klassen abgeleitet, die hinter den entsprechenden Seiten stehen. Im Klassendiagramm sind beispielhaft die Klassen `Qc_a1_Action`, `Qc_a2a_Action` und `Qc_a2b_Action` aufgeführt. So wird bei jedem Aufruf einer Seite überprüft, ob der Nutzer noch im System angemeldet ist und ob er über die notwendigen Rechte verfügt.

Für Seiten, die mehr als einen Button besitzen, existiert im Struts-Framework eine eigene `LookupDispatchAction` genannte Klasse. Hier wird für jeden Button eine Methode hinterlegt, die durch das Drücken dieses Knopfes aufgerufen wird.

Wie in Abbildung 4.11 oben dargestellt, existiert für diese Klasse eine ähnliche Vererbungshierarchie wie für die reguläre `Action`-Klasse. Durch Angaben in der jedem Java-Web-Projekt beliegenden Konfigurations-Datei `web.xml` wird verhindert, dass von außerhalb direkt auf die JavaServer Pages zugegriffen werden kann, unter Umgehung der vorgeschalteten `Action`-Klassen.

Beim Hochladen einer Bilddatei wird diese auf der Festplatte des Servers in einem „von außen“ unzugänglichen Verzeichnis zwischengespeichert. Damit das Bild und der entsprechende Thumbnail dennoch im Browser dargestellt werden können, existiert die Klasse `ShowTmpPicAction`. Diese liest die angeforderte Datei aus dem entsprechenden Verzeichnis und reicht den Datenstrom unter Angabe des adäquaten Content-Types an den Browser weiter. Der Browser benötigt den Content-Type, um die Bilddatei korrekt darstellen zu können. An allen Stellen einer Internetseite, an denen ein hochgeladenes Bild dargestellt wird, erfolgt ein Aufruf der `ShowTmpPicAction`. Um den Missbrauch dieser Klasse zu verhindern, existieren vier Mechanismen. Als erstes wird der Name des darzustellenden Bildes nur verschlüsselt an die `ShowTmpPicAction` übergeben und von dieser wieder dechiffriert. Der zweite Schutz besteht darin, dass der Name eines hochgeladenen Bildes sowohl statische als auch dynamische Anteile besitzt. Da unter anderem die SessionID[5] , das aktuelle Datum und die momentane Uhrzeit in den Namen mit einfließen, ist dieser praktisch nicht zu erraten. Als dritte Hürde wird die Existenz eines bestimmten Session-Attributs geprüft. Dies bewirkt, dass nur am System angemeldete Nutzer die Klasse nutzen können. Die letzte Hürde zum unbefugten Betrachten eines hochgeladenen Bildes stellt die Tatsache dar, dass dieses nur sehr kurze Zeit auf dem Server zwischengespeichert wird. Nach dem Sichern der zugehörigen Frage werden die hochgeladenen Medien von der Festplatte gelöscht. Um zu verhindern, dass ein angemeldeter Nutzer über die Klasse Zugang zum Dateisystem erhält, wird der entschlüsselte Dateiname auf kritische Zeichenfolgen wie „..“ und „/“ überprüft.

Tests und Erfahrungen in anderen Projekten haben gezeigt, dass die Darstellung mehrerer, direkt aus der Datenbank geladener Bilder zu viel Zeit in Anspruch

[5] Jeder Nutzer erhält bei der Anmeldung im System eine SessionID. Diese schickt der Browser bei jeder erneuten Anfrage (Request, vgl. Abschnitt 2.1.5) an den Server mit. Hierdurch können die serverseitig gespeicherten Daten bei jedem Zugriff eindeutig einem Nutzer zugeordnet werden.

nimmt. Aus diesem Grund wurde ein Caching-System entwickelt, das die Bilder beim ersten Laden aus der Datenbank in einem geschützten Verzeichnis des Servers speichert. Bei jedem erneuten Zugriff auf das Bild wird zunächst überprüft, ob das Bild schon auf der Festplatte gespeichert wurde und im Erfolgsfall direkt auf das Bild zugegriffen. War das Bild noch nicht gespeichert, wird dies bewerkstelligt. Zum Anzeigen von in der Datenbank gespeicherten Bildern wird auf die vom Browser nicht aufrufbare `ShowDBPicAction`-Klasse zurückgegriffen. Diese dekodiert zunächst den verschlüsselt übergebenen Dateinamen und überprüft ihn auf kritische Zeichenfolgen. Anschließend wird dem Browser der mit dem adäquaten Content-Type versehene Datenstrom übermittelt. Auf die Funktionalität der `ShowDBPicAction` kann nur unter Nutzung der abgeleiteten Klassen `A_ShowDBPicAction`, `FK_ShowDBPicAction` und `PK_ShowDBPicAction` zugegriffen werden. Hier wird, analog zu den abgeleiteten `ValidationAction`-Klassen, unter Zuhilfenahme komplexer Datenbankabfragen ermittelt, ob der Nutzer das Recht besitzt, auf ein Bild zuzugreifen. Dies ist der entscheidende Unterschied gegenüber dem Hochladen eines Medium (siehe oben). Da das Medium auch in der Datenbank vorhanden ist, kann über eine Datenbankabfrage überprüft werden, ob der Nutzer die entsprechenden Rechte besitzt. Selbst wenn ein Nutzer versucht, auf ein anderes Bild zuzugreifen als vorgesehen, wird er nur Zugang zu den Bildern erhalten, für die er die Rechte hat. Auf die Implementierung einer entsprechenden Methode für den Modulkoordinator konnte verzichtet werden, da dieser keine Möglichkeit besitzt, auf bereits erstellte Prüfungsfragen in seiner Rolle als Modulkoordinator zuzugreifen. Dies ist ihm nur in seiner Rolle als an ein Fach gebundener Autor möglich.

4.15 Ordnungsfunktionalität

An mehreren Stellen im Programm erscheinen Seiten, die eine tabellarische Übersicht von Prüfungsfragen enthalten. Ein Beispiel findet sich in Abbildung 4.1 Mitte. Die Tabellen können gleichzeitig nach mehreren Kriterien sortiert werden. Dazu klickt der Nutzer zunächst auf eine Tabellenüberschrift und danach auf eine andere. Dadurch sind die Tabelleninhalte primär nach der zuletzt gewählten und sekundär nach der zuerst gewählten Tabellenüberschrift sortiert. Der Nut-

zer kann die Inhalte auf- oder absteigend sortieren. Dieses Vorgehen lässt sich bis zur maximalen Spaltenanzahl wiederholen. Die Datenbankabfrage zum Füllen der Tabelle wird nur ein einziges Mal durchgeführt. Die Daten werden in den Speicher geladen und dort sortiert. Die Sortierung läuft unter Zuhilfenahme der Java-eigenen Implementierung des Mergesort-Algorithmus. Dieser besitzt eine Worst-Case-Laufzeit von $O(nlog(n))$ und ist damit einer der effizientesten stabilen Sortieralgorithmen[6]. Bei der Umsetzung der Sortierung im EPM-System wurde auf die Reflection-Technologie zurückgegriffen. Per Reflexion oder Introspektion kann ein Programm zur Laufzeit Informationen über Klassen, Objekte, Methoden und Attribute abfragen. Auf diese Weise können verschiedene Klassen mit verschiedenen Attributen an die Sortierfunktion übergeben werden. Es musste nicht für jede Klassen/Attribut-Kombination eine gesonderte Sortierfunktion implementiert werden.

4.16 Unterbinden der doppelten Speicherung

Mittels der Navigationselemente innerhalb eines Browsers kann der Nutzer durch Drücken des Zurück-Buttons auf bereits besuchte Seiten zurück gelangen. Auch innerhalb der Seiten des EPM-Systems kann der Nutzer sich in seiner Surf-Historie zurück bewegen. Um zu verhindern, dass der Nutzer durch eine Fehlbedienung denselben Datensatz durch wiederholtes Senden desselben Formulars mehrfach speichert, wurde ein Schutzmechanismus implementiert. Dieser wurde mit Hilfe des Entwurfsmusters (vgl. Abschnitt 2.1.5 oben) Synchronizer Token realisiert. Durch den Befehl `saveToken()` wird in der Session des Nutzers ein eindeutiges Zeichen, ein Token, generiert. Gleichzeitig wird das Token innerhalb der Internetseite als verstecktes Formularfeld gespeichert. Nach dem Abschicken des Formulars wird überprüft, ob das Token valide ist. Ist dies der Fall, werden die entsprechenden Speicheroperationen ausgeführt und das Token invalidiert. Schickt der Nutzer dasselbe Formular erneut ab, liefert die Überprüfung den Wert „false“ und der Nutzer erhält eine entsprechende Fehlermeldung. Zusätzlich wird bei der

[6] Ein Sortieralgorithmus wird als stabil bezeichnet, wenn Elemente mit gleichem Sortierschlüssel nach dem Sortieren in der gleichen Reihenfolge wie vor dem Sortieren auftreten. Die erste Zwei der Zahlenfolge 2,1,2 steht bei einem stabilen Sortierverfahren nach dem Sortieren weiterhin vor der zweiten Zwei.

Wahl eines Koordinators vor dem Abspeichern überprüft, ob bereits ein Koordinator für das entsprechende Modul bzw. Fach eingesetzt wurde und gegebenenfalls eine entsprechende Fehlermeldung ausgegeben.

5 Diskussion

5.1 Vergleich mit anderen Systemen

Im Folgenden wird das EPM-System mit einigen konkurrierenden Produkten verglichen. Es werden Vor- und Nachteile dieser Systeme aufgezeigt und ein mögliches Zusammenspiel der jeweiligen Systeme diskutiert.

5.1.1 IMPP Online-Tools

Das IMPP (vgl. Abschnitt 2.2.1) stellt mit den Online-Tools internetbasierte Werkzeuge zur Erzeugung einer Klausur zur Verfügung. Mit Hilfe der Online-Tools können neue Klausuren angelegt oder bereits vorhandene bearbeitet und kopiert werden. Dabei wird auf den im IMPP vorhandenen Aufgabenpool zurückgegriffen. Der Zugriff erfolgt direkt mittels Browser. Die Tools laufen auf Servern des IMPPs.

Der Vorteil des IMPP-Programms liegt darin, dass durch den Einkauf von IMPP-Fragen nach IMPP-Normen gereviewte Fragen in Klausuren eingesetzt werden können. Die Dozenten der Medizinischen Fakultät der WWU Münster wären nicht mehr gezwungen, eigene Prüfungsfragen zu erstellen. Der aufwendige Reviewprozess, der innerhalb der Medizischen Fakultät erfolgt, könnte auf ein Minimum beim Betrachten des Gesamtkontextes einer Klausur reduziert werden. Die darauf folgende Korrektur bzw. Substitution einzelner Fragen würde entfallen, ebenso wie das Reviewsystem innerhalb des EPMs (Abschnitt 3.3).

Der größte Nachteil besteht in den Kosten, die mit dem Erwerb von Prüfungs-

fragen und der Zusammenstellung einer Klausur verbunden sind. Zudem können Forschungsschwerpunkte und Expertenmeinungen der Medizinischen Fakultät der WWU Münster oft nur schwer oder gar nicht berücksichtigt werden. Das IMPP bietet als zusätzliche Dienstleistung eine Klausurauswertung an. Diese Auswertung entspricht jedoch nicht der für die Medizinische Fakultät gültigen Prüfungsordnung inklusive Bestehensgrenze, Ankerklausel, Nachteilsausgleich und Teilleistungsnachweis (vgl. Abschnitt 2.2.3). Eine gesonderte Auswertung innerhalb der Medizinischen Fakultät ist in jedem Fall nötig.

Zusammenfassend lässt sich sagen, dass sich der IMPP-Fragenpool und das EPM-System ideal ergänzen. Der Großteil der Fragen einer Klausur wird mittels des Prüfungsmanagement-Anteils des EPM-Systems erstellt und direkt im System abgespeichert. Sind einzelne Dozenten nicht in der Lage, rechtzeitig eine genügend große Anzahl Fragen zu verfassen, kann auf den IMPP-eigenen Pool zurückgegriffen werden. Die Fragen werden anschließend mittels eines aufwendigen Verfahrens (vgl. Abschnitt 5.4) in die lokale Datenbank importiert und die Klausuren zusammengestellt. Die IMPP-Tools zum Generieren einer ganzen Klausur sind dabei nicht erforderlich. Die Auswertung der Prüfungsergebnisse erfolgt mittels des EPM-Prüfungssystems (Abschnitt 3.1). Dieser Ansatz entspricht der in Münster gängigen Praxis.

5.1.2 HIS-GX

Die Firma HIS ([28]) bietet ein breites Produktportfolio, das auf die Bedürfnisse von Hochschulen zugeschnitten ist. HIS steht für Hochschul-Informations-System GmbH. Mit dem HIS-GX Modul POS in der Version 8.0 bietet HIS eine Prüfungsverwaltung an. Dieses wird auch unter der Bezeichnung HISPOS-GX vertrieben. Das Modul ermöglicht unter anderem die Prüfungsanmeldung, die Kontrolle von Vorleistungen, die Erfassung von Prüfungsergebnissen, die automatische Berechnung von Gesamtnoten und die Aufbereitung und Auswertung der Prüfungsdaten. Auf den mit dem POS-Modul bestückten Arbeitsplatzrechnern muss Windows 2000 SP 4 oder Windows XP SP 2 installiert sein. Bei Mehrbenutzersystemen kann der Datenbankserver mit PostgreSQL in den Versionen 7.4.2 oder 8.0.3 oder mit INFORMIX in der Version 9.x ausgestattet sein. Der

Server kann in beiden Fällen unter Linux oder Windows betrieben werden. Zur Nutzung von INFORMIX wird für jeden Nutzer eine entsprechende Lizenz benötigt. Zudem muss auf jedem PC das INFORMIX-CLIENT SDK installiert sein. Bei Einplatzsystemen wird PostgreSQL in der Verison 8.0.3 unter Windows eingesetzt.

Das HIS-GX Modul QIS in Version 8.0 erweitert viele HIS-Module um eine Internet-Schnittstelle. Das Kürzel QIS steht für **Q**ualitätssteigerung der Hochschule im **I**nternet durch **S**elbstbedienung. Auf viele Services eines Moduls kann so direkt via Browser zugegriffen werden, die Installation einer eigenen Applikation entfällt. Das QIS Modul POS ist die Internet-Schnittstelle für das HIS-GX Modul POS. Es bietet z. B. für den Prüfer die Möglichkeit der Noteneingabe und des Im- und Exports von Prüfungsnoten mittels Excel. Der Student kann sich zu Prüfungen anmelden, er kann seine Noten einsehen und seine Bescheinigungen im pdf-Format herunterladen. Diese sind dabei mit einer digitalen Signatur versehen. Der Dekan erlangt durch das auch QISPOS genannte Modul Zugriff auf prüfungs- und prüferbezogene Notenansichten. Zusätzlich enthält das Programm Schnittstellen für Mitarbieter des Prüfungsamtes und für Administratoren. Als Webserver wird der Apache zum Betrieb unter Linux empfohlen. Gegebenenfalls können auch der Windows 2000 Server SP 4 oder der Windows 2003 Server eingesetzt werden. Als Applikationsserver dient der Tomcat 5. Dieser kann unter Linux oder einem Windows Server betrieben werden. Es wird auf die für das POS-Modul eingerichtete Datenbank zugegriffen.

Zur Auswertung der mittels POS erfassten und verarbeiteten Daten dient das Modul ISY in der Version 8.0. ISY ist eine Client-Server-Applikation und greift zur Datenevaluierung auf eine lokal installierte Access-Datenbank zurück. Es können Etiketten, Bescheide, Listen und Statistiken erstellt werden. Durch Access-Operationen kann das System um hochschuleigene Auswertungen ergänzt werden.

Die HIS-GX Module POS, QIS und ISY lassen sich hinsichtlich ihres Leistungsumfangs und der technischen Voraussetzungen mit dem Prüfungsanteil des EPMs (Abschnitt 3.1) vergleichen. Der Vorteil des HIS-Systems liegt in der Nutzung eines etablierten Systems, das unter anderem für die Bachelor/Master-Studien-

gänge der WWU Münster benutzt wird (vgl. [91]).

Den Vorteilen stehen eine Reihe von Nachteilen gegenüber. An der Medizinischen Fakultät der WWU Münster existiert ein Studierendenportal namens Medicampus[1]. Es ist mittels des Content-Management-Systems TYPO3 und unter Nutzung der Programmiersprache PHP realisiert. Ziel des Portals ist die Bereitstellung aller für die Studierenden der Medizin relevanten Informationen innerhalb eines Kontextes. Hier finden die Studierenden auch ihre Prüfungsergebnisse. Die Ausgaben des HIS-Systems könnten nur mit erheblichen Aufwand in dieses System integriert werden. Die Anbindung an das CADS-System (vgl. Abschnitt 2.1.9) ist ebenfalls mit Aufwand verbunden. Im HIS-System können keine Fragen gespeichert werden. Sowohl Prüfungsmanagement als auch Reviewverfahren inklusive NRW-weitem Fragepool würden bei der Nutzung von HIS entfallen oder müssten parallel betrieben werden. Das HIS-System ermöglicht zwar das Speichern einzelner Noten für Modul-Fach-Kombinationen und das Errechnen einer Gesamtnote für ein Fach, eine vollautomatische Auswertung der Prüfungsergebnisse mittels Vergleichs der Antworten der Studierenden mit den richtigen Lösungen ist jedoch nicht möglich. Die Auswertung müsste weiter über das EPM-System erfolgen und die Ergebnisse in die HIS-Datenbank importiert werden. Die Prüfungsordnung müsste zu Großteilen sowohl innerhalb des EPM-Systems als auch innerhalb des HIS-Systems gepflegt werden. Die Umsetzung und Interpretation der neuen Studienordnung erfordert weiterhin eine große Anpassungsfähigkeit des zugrunde liegenden Systems. Die EDV-technischen Veränderungen müssten in zwei Systemen bewerkstelligt werden, was eine deutliche Verminderung der Reaktionsfähigkeit und einen höheren Arbeitsaufwand bedeuten würde. Beide Systeme müssten bei jedem Release-Wechsel aufeinander abgestimmt werden. Zur Einarbeitung in das HIS-System müssten Workshops und Lehrgänge besucht werden.

Zusammenfassnd lässt sich feststellen, dass beide System synchron einsetzbar sind. Dies würde jedoch zu einem deutlichen Anstieg des Arbeitsaufwandes und der Arbeitszeit führen. Gründe hierfür sind die zum Teil doppelte Datenhaltung und die Anpassungen, die an den Standardmodulen bei jedem Versionswech-

[1] Zurzeit befindet sich eine überarbeitete Version des Portals in der Entwicklung. Diese nutzt die gleichen Technologien wie die Vorgängerversion.

sel vorgenommen werden müssten. Es müssten Schulungen bezahlt und zusätzliche Soft- und Hardware[2] angeschafft und administriert werden. Ein Großteil der vollen Funktionsvielfalt des HIS-Systems würde nicht benötigt. Die zusätzlichen Kosten zum Betrieb des HIS-Systems stehen in keinem Verhältnis zum voraussichtlichen Nutzen.

5.1.3 IMSm

Die Abkürzung IMSm steht für **I**tem **M**anagement **S**ystem **m**edicine. Dieses System lässt sich vom angesteuerten Funktionsumfang her am ehesten mit dem EPM-System vergleichen. Das System befindet sich jedoch noch in der Entwicklungsphase und wird noch nicht produktiv eingesetzt. Realisiert wird das IMSm am Kompetenzzentrum Prüfungen für die Medizin Baden-Württemberg in Kooperation mit den Medizinischen Fakultäten der Universitäten Berlin, Heidelberg und München.

Der angestrebte Funktionsumfang entspricht, außer in Detailfragen, dem Funktionsumfang der drei Teilbereiche des EPM-Systems. Aufschlüsselungen der geplanten Features finden sich unter [35] und [36].

Da keine öffentlich zugänglichen Dokumente darüber existieren, welche Features bereits in welcher Weise umgesetzt wurden, muss ein weitergehender Vergleich an dieser Stelle entfallen. Das Entwicklung des IMSms zeigt jedoch die Aktualität des Themas.

5.1.4 Weitere Systeme

Neben den hier besprochenen Systemen existieren noch eine Reihe von Eigenentwicklungen, die fakultätsintern genutzt werden. Beispielhaft genannt sei hier

[2] Die Firma HIS bietet den Hochschulen unter dem Namen HISPRO einen Service an, der es den Hochschulen erlaubt, über das Internet auf HIS-Module auf einem von HIS administrierten Server zuzugreifen. Es wäre zu entscheiden, welche Alternative der Hochschule auf Dauer weniger Kosten verursachen würde und ob eine Auslagerung aufgrund der individuellen Anpassungen an die Bedürfnisse der Medizinischen Fakultät der WWU Münster überhaupt sinnvoll ist.

das an der Charité Berlin eingesetzte System zur statistischen Auswertung von Prüfungen, das aufgrund jahrelangen Einsatzes als eines der ausgereiftetsten betrachtet werden kann.

Wegen fehlender Informationen kann auf diese Systeme nicht näher eingegangen werden.

Es existieren auch verhältnismäßig kleine Programme, deren Leistungsumfang nur einen entsprechend kleinen Teil des EPMs abdeckt. Beispielhaft seien hier die Programme Klaus und Examenis genannt. Klaus ist ein Java-Programm der Firma Blubbsoft GbR ([6]) und unterstützt den Nutzer bei der Durchführung papierbasierter Klausuren durch Klausurentwurf, Einlesen der Ergebnisse via Scanner und die anschließende Rohauswertung der Prüfungsergebnisse. Das System Examenis der Creative Education und Consulting GmbH ([19]) hat einen ähnlichen Funktionsumfang. Neben der papierbasierten Klausurabsolvierung bietet das System auch die Möglichkeit einer webbasierten Prüfungsausführung. Das Programm ist mittels Active Server Pages-Technologie (ASP-Technologie[3]) implementiert.

5.2 Konfrontation mit den Anforderungen

Die im Abschnitt 3.2 aufgezählten Anforderungen an das Prüfungsmanagement wurden in vollem Umfang implementiert. Im Kapitel 3.2 wird an den entsprechenden Stellen auf die Anforderungen verwiesen. Die nur angedachten Workflows wurden verfeinert und umgesetzt.

Der generelle Workflow für den Reviewprozess wurde ausgearbeitet (vgl. Abschnitt 3.3). Dieser wurde bisher nicht implementiert. Ursache ist, dass zunächst, entgegen der ursprünglichen Planung, die gesamte Datenbankstruktur des Prüfungssystems (vgl. Abschnitt 3.1) und darauf aufsetzende Programme entwickelt werden mussten. Zusätzlich wurden Daten (ein-)gepflegt und der Support für das Studiendekanat geleistet. Erst nach der Schaffung einer weiteren Arbeitsstelle, die diesen Tätigkeitsbereich übernahm, konnte mit der Planung und Implemen-

[3]Die ASP-Technologie wird seit 2002 nicht mehr weiterentwickelt und wurde von Microsofts Nachfolgeprodukt ASP.NET abgelöst.

tierung des Prüfungsmanagementsystems begonnen werden. Insgesamt nahm das Prüfungssystem etwa 60 Prozent der für das Projekt zur Verfügung stehenden Zeit in Anspruch.

5.3 Probleme und Herausforderungen

Bei Design und Implementierung des EPM-Systems waren eine Reihe von Problemen zu lösen.

Die Anbindung des CADS-Systems stellt einen vermehrten Arbeitsaufwand gegenüber der Nutzung eines eigenen Authentifizierungsmechanismusses dar. Daten, die normalerweise über eine einzelne Datenbankabfrage erhältlich sind, müssen durch zwei separate SQL-Queries erfragt werden, zum einen auf der EPM-, zum anderen auf der CADS-Datenbank.

Es entstand das Problem, dass temporär keine Verbindung zum CADS-System aufgebaut werden konnte. Zu Hochzeiten führt die Hälfte der Anfragen auf Client-Seite zu einem Connection-Timeout. Dieses Problem machte die Entwicklung zuweilen nahezu unmöglich, da auf jeder Seite des Prüfungsmanagement-Systems mindestens eine CADS-Abfrage gestartet wird. Zusammen mit den Administratoren wurden verschiedene potentielle Fehlerursachen extrahiert, getestet und verworfen. Zu diesem Zweck wurden eigens Testprogramme in PHP und Java geschrieben und mit unterschiedlichen IP-Nummern, Rechnern, Netzwerken und Anschlussbuchsen ausgeführt. Der anfängliche Verdacht, es handele sich um ein Java-spezifisches Problem, erhärtete sich nicht. Auch ein Versagen des eingesetzten Laptops konnte ausgeschlossen werden. Nach fünf Monaten wurde das Problem gelöst. Der Timeout entstand nur bei dem kombinierten Einsatz einer Entwicklungsumgebung unter Windows und innerhalb eines speziellen Netzwerkes. Die Ursache lag in der fehlerhaften Konfiguration einer externen Firewall.

Bei der Nutzung neuer Technologien kam es zu Schwierigkeiten. Beispielsweise funktionierte die Nutzung der Struts-EL-Tags (vgl. Abschnitt 2.1.6) nicht immer; in diesem Fall wurde auf die nativen Struts-Tags ausgewichen.

Die Dokumentation einiger freier Bibliotheken stellte sich als unzureichend heraus. In diesen Fällen mussten die gewünschten Funktionalitäten durch Internet-Recherche, das Durchsuchen des Quelltextes und Trial & Error in Erfahrung gebracht werden. Mangelhafte Dokumentation und Stringenz waren auch bei der Implementierung des Excel-Imports die Hauptprobleme (vgl. Abschnitt 4.7.2).

Die Entwicklung einer möglichst einfachen Bedienoberfläche (vgl. Anforderung 3.2.9 b) erwies sich aufgrund der Komplexität des Systems als Herausforderung. Es wurde berücksichtigt, welche Daten und Hilfestellungen dem Nutzer an welcher Stelle des Systems die Arbeit erleichtern.

Vor allem bei Entwurf und Implementierung des Prüfungssystems war hoher Zeitdruck gegeben, da Deadlines, etwa zur Prüfungsanmeldung oder zur Auswertung der Prüfungsfragen, gehalten werden mussten.

Die durch die neue Approbations- bzw. Studienordnung entstandenen Anforderungen führten zu einer komplexen Datenbankstruktur (vgl. die Datenbankübersicht im Anhang). Der Komplexitätsgrad wurde zusätzlich durch die Tatsache erhöht, dass die Datenbank den Anforderungen aller drei Teilbereiche des EPM-Systems gerecht werden musste. Bei der Struktur der Datenbank musste die Balance zwischen Effizienz und Flexibilität gehalten werden.

Aufgrund der Aktualität bei der Umsetzung der neuen Studienordnung kam es, gerade zu Beginn des Projekts, zu einer Fülle neuer Anforderungen. Diese führten dazu, dass Datenbankänderungen vorgenommen werden mussten, die auch eine Anpassung aller aufsetzenden Programme nach sich zog. Eine iterative Softwareentwicklung, etwa nach dem Wasserfall-Modell (vgl. [87]), war nicht möglich.

Aufgrund dieser Erfahrungen wurde die Datenbank in der Folgezeit flexibler gestaltet, um auf zusätzliche Anforderungen besser reagieren zu können. Es ist zukünftig beispielsweise möglich, Prüfungsfragen beliebigen Typs in der Datenbank abzulegen. Momentan sind nur 1-aus-5-MC-Fragen in Gebrauch. Das Rollensystem wurde so gestaltet, dass problemlos neue Rollen mit differenzierten Rechten auf der Ebene von Universitäten, Fächern, Universität/Semster/Modul- und U/S/M/F-Kombinationen angelegt werden können. Potentiell können einer Frage nicht nur Bilder, sondern Medien beliebigen Typs zugeordnet werden. Die Forde-

rung, dass mehreren Fragen derselbe Fragestamm zugeordnet ist oder, dass sich Fragen dieselben Medien teilen können, entstand erst im Laufe der Entwicklung und bedurfte keiner Modifikation der Datenbank. Die Anforderung nach einer individualisierten Klausur, in der die Fragen für jeden Studierenden in einer anderen Reihenfolge erscheinen, konnte durch ein zusätzliches Attribut innerhalb einer Tabelle realisiert werden, ebenso wie die Forderung nach einer Beschriftung der Medien. Lediglich die Implementierung der Beziehung zwischen Leistungsnachweisen und Teilleistungsnachweisen als m:n-Relation erwies sich als unnötig. Hier reicht eine 1:n-Relation.

Um den genannten Anforderungen gerecht zu werden, mussten nicht nur sehr komplexe Datenbankabfragen implementiert werden, sondern an vielen Stellen im Programm war auch ein Setzen bzw. Löschen von Session-Attributen nötig. Insbesondere die große Anzahl von Fallunterscheidungen bei den Editierungsdialogen stellte eine große Herausforderung dar. Diese Komplexität machte eine ausführliches Testen unumgänglich.

5.4 Einsatz und Nutzen des Systems

Das Prüfungssystem (vgl. Abscnitt 3.1) befindet sich seit Januar 2005 im Einsatz. Durch dieses ist es möglich, die Prüfungsanmeldungen der Studierenden gezielt zu steuern, semesterweite Klausuren zu generieren, die Ergebnisse zu importieren und sowohl Studierenden als auch Studiendekanat differenzierte Übersichten über die Studienleistungen im klinischen Abschnitt des Medizinstudiums zu bieten. Aufgrund der individuellen Anforderungen, die mit der Umsetzung der neuen Studienordnung der Medizinischen Fakultät der WWU Münster einhergehen, wäre die Anpassung bereits existenter Systeme mit einem signifikant höherem Arbeits- und Zeitaufwand verbunden.

Der Teilbereich des Prüfungsmanagements wird zurzeit noch nicht in vollem Funktionsumfang eingesetzt. Bezüglich des Gebrauchs der Erstellungs- und Editierungsdialoge des Prüfungskoordinators kann jedoch ein erstes Resumé gezogen werden. Diese Werkzeuge ermöglichen der zuständigen Abteilung, das Einlesen des Prüfungsfrageninhalts nicht mehr mit Hilfe eines aufwendigen Verfahrens

bewerkstelligen zu müssen. In einem ersten Schritt wurden die vom Studiendekanat zur Verfügung gestellten Excel-Dateien reformatiert. Dann wurde ein Visual Basic-Skript ausgeführt, um Formatierungen und Sonderzeichen innerhalb der Excel-Dateien zu ersetzen. Im nächsten Schritt wurden die Dateien mit einer Spezialsoftware in eine äquivalente MySQL-Datenbank transformiert. Im finalen Schritt wurden mittels eines PHP-Skriptes die so generierten Datenbankinhalte in die reguläre Prüfungsdatenbank importiert. Der gesamte Prozess nahm für die Prüfungsfragen eines Semesters etwa 100 Zeitstunden in Anspruch. Aufgrund der Komplexität des Vorgangs und der Tatsache, dass einige Operationen direkt auf der Produktivdatenbank vorgenommen wurden, wurde diese Arbeit bisher von einem in Vollzeit angestellten Mitarbeiter geleistet. Im SS 06 konnte diese Arbeit durch das Prüfungsmanagement-System komplett von einer studentischen Hilfskraft übernommen werden. Der Aufwand für diese Arbeit betrug ca. 90 Zeitstunden. Der nur geringe Rückgang der Arbeitszeit erklärt sich dadurch, dass nur ein sehr kleiner Teil der Fragen (174 von 1222) mittels des Hochladens der Excel-Dateien bewerkstelligt werden konnte, da bereits leere „Fragenhülsen“ im System hinterlegt waren, die zwar die Fragen und die richtige Lösung enthielten, jedoch keinerlei Inhalt wie Text oder Bilder. Diese Fragenhülsen mussten im Editierungsdialog mittels Copy & Paste gefüllt werden. Wenn im nächsten Semester alle Prüfungsfragen mittels des Excel-Imports ins System integriert werden, ist mit einem Zeitaufwand von 15 bis 20 Zeitstunden zu rechnen. Es ergäbe sich mindestens eine Reduktion um den Faktor fünf. Das Tool zum Modifizieren der Prüfungsfragen wurde temporär so abgeändert, dass auch Frageninhalte vergangener Semester nachgepflegt werden können. Mit Stand zum 12.10.2006 wurden so insgesamt 1248 Prüfungsfrageninhalte eingefügt.

Zusammenfassend lässt sich sagen, dass ohne das Prüfungssystem die Realisierung der neuen Studienordnung kaum möglich gewesen wäre. Der administrative Aufwand hätte vom Studiendekanat nur schwer bewältigt werden können. Die Entscheidung, bei der Realisierung der Prüfungssystems auf eine Eigenentwicklung zu setzen statt auf Fremdsysteme zurückzugreifen, hat sich als gute Lösung hinsichtlich Flexibilität, Wartbarkeit und Effizienz erwiesen. Alle anderen, oben diskutierten Ansätze wären kaum oder nur mit erhöhtem Aufwand gangbar gewesen.

Das Prüfungsmanagement stellt den nächsten Schritt dar, das Arbeitsvolumen bei der Zusammenstellung einer semesterweiten Klausur zu minimieren. Durch den Einsatz der Werkzeuge zur Fragenerstellung und -editierung für den Prüfungskoordinator wurde bereits eine Arbeitserleichterung erreicht. Der Einsatz des Gesamtsystems wird voraussichtlich auf Seiten des Studiendekanats zu einer weiteren Verringerung des administrativen Aufwands führen.

Die zu Beginn der Entwicklung an das Prüfungssystem und das Prüfungsmanagement gestellten Anforderungen wurden realisiert. Durch die Aktualität der Studienordnung kam es im Verlaufe der Entwicklung zu einer Anzahl zusätzlicher Anforderungen, die nicht alle umgesetzt werden konnten. Hier besteht Raum für eine Weiterentwicklung des EPM-Systems.

5.5 Ausblick

Im WS 06/07 werden die Prüfungsfragen mit den Erstellungs- und Editierungstools des Prüfungskoordinators eingelesen werden. Hierzu werden mehrere studentische Hilfskräfte eingesetzt werden.

Das Prüfungsmanagement wird in seiner Gesamtheit in Produktion gehen, sobald ein Mitarbeiter für Folgeaufgaben geschult ist. Neben der Betreuung der laufenden Anwendung können Verbesserungsvorschläge evaluiert und das System um zusätzliche Funktionalitäten erweitert werden.

Momentan wird ein Programm entwickelt, mit dessen Hilfe eine Klausur aus den im EPM-System vorhandenen Fragen zusammengestellt werden kann. Die Klausurerstellung erfolgte bisher durch direkte Manipulationen der EPM-Datenbank. Das Programm wird zunächst nur die Fragen des aktuellen Semesters zur Auswahl stellen. In einem zweiten Schritt werden auch Fragen vergangener Semester berücksichtigt. Es wird unterschieden zwischen der Klausurerstellung für Erstschreiber und der für Wiederholungsklausuren. Im ersten Fall orientiert sich die Klausurzusammenstellung anhand der entsprechenden U/S/M/F-Kombinationen, im zweiten Fall anhand der zugrundeliegenden Leistungsnachweise.

Es ist vorgesehen, die Medizinprüfungen in Zukunft vollautomatisch unter Nutzung entsprechender Software zur Absolvierung der Prüfungen durchzuführen. Bisher werden die Klausuren papierbasiert geschrieben, die Ergebnisse mittels Scanners eingelesen und dann in die EPM-Datenbank importiert. Als gangbare Alternativen stehen z. B. LPLUS ([42]), der CAMPUS-Prüfungsplayer des Universitätsklinikums Heidelberg ([84]) oder das weiter oben erwähnte Examenis zur Auswahl. Somit könnte in Zukunft der gesamte Prozess zur Erstellung einer Klausur, vom Eintreiben der Fragen, ihrer Zusammenstellung, dem Absolvieren der Prüfung bis hin zur Auswertung der Ergebnisse nahezu vollautomatisch erfolgen. Die Arbeitsbelastung würde so voraussichtlich sowohl für das Prüfungsdekanat als auch für die IT-Seite deutlich gesenkt werden. Zudem entfallen die Druckkosten.

Es ist geplant, für das CADS-System eine LDAP-Schnittstelle[4] zu implementieren. Unter Nutzung dieser Schnittstelle wäre es möglich, dass sich die Studierenden an den Rechnern, an denen sie zukünftig die elektronischen Klausuren absolvieren werden, mit Hilfe ihrer CADS-Zugangsdaten anmelden können. Zu Prüfungsanmeldung, Absolvierung der Klausuren und Betrachten der Ergebnisse bräuchten die Studierenden nur eine einzige Nutzername/Passwort-Kombination.

Es ist denkbar, eine Webapplikation zu schreiben, mit deren Hilfe die Studierenden Übungsklausuren durchführen können. Hierzu können mittels des Prüfungsmanagementteils extra Prüfungsfragen erstellt werden. Alternativ wird auf Altfragen zurückgegriffen.

Das Akzento-System (vgl. Abschnitt 2.1.9) wird Studierenden der Medizin an der WWU die Möglichkeit geben, sich für Seminare und Vorlesungen anzumelden. Die Anmeldung zu einzelnen Veranstaltungen könnte mit einer Anmeldung zu der entsprechenden Prüfung gekoppelt werden.

Es besteht die Möglichkeit, die Notenabfrage via Telefon bzw. über ein WAP-fähiges Handy zu ermöglichen, wie dies beim HIS-GX-System (vgl. Abschnitt 5.1.2) durch Einbindung der Module QIS TEL und QIS WAP möglich ist. Auch

[4] LDAP steht für **L**ightweight **D**irectory **A**ccess **P**rotocol und dient zur Abfrage und Modifikation von Informationen eines Verzeichnisdienstes. Einzelheiten finden sich unter [56].

eine automatische Benachrichtigung via SMS ist vorstellbar.

Die Realisierung des Reviewsystems (vgl. Abschnitt 3.3) würde zu einem stetig anwachsenden Pool qualitativ hochwertiger Prüfungsfragen führen. Bei der Zusammenstellung einer Klausur könnte zum Teil auf Altfragen zugegriffen oder es könnten neue Fragen von Vaterfragen abgeleitet werden. Die Qualität der Fragen würde zu einer Qualitätssteigerung der Klausuren beitragen und den Studierenden als optimale Vorbereitung zur Ärztlichen Prüfung (Staatsexamen) dienen.

Durch eine Implementierung der vom IMS Global Learning Consortium entworfenen Spezifikation Question & Test Interoperability ([32]) kann ein internationaler Standard zum Austausch von Prüfungsfragen auf Basis von XML realisiert werden. Dies würde beispielsweise den Austausch mit dem IMSm oder mit anderen Prüfungsdatenbanken erlauben. Denkbar wäre auch der Import von EPM-Prüfungsfragen in das TED-System der Medizinischen Fakultät Münster. Die Studierenden könnten in einer Veranstaltung darüber entscheiden, welche Antwort ihrer Meinung nach die richtige ist. Dies ermöglicht Lehrenden und Lernenden eine Einschätzung ihres momentanen Wissenstandes und führt zu einer Interaktion und Bereicherung des Lehrgeschehens.

Auf einem Workshop, zu dem Angehörige der Medizinischen Fakultäten NRWs geladen waren, wurden das Prüfungssystem und das Prüfungsmanagement vorgestellt. Vertreter der Medizinischen Fakultäten Bochum, Köln und Düsseldorf zeigten großes Interesse bezüglich eines zukünftigen Einsatzes an ihren Fakultäten. Es stehen zwei gangbare Alternativen zur Auswahl. Zum einen können die Universitäten die Hard- und Software der Medizinischen Fakultät Münsters nutzen. Die Prüfungsfragen und -ergebnisse würden dann in einer gemeinsamen Datenbank gespeichert. Datenbank und Informationssystem sind so entworfen, dass dies ohne Probleme möglich ist. Die zweite Möglichkeit besteht darin, die EPM-Software auf einem eigenen Server zu installieren und zu betreiben. Die CADS-Funktionalitäten wurden bei Entwurf und Implementierung des EPM-Systems derart gekapselt, dass die entsprechenden Funktionalitäten leicht an ein vor Ort vorhandenes Authentifizierungssystem angepasst werden können.

Um die Installation und Administration einer eigenen EPM-Version zu erleichtern, könnten die Werkzeuge des Prüfungssystems von PHP/TYPO3 nach Java

EE portiert werden.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Apache Software Foundation (2006) Logging Services, Log4J. Internet: `http://logging.apache.org/log4j/docs/`
(Zugriff 05.10.2006 07:34 MEZ)

[2] Armstrong E, Ball J, Bodoff S, Carson DB, Evans I, Green D, Haase K, Jendrock E (2006) The J2EE 1.4 Tutorial, For Sun Java System Application Server Platform Edition 8.2, Sun Microsystems Inc., Santa Clara

[3] Ball J, Carson DB, Evans I, Haase K, Jendrock E (2006) The Java EE 5 Tutorial, For Sun Java System Application Server Platform Edition 9, Sun Microsystems Inc., Santa Clara

[4] Bergsten H (2004) JavaServer Faces, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo

[5] BGBL (Bundesgesetzblatt) (2002) Approbationsordnung für Ärzte, ÄAppO 2002.
Internet: `http://www.bmg.bund.de` (Zugriff 05.10.2006 07:34 MEZ)

[6] Blubbsoft GbR (2006) Klaus – Klausurenkorrektur in der Hälfte der Zeit. Internet: `http://www.klausuren-klaus.de/`
(Zugriff 05.10.2006 07:34 MEZ)

[7] Böhm O (2006) Aspektorientierte Prgrammierung mit AspectJ 5, Einsteigen in AspectJ und AOP, dpunkt.verlag, Heidelberg

[8] Buchmann A (2004) JavaScript interaktiv, Omnigena Verlags GmbH, Düsseldorf

[9] Bundesamt für Sicherheit und Informationstechnik (2006) Gefahren und Risiken im Umgang mit JavaScript/JScript. Internet: http://www.bsi.de/fachthem/sinet/gefahr/aktiveinhalte/definitionen/javascriptgefahren.htm (Zugriff 05.10.2006 07:35 MEZ)

[10] Burke B (2006) Enterprise JavaBeans 3.0, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo

[11] Butz T (2006) Vergleich von MySQL mit anderen DBs. Internet: http://www.tbee.de/mysql/t5_mysql_vergleich.php (Zugriff 05.10.2006 07:35 MEZ)

[12] Campione M, Walrath K, Huml A (2001) The Java Tutorial, Third Edition, A Short Course on the Basics, Sun Microsystems Inc., Santa Clara

[13] Cerami E (2005) XML for Bioinformatics, Springer, New York

[14] Dumke R (2005) Programmiersprachkonzepte, 4. Lehrhilfe zur Vorlesung: Aspektorientierte Programmierung mittels AspectJ. Internet: http://ivs.cs.uni-magdeburg.de/$\sim$dumke/PSK/AOP2.html (Zugriff 05.10.2006 07:35 MEZ)

[15] ecma INTERNATIONAL (2006) TC45 - Office Open XML Formats. Internet: http://www.ecma-international.org/memento/TC45-M.htm (Zugriff 05.10.2006 07:36 MEZ)

[16] Eickstädt D, Reuhl T (2004) J2EE mit Struts & Co., Java–Projekte mit Struts, Tomcat, Jboss und Eclipse, Markt+Technik, München

[17] Elliotte RH (2004) XML, Das mitp–Standardwerk zur professionellen Programmierung mit XML, mitp, Bonn, 2. Aufl.

[18] Elliotte RH, Means WS (2005) XML in a Nutshell, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo, 3. Aufl.

[19] Examenis (2006) Examenis das Autoren– und Testsystem.
Internet: http://www.examenis.de/examenis/
(Zugriff 05.10.2006 07:36 MEZ)

[20] Fielding R, Irvine UC, Gettys J, Compaq/W3C, Modul J, Compaq, Frystyk H, W3C/MIT, Masinter L, Xerox, Leach P, Microsoft, Berners–Lee T., W3C/MIT (1999) Hypertext Transfer Protocol —- HTTP/1.1. Internet: http://www.ietf.org/rfc/rfc2616.txt
(Zugriff 05.10.2006 07:37 MEZ)

[21] Fischer M, Vergleich JSF vs. Struts – Features der beiden Webframeworks gegenübergestellt. Internet: http://www.doubleslash.de/de/Download/AKT_javaserverfaces_vs_struts.pdf (Zugriff 05.10.2006 07:37 MEZ)

[22] Freed N, Innosoft, Borenstein N, First Virtual (1996) Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. Internet: http://www.ietf.org/rfc/rfc2045.txt
(Zugriff 05.10.2006 07:38 MEZ)

[23] Gamma E, Helm R, Johnson R, Vlissides J (2001) Entwurfmuster, Elemente wiederverwendbarer objektorientierter Software, Addison–Wesley, München Boston San Franzisko Harlow Don Mills Sydney Mexiko Stadt Madrid Amsterdam, 5., korrigierter Nachdruck

[24] Haiges S, Bien A, May M, Woehrlin B (2005) Struts, Java Framework für Webanwendungen, Software & Support Verlag GmbH, Frankfurt, 2. überarbeitete Aufl.

[25] Hall M (2006) Apache Jakarta Struts 1.2 Tutorial and Training Materials. Internet: http://courses.coreservlets.com/Course-Materials/struts.html (Zugriff 05.10.2006 07:38 MEZ)

[26] Hennebrüder S (2006) LaLiLuna – Some nice tutorials, LaLiLuna – Tutorials for struts, ejb, myeclipse, EJB Tutorials, Struts Tutorials, Hibernate Tutorials and some others.... Internet: http://www.laliluna.de/
(Zugriff 05.10.2006 07:39 MEZ)

[27] Hillyer M (2006) VBMySQL.com – Users are Evil (or, How to Protect Yourself From SQL Injection).
Internet: http://www.vbmysql.com/articles/security/users-are-evil-or-how-to-protect-yourself-from-sql-injection/
(Zugriff 05.10.2006 07:40 MEZ)

[28] HIS (2006) HIS Hoschul Informations System GmbH.
Internet: http://www.his.de/ (Zugriff 05.10.2006 07:41 MEZ)

[29] Horn T (2006) SQL–Grundlagen. Internet: http://www.torsten-horn.de/techdocs/sql.htm$#$Vergleich-MySQL-PostgreSQL-MaxDB
(Zugriff 05.10.2006 07:42 MEZ)

[30] Husted T, Dumoulin C, Franciscus G, Winterfeldt D (2003) Struts in Action, Building web applications with the leading Java framework, Manning Publications Co., Greenwich

[31] IMPP (2006) Institut für medizinische und pharmazeutische Prüfungsfragen, Rechtsfähige Anstalt des öffentlichen Rechts.
Internet: http://www.impp.de/ (Zugriff 05.10.2006 07:42 MEZ)

[32] IMS Global Learning Consortium, Inc. (2006) IMS Global Learning Consortium: IMS Question & Test Interoperability Specification. Internet: http://www.imsglobal.org/question/index.html
(Zugriff 05.10.2006 07:43 MEZ)

[33] Killersreiter B (2004) themenorientierte Semesterabschlussprüfung, Leitfaden zur Entwicklung von MC Fragen für Dozentinnen und Dozenten der Medizinischen Fakultät Münster. Internet: http://campus.uni-muenster.de/lehren/semesterabschlusspruefung.pdf
(Zugriff 05.10.2006 07:44 MEZ)

[34] Kofler M (2005) MySQL 5, Einführung, Programmierung, Referenz, Addison–Wesley, München Boston San Franzisko Harlow Don Mills Sydney Mexiko Stadt Madrid Amsterdam, 3. Aufl.

[35] Kompetenzzentrum Prüfungen für die Medizin Baden–Württemberg

(2006) IMS medicine, Item Management System medicine.
Internet: http://www.ims-m.org (Zugriff 05.10.2006 07:44 MEZ)

[36] Kompetenzzentrum Prüfungen für die Medizin Baden–Württemberg (2006) IMS medizin, Item Management System medizin, System zur Verwaltung, zum Austausch und zur Qualitätssicherung von Prüfungsinhalten. Internet: http://www.klinikum.uni-heidelberg.de/fileadmin/kompzent/flyer_k02.pdf (Zugriff 05.10.2006 07:45 MEZ)

[37] Laddad R (2006) I want my AOP!, Part 1, Separate software concerns with aspect-oriented programming. Internet: http://www.javaworld.com/\javaworld/jw-01-2002/jw-0118-aspect.html
(Zugriff 05.10.2006 07:47 MEZ)

[38] Landtag NRW (2003) Gesetz zur Gleichstellung von Menschen mit Behinderung und zur Änderung anderer Gesetze. Internet: http://www.kmdb.de/pdf/LGG-NW_2003.pdf (Zugriff 05.10.2006 07:47 MEZ)

[39] Letzel S (2003) Lernzielkatalog Arbeitsmedizin. Internet: http://www-dgaum.med.uni-rostock.de/PDF/lernzielkatalog.pdf
(Zugriff 05.10.2006 07:48 MEZ)

[40] Lier S (2005) MEQ. Internet: http://notesweb.uni-wh.de/wg/medi/wgmedi.nsf/ContentByKey/SLIR-5MXH4T-DE-p
(Zugriff 13.04.2006 17:14 MEZ)

[41] Lorenz A, Pilgrim J von, Six HW (2005) Einleitung und Basiskonzepte. In: Lorenz A, Pilgrim J von, Six HW (Hrsg) Methodische Entwicklung von Benutzungsschnittstellen für Webapplikationen. Fernuniversität in Hagen, Hagen, Kurseinheit 1, 5. Aufl.

[42] LPLUS GmbH (2006) LPLUS, Die intelligente Art des Testens.
Internet: http://www.lplus.de/ (Zugriff 05.10.2006 07:50 MEZ)

[43] McLaughlin B, Flanagan D (2004) Java 1.5 Tiger, A Developer´s Notebook, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo

[44] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2005) Ankerklausel.
Internet: https://medicampus.uni-muenster.de/anklerklausel.html
(Zugriff 05.10.2006 07:52 MEZ)

[45] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2003) Bestehensgrenzen und Notenvergabe. Internet: http://campus.uni-muenster.de/studieren/studiengang/pruefungen/bestehensgrenzen.html (Zugriff 05.10.2006 07:52 MEZ)

[46] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2003) Das Konzept der Wiederholungsprüfungen. Internet: http://campus.uni-muenster.de/studieren/studiengang/pruefungen/wiederholungen.html (Zugriff 05.10.2006 07:52 MEZ)

[47] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2004) Entwurf eines Lernzielkatalogs für die Medizinische Fakultät der Westf. Wilhelms – Universität Münster nach einer Vorage des Fachbereiches Medizin der Universität Hamburg, Stand 01.02.2004. Internet: http://campus.uni-muenster.de/download/Lernzielkatalog_MS_05.pdf
(Zugriff 05.10.2006 07:53 MEZ)

[48] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2004) Erklärung der Ebenen und Buchstabenbezeichnung für die fachbezogenen Lernziele. Internet: https://medicampus.uni-muenster.de/fileadmin/medicampus/dekanat/pdf/prolog_lernziele.pdf
(Zugriff 05.10.2006 07:55 MEZ)

[49] Medizinische Fakultät der Westfälischen Wilhelms–Universität Münster (2005) Studienordnung für den Studiengang Medizin an der Medizinischen Fakultät der Westfälischen Wilhelms–Universität Münster mit dem Abschluss der „Ärztlichen Prüfung“ (Staatsexamen) vom 24. Oktober 2005. Internet: http://medicampus.uni-muenster.de/fileadmin/medicampus/dekanat/pdf/gesetze/studienordnung.pdf
(Zugriff 05.10.2006 07:55 MEZ)

[50] Microsoft (2006) Office 2003: XML Reference Schemas. Internet:

http://www.microsoft.com/downloads/details.aspx?familyid=fe118952-3547-420a-a412-00a2662442d9&displaylang=en (Zugriff 05.10.2006 07:56 MEZ)

[51] Microsoft (2006) Scripting. Internet: http://www.microsoft.com/jscript/ (Zugriff 05.10.2006 07:58 MEZ)

[52] Monson–Haefel R (2004) Enterprise JavaBeans, Developing Enterprise Java Components, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo, 4. Aufl.

[53] Mozilla.org (2006) JavaScript. Internet: http://www.mozilla.org/js/ (Zugriff 05.10.2006 07:59 MEZ)

[54] Münz S (2005) SELFHTML 8.1.1, HTML–Dateien selbst erstellen. Internet: http://de.selfhtml.org/ (Zugriff 05.10.2006 08:07 MEZ)

[55] National Institutes of Health (2006) ImageJ, Image Processing and Analysis in Java. Internet: http://rsb.info.nih.gov/ij/ (Zugriff 05.10.2006 08:07 MEZ)

[56] Network Working Group (2006) Lightweight Directory Access Protocol (LDAP): The Protocol. Internet: http://www.ietf.org/rfc/rfc4511.txt (Zugriff 05.10.2006 08:08 MEZ)

[57] OASIS (2005) Open Document Format for Office Applications (OpenDocument) v1.0, OASIS standard, 1 May 2005. Internet: http://www.oasis-open.org/committees/download.php/12572/OpenDocument-v1.0-os.pdf (Zugriff 05.10.2006 08:09 MEZ)

[58] O´Brien TM (2005) Jakarta Commons Cookbook, O´Reilly, Bejing Cambridge Farnham Köln Paris Sebastopol Taipei Tokyo

[59] Oracle Corporation (2006) JSF Resources. Internet: http://www.oracle.com/technology/tech/java/jsf.html (Zugriff 05.10.2006 08:13 MEZ)

[60] Oracle Coporation (2006) Oracle Database 10g Express Edition FAQ.

Internet: http://www.oracle.com/technology/products/database/xe/pdf/dbxe_faq.pdf (Zugriff 05.10.2006 08:13 MEZ)

[61] Ray ET (2004) Einführung in XML, O´Reilly, Beijing Cambridge Fanham Köln Paris Sebastopol Taipei Tokyo, 2. Aufl.

[62] Samaschke K (2005) Java zu J2SE 5, Einstieg für Anspruchsvolle, Addison–Wesley in Kooperation mit Pearson Studium, München

[63] Schmidt M (2005) ImageInfo – Java class to extract image file properties. Internet: http://schmidt.devlib.org/image-info/
(Zugriff 05.10.2006 08:14 MEZ)

[64] Sharma, A (2004) Prevent a cross–site scripting attack, First step: recognise the signs and halt an XSS intrusion. Internet: http://www-128.ibm.com/developerworks/library/wa-secxss/?ca=dgr-lnxw914PreventXSS (Zugriff 05.10.2006 08:15 MEZ)

[65] Silberschatz A, Korth H F, Sudarshan S (2002) Database System Concepts, Mc Graw Hill, Boston

[66] Singer L (2004) Kurz & Gut, Model–View–Controller.
Internet: http://www.se.uni-hannover.de/documents/kurz-und-gut/ws2004-seminar-entwurf/mvc_lsinger.pdf
(Zugriff 05.10.2006 08:15 MEZ)

[67] Stark T (2005) J2EE, Einstieg für Anspruchsvolle, Addison–Wesley in Kooperation mit Pearson Studium, München

[68] Steyer R (2005) Das JavaScript Codebook, Addison–Wesley, München Boston San Franzisko Harlow Don Mills Sydney Mexiko Stadt Madrid Amsterdam

[69] Steyer R (2003) JavaScript in 21 Tagen, Markt+Technik, München

[70] Sourceforge (2006) Cewolf. Internet: http://cewolf.sourceforge.net/new/index.html (Zugriff 05.10.2006 08:16 MEZ)

[71] Sourceforge (2006) dom4J. Internet: http://www.dom4j.org/ (Zugriff 05.10.2006 08:16 MEZ)

[72] Sourceforge (2006) Java Excel API – A Java API to read, write and modify Excel spreadsheets. Internet: http://jexcelapi.sourceforge.net/ (Zugriff 05.10.2006 08:17 MEZ)

[73] Sun Developer Network (2006) Java Technology. Internet: http://java.sun.com/ (Zugriff 05.10.2006 08:17 MEZ)

[74] Taboada P G (2005) Von Sackgassen und Neubaustraßen. Javamagazin 10|2005: 28–33

[75] Telekom – Presse (2006) Oracle Database XE verfügbar. Internet: http://www.telekom-presse.at/channel_software/news_22748.html (Zugriff 05.10.2006 08:17 MEZ)

[76] The Apache Jakarta Project (2006) Jakarta Commons HttpClient. Internet: http://jakarta.apache.org/commons/httpclient/ (Zugriff 05.10.2006 08:18 MEZ)

[77] The Apache Jakarta Project (2006) Jakarta POI – Java API To Access Microsoft Format Files. Internet: http://jakarta.apache.org/poi/ (Zugriff 05.10.2006 08:19 MEZ)

[78] The Apache Struts Project (2006) Struts. Internet: http://struts.apache.org/

[79] The Apache MyFaces Project (2006) The Apache MyFaces Project. Internet: http://myfaces.apache.org/ (Zugriff 05.10.2006 08:21 MEZ)

[80] Turau V, Saleck K, Lenz C (2004) Web–basierte Anwendungen entwickeln mit JSP 2, Einsatz von JSTL, Struts, JSF, JDBC, JDO, JCA, dpunkt.verlag, Heidelberg, 2. vollständig überarbeitete Aufl.

[81] UKM Pulsschlag Ausgabe 09/02 (2002) Neues Curriculum und benotete Scheine, Umsetzung der neuen Approbationsordnung bringt viel Arbeit / Symposium am 7./8. Dezember. Internet: http://www.klinikum.uni-muenster.de/organisation/pulsschlag/alt/ausgaben/2002/09_02/neuescurriculumundbenotetescheine.php (Zugriff 05.10.2006 08:21 MEZ)

[82] Ullenboom C (2006) Java ist auch eine Insel, Programmieren mit der Java Standard Edition Version 5, Galileo Press GmbH, Bonn, 5., aktualisierte und erweiterte Aufl.

[83] Universität Ulm – Fakultät für Mathematik und Wirtschaftswissenschaften (2001) JavaScript, Gefahren und Anwendungsmöglichkeiten durch JavaScript. Internet: http://www.mathematik.uni-ulm.de/sai/ws01/portalsem/wiede/ (Zugriff 05.10.2006 08:22 MEZ)

[84] Universitätsklinikum Heidelberg, Labor für Computerunterstützte Ausbildung in der Medizin (2006) Prüfungssystem. Internet: http://galaxy.mi.hs-heilbronn.de:3333/myzms/content/e54/e354/index_ger.html (Zugriff 05.10.2006 08:22 MEZ)

[85] Universitätsklinikum Münster (2006) Engagement, Kreativität und Kompetenz. Internet: http://www.klinikum.uni-muenster.de/organisation/pulsschlag/ausgaben/2006/01_06/engagementkreativitaetundkompetenz.php (Zugriff 05.10.2006 08:23 MEZ)

[86] upm – Mediendienst der Universität Münster (2004) Pressemitteilung upm, Themenbezogener Unterricht und benotete Scheine, 136 Erstsemester der Medizin studieren nach neuer Approbationsordnung, Münster (upm), 15. April 2004. Internet: http://cgi.uni-muenster.de/exec/Rektorat/upm.php?rubrik=Alle&neu=0&monat=200404&nummer=05174 (Zugriff 05.10.2006 08:24 MEZ)

[87] Virtuelles Software Engineering Kompetenzzentrum (2006) Verfahren, Wasserfallmodell. Internet: http://www.software-kompetenz.de/?10121 (Zugriff 05.10.2006 08:24 MEZ)

[88] Vonhoegen H (2004) Einstieg in JavaServer Pages 2.0, Galileo Press GmbH, Bonn

[89] W3C (2006) Extensible Markup Language (XML).
Internet: `http://www.w3.org/XML/` (Zugriff 05.10.2006 08:24 MEZ)

[90] Weßendorf M (2006) Struts, Websites mit Struts 1.2 & 1.3 und Ajax effizient entwickeln, W3L, Herdecke Bochum, 2. Aufl.

[91] Westfälische Wilhelms–Universität Münster (2005) Westfälische Wilhelms–Universität Münster: Jahresbericht 2005 – Automation von Geschäftsprozessen. Internet: `http://www.uni-muenster.de/Rektorat/jb05/Jb00dk.htm`(Zugriff 05.10.2006 08:30 MEZ)

[92] Wiesner S (2005) Struts 1.2, Tutorial für Java–Entwickler, Galileo Press GmbH, Bonn, 2., aktualisierte Aufl.

[93] Winkler J (2006) JavaScript: Einführung. Internet: `http://htmlworld.de/program/js_ov.php` (Zugriff 05.10.2006 08:30 MEZ)

[94] Wutka M (2002) J2EE Developer´s Guide, JSP, Servlets, EJB 2.0, JNDI, JMS, JDBC, Corba, XML, RMI, Markt+Technik, München

[95] Zimmermann J, Beneken G (2000) Verteilte Komponenten und Datenbankanbindung, Mehrstufige Architekturen mit SQJ und Enterprise JavaBeans 2.0, Addison–Welsley, München Boston San Franzisko Harlow Don Mills Sydney Mexiko Stadt Madrid Amsterdam

[96] Zwintzscher O (2005) Software–Komponenten im Überblick, Einführung, Klassifizierung Vergleich von JavaBeans, EJB, COM+, .Net, CORBA, UML 2, W3L GmbH, Herdecke Bochum

Lebenslauf

Zur Person	Name	Wolfram Urich
	Abschluss	Diplom-Mathematiker
	Geburtsdatum	09.01.1974
	Geburtsort	Rheda-Wiedenbrück
	Familienstand	ledig
Berufliche Praxis	seit Jan. 2007	Financial Analyst bei der Citibank in Düsseldorf
	Aug. 2004 – Sept. 2006	Wissenschaftlicher Mitarbeiter am Institut für Medizinische Informatik und Biomathematik in Münster
	Apr. 2001 – Okt. 2003	Studentische Hilfskraft im Projektteam »Medweb«
Ausbildung	Okt. 2004 – Sept. 2006	Promotionsstudiengang zum Dr. rer. medic.
	Okt. 1997 – Juli 2004	Studium der Mathematik an der WWU Münster Schwerpunkt: Informatik
	Okt. 1993 – Okt. 1997	Lehramtsstudium der Primar- und Sekundarstufe I an der WWU Münster Fächer: Mathematik, Deutsch, Religion
	Aug. 1985 – Juni 1993	St.-Georg-Gymnasium Bocholt

Münster, 25. Mai 2007

A Anhang - Datenbankschema

Die folgende Seite enthält eine Übersicht über die dem EPM zugrunde liegende Datenbank mit Stand vom 12.10.2006.

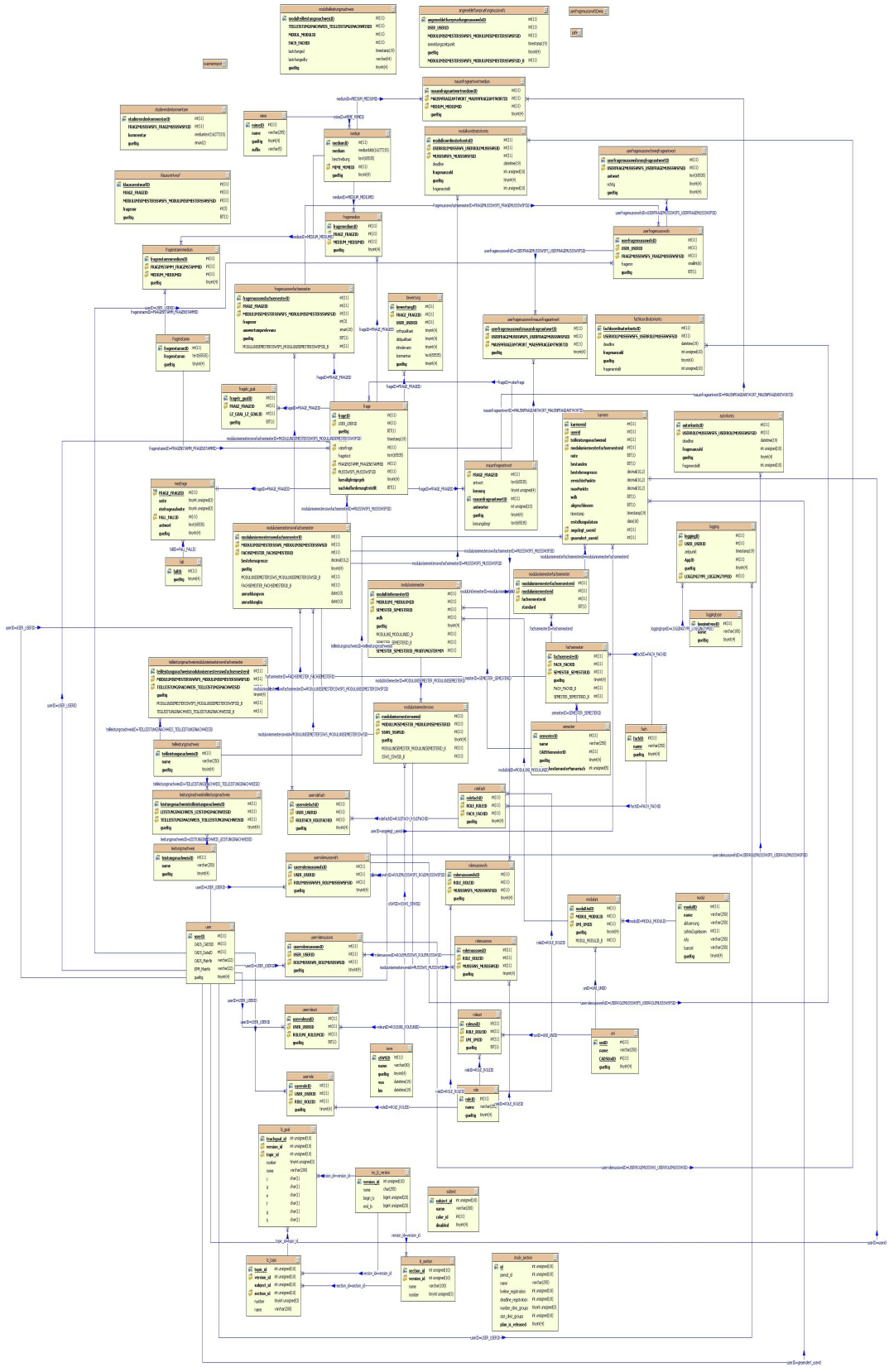


Abbildung A.1: Übersicht über die EPM-Datenbank