Gruppe 07
René Valdéz-Voges
Sabine Pertl
Peter Ellenberg
Gülten Demircan

Projektdokumentation

"AW Fach Auskunft"

mit

Jakarta

Ein Projekt in der Vorlesung

Software Engineering II

Prof. Dr. Streng

Wintersemester 2002 / 2003

Dokumentkontrolle

Datei	Doku-AWFachAuskunft.doc
Stand	23.01.2003 20:05
Druck	02.02.2003 19:27

Dokument Historie				
Version	Datum	Autor	Bemerkung	Status
0.1	15.01.2003	R.Valdéz	Dokument angelegt	new
0.2	23.01.2003	R.Valdéz	Dokument reviewed Klassendiagramme hinzugefügt	update
Release Stati: new → update (intern/extern) → release (intern/extern) → closed				

Inhaltsangabe:

Abbildungsverzeichnis:

1 Einleitung

Dieses Dokument wurde im Rahmen der Vorlesung "Software Engineering 2" bei Prof. Dr. Streng erstellt. Es soll als Dokumentation des Projekts "AW Fach Auskunft" dienen, und weitere Gruppen, die an diesem Projekt weiterarbeiten sollen, unterstützen. Demnach wird es im Laufe der Semester weiter ergänzt werden.

2 Projektbeschreibung

Es soll eine Anwendung konzipiert und implementiert werden, welche es dem Anwender ermöglichen soll, gezielt AW Fächer zu finden. AW Fächer sind Vorlesungen des Fachbereichs 13, die von allen Fachbereichen der Fachhochschule München belegt werden müssen. Da sehr viele dieser Fächer angeboten werden, soll es für Studenten leicht möglich sein, ein betreffendes Fach zu finden. Im Moment gibt es nur ein PDF Dokument, in dem alle Fächer aufgelistet sind. Eine gezielte Auswahl von Fächern nach verschieden Kriterien ist nicht möglich, da diese nur durchnummeriert in diesem Dokument vorliegen. Daher ist es wichtig, eine Funktionalität zur Verfügung zu stellen, die ein leichtes Auffinden eines AW Fachs zulässt. Zum anderen muss die Liste an AW Fächern auch von einem Sachbearbeiter in dem Fachbereich administriert werden. Die Daten sollen persistent in einer Datenbank gespeichert werden. Diese Anforderungen werden im folgenden Use-Case Diagramm veranschaulicht.

3 Use Cases

In den Use Cases der Anwendung wurden zwei Akteure identifiziert:: Ein *Anwender* und ein *Administrator*. Nun werden die einzelnen Use Cases beschrieben, die Grafik Abbildung 1 veranschaulicht diese.

3.1 AW Fach suchen

Ein Anwender muss AW Fächer finden können. Dies wird durch Selektion nach verschiedenen Kriterien ermöglicht. Eine Volltextsuche ist zuerst nicht geplant, da es sich um strukturierte Daten handelt, die in einer begrenzen Anzahl vorliegen (c.a. 160).
Auch der Administrator soll leicht ein AW Fach finden können, deshalb nutzt auch dieser den Use-Case. Der Akteur Anwender nutzt nur diesen Use-Case.

3.2 AW Fach anlegen

Der Administrator soll ein neues AW Fach anlegen können. Hierbei sind keine weiteren Use-Cases notwendig.

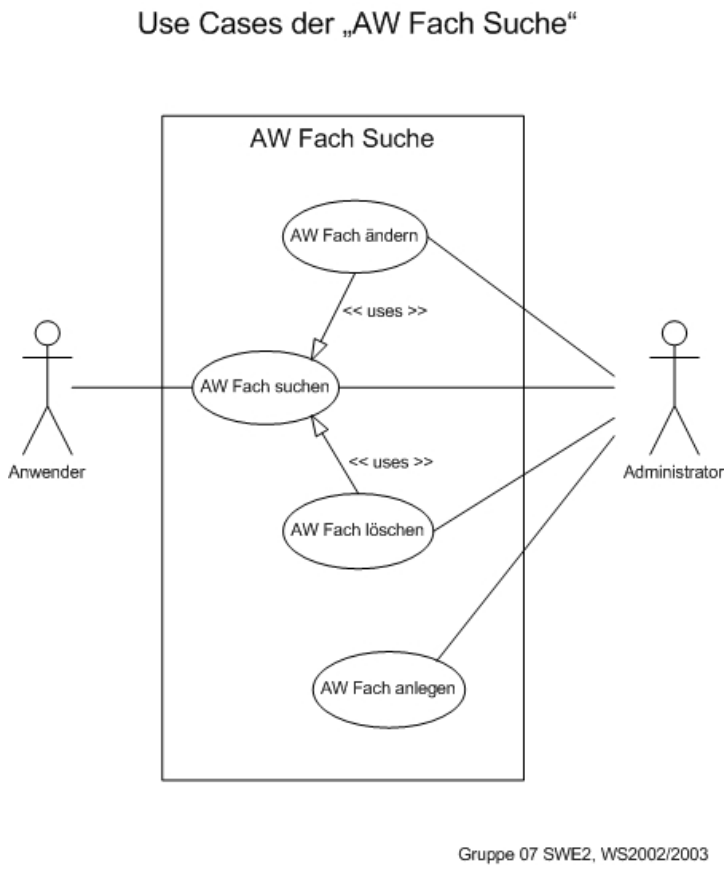

Abbildung 1: Use Cases

3.3 AW Fach ändern

Ist ein AW Fach angelegt, soll auch eine nachträgliche Änderung möglich sein. Dazu bedient sich der Administrator des Use Cases AW Fach suchen, um das AW Fach zu finden, das geändert werden soll.

3.4 AW Fach löschen

Ist ein AW Fach obsolete muss es gelöscht werden. Hierzu muss der Administrator wieder das AW Fach suchen und kann dann den Eintrag löschen.

4 Pakete

Die Abbildung 2 veranschaulicht, in welche Pakete die Anwendung aufgeteilt wurde. Die AW Auskunft ist im root Package *siab.aw.*. Die anderen Packages sind Teil des Gesamtframeworks "SIAB" und die Pfeile erläutern die Schnittstellen zu den anderen Komponenten. Nun soll näher auf die einzelnen Pakete eingegangen werden.

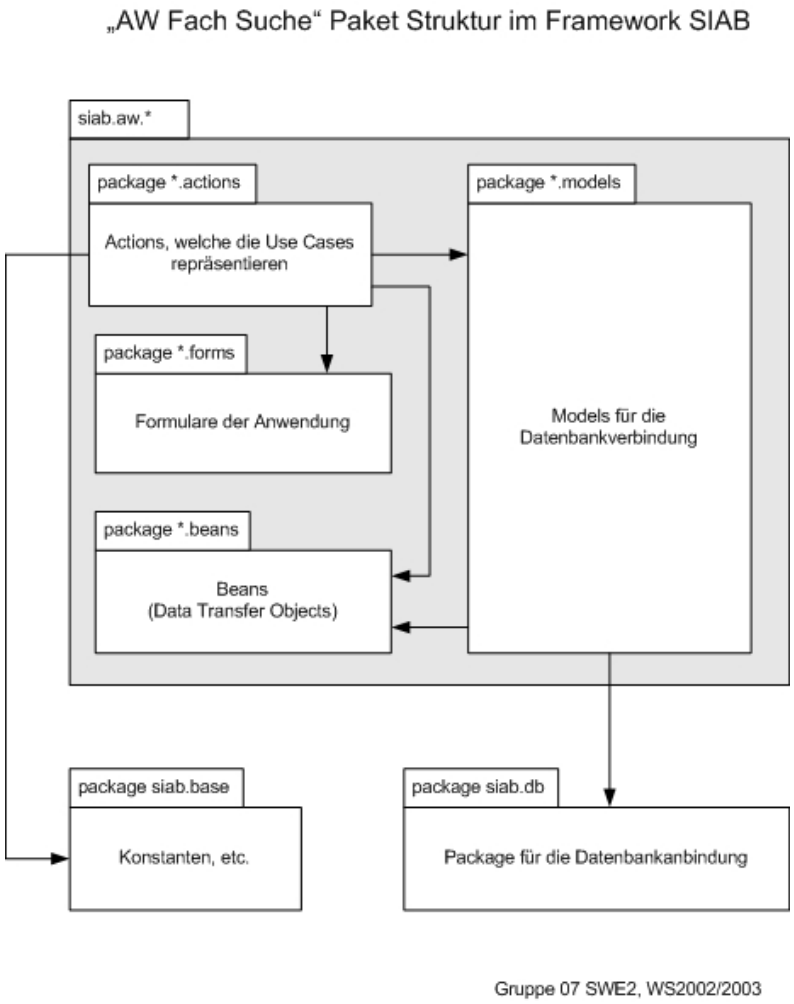

Abbildung 2: Packagediagramm

4.1 siab.aw.actions

In diesem Package liegen alle Action Klassen, die von der Klasse "Action" des Struts-Frameworks abgeleitet wurden. Da sie vom ActionServlet des Struts Frameworks aufgerufen werden, eine Referenz des Servlets erhalten und die Request - Response Objekte verwalten, müssen sie dem Controller im MVC Pattern zugeordnet werden. Daher repräsentieren diese auch im weiteren Sinne die Use Cases der Anwendung.

4.2 siab.aw.forms

Struts unterstütz den Programmierer durch Formular Klassen. Diese nehmen als einfache Beans mit getter und setter Methoden die Eingaben des Benutzers auf und können schon bevor sie an die betreffende Action weitergeleitet werden, eine Eingabevalidierung vornehmen. Sie können daher der View zugeordnet werden.

4.3 siab.aw.beans

In diesem Paket liegen Beans, die einzelnen Tabellen der Anwendung entsprechen. Sie verfügen ausschließlich über getter und setter und implementieren auch keine Programmlogik. Man kann sie auch als DTO, Data Transfer Objects bezeichnen, da sie nur zur Informationsübermittlung zwischen View und Model dienen.

4.4 siab.aw.models

Hier liegen die Klassen, die den Zugriff auf die Datenbank vollziehen. Sie müssen entsprechende Interfaces implementieren, so dass eine definierte Schnittstelle existiert. Im MVC Muster verkörpern diese Klassen das Model.

5 Klassen

Nun wird auf die Klassen in den Packages eingegangen. Es wird kurz beschrieben, was sie tun, und wo sie in der Anwendung eingereiht werden können.

5.1 Action Klassen

Wie bereits erwähnt liegen diese Klassen im Package "siab.aw.action". Bis auf "AnzeigenAction" repräsentieren sie die Use-Cases.

5.1.1 AnzeigenAction

Diese Klasse sorgt für die Anzeige. Es werden die Daten für die Select Boxes in der View über das Model geholt und einzelne AW Fächer können über diese angezeigt werden. Es ist keine Eingabe oder ein Formular notwendig, sie wird direkt aufgerufen.

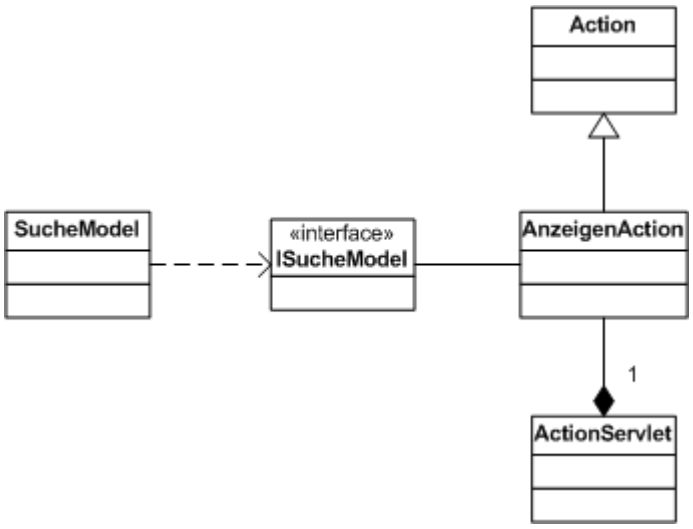

Abbildung 3: Klassendiagramm "AnzeigenAction"

5.1.2 SucheAction

In der Eingabemaske der View wird bestimmt, nach welchen Kriterien ein oder mehrere Aw Fächer gesucht werden sollen. Die Klasse ruft nun die entsprechenden Methoden des Models auf und leitet die Ergebnisse an die View weiter.

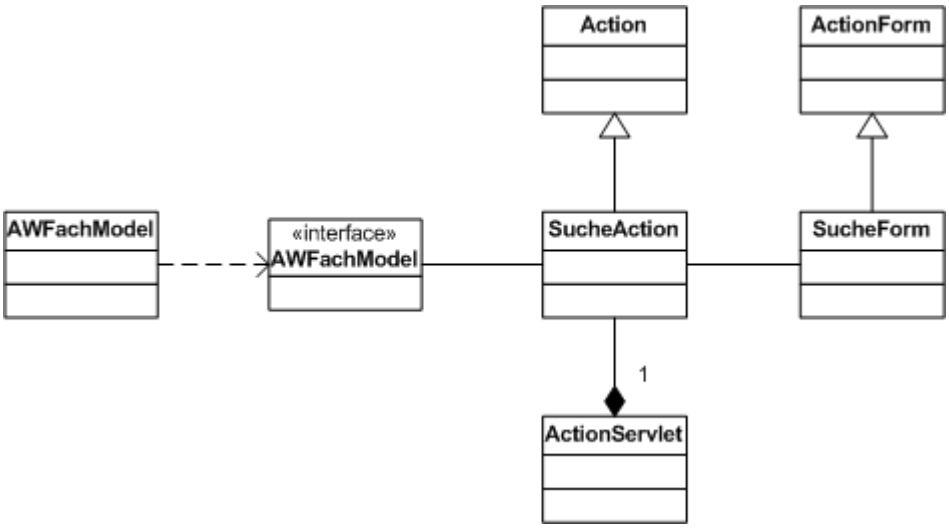

Abbildung 4: Klassendiagramm "SucheAction"

5.1.3 AnlegenAction

Wird das Formular für das Anlegen eines AW Fachs korrekt gefüllt, kann diese Klasse die Daten für ein neues AW Fach an das Model weiterreichen.

5.1.4 AendernAction

Es wird ebenfalls ein Formular mit den neuen Daten gefüllt und nach Validierung an diese Klasse weitergeleitet. Es wird wieder die entsprechende Methode des Model aufgerufen, welche das AW Fach in der Datenbank anlegt.

5.1.5 LoeschenAction

Diese Klasse empfängt eine entsprechende Form, aus der die ID des AW Fachs, das gelöscht werden soll, festgestellt wird. Eine Methode des Models, von dieser Klasse aufgerufen, löscht dann das AW Fach aus der Datenbank.

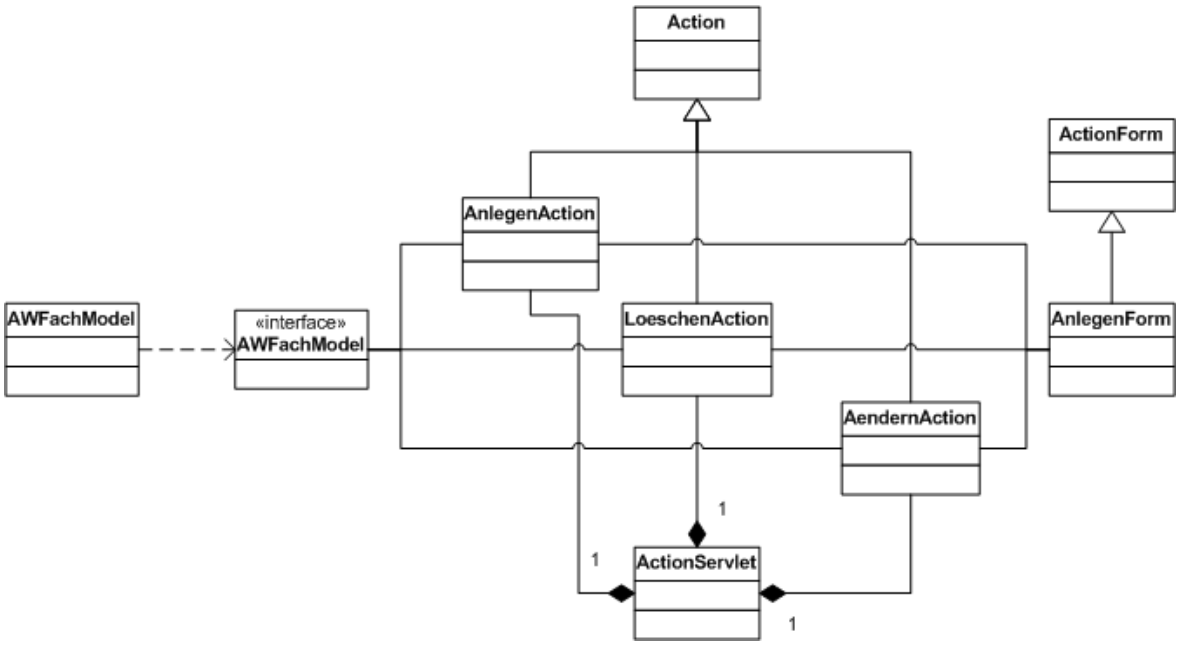

Abbildung 5: Klassendiagramm "AnlegenAction", "LoeschenAction", "AendernAction"

5.2 Form Klassen

Die Formklassen nehmen die Eingaben des Users entgegen. In der View (also die JSPs) werden sie in den <*html:form*> Tags definiert und gefüllt. Da die Eingabedaten sich gleichen, wurden nur zwei Form Klassen erstellt.

5.2.1 SucheForm

In diese Form kommen die Suchkriterien, die der Benutzer definiert. Ein weiteres Feld für eine spätere Erweiterung für die Volltextsuche wurde bereits berücksichtigt.

5.2.2 AnlegenForm

Hier wird im Prinzip ein AW Fach definiert. dies ist sowohl beim Anlegen, wie auch beim Ändern oder Löschen so, deshalb wird diese Form von allen drei Actionklassen benutzt.

5.3 Bean Klassen

Diese enthalten Daten, die in der Datenbank enthalten sind, und dienen als Austauschobjekte zwischen View und Model. Folgende Objekte gibt es:

Name der Bean	Beschreibung
AWFach	Repräsentiert ein AW Fach
Gebaeude	Daten für ein Gebäude
Kompetenz	Daten für Kompetenz
Prof	Professor
Pruefung	Daten über Prüfungen
Tag	Tage
Termin	Termin = Tag + Uhrzeit+Gebäude+ Zimmernummer
Thema	Ein Thema, dem ein AW Fach zugeordnet ist
Uhrzeit	Uhrzeit
ErrorMessage	Dient für die Übermittlung von Fehlermeldungen

5.4 Model Klassen

Das Model wird durch zwei Klassen umgesetzt. Da die Suche andere Daten benötigt, als die AW Fächer, wurden es einem anderen Model zugerechnet.

5.4.1 SucheModel

Es werden alle Daten aus der Datenbank geholt, die für die Suche eines AW Fachs benötigt werden. Es muss das Interface ISucheModel implementieren, in dem die benötigten Methoden abstrakt definiert sind.

5.4.2 AWFachModel

Auch hier wurde ein Interface IAWFachModel zuerst definiert, und anschließend implementiert. Hier werden AW Fächer geholt, angelegt, geändert oder gelöscht.

6 JSP Dateien

Nun soll auf die View eingegangen werden. Es wurde korrespondierend mit jeder Action eine JSP erstellt, welche die Daten anzeigt. Besitzt der User Admin Rechte, werden mehr Hyperlinks angezeigt, über die er dann die Aktionen Anlegen, Änder oder Löschen ausführen kann. Es wurde grundsätzlich versucht, die Tags aus den von Struts mitgelieferten Taglibs zu verwenden. Dies gelang leider nicht immer (vgl. hierzu auch Kapitel 9.4).

6.1 header.jsp

Alle JSP Seiten inkludieren einen Header, in dem eine Menüführung eingebaut werden kann. Dort können auch Seitenübergreifende Informationen stehen.

6.2 footer.jsp

Wie beim Header dient der Footer für Informationen, die in der ganzen Anwendungen angezeigt werden sollen.

6.3 index.jsp

Dient als Begrüßungsseite und Einstiegspunkt in die Anwendung.

6.4 suche.jsp

Hier befindet sich die Eingabemaske für die Suche. In Select Boxes werden die Suchkriterien angezeigt. In der Tabelle, welche die gefundenen AW Fächer anzeigt, kann auch ein Fach geändert oder gelöscht werden, wenn der User über Admin Rechte verfügt.

6.5 anlegen.jsp

In der Eingabemaske kann der Administrator alle benötigten Daten für ein AW Fach eingeben. In den Select Boxes wird das AW Fach einem Gebäude, Prüfungsart, etc. zugeordnet.

6.6 aendern.jsp

Hier ist die Eingabemaske bereits mit den Daten des zu ändernden AW Fachs gefüllt. Änderungen können dann in den Textfeldern oder Select Boxes vorgenommen werden.

6.7 loeschen.jsp

Es wird wie beim Ändern das AW Fach angezeigt. Allerdings lassen sich die Werte nicht ändern. Durch ein Bestätigen wird das angezeigte AW Fach gelöscht.

6.8 error.jsp

Diese JSP wird dann aufgerufen, wenn ein Fehler in der Anwendung aufgetreten ist. Da nicht immer nur bei der Eingabe, sprich über eine Form, ein Fehler auftreten kann, wurde auf den Error Mechanismus des Struts Frameworks verzichtet.

7 Datenbankdesign

Die Daten über die AW Fächer sollen in der Datenbank abgespeichert werden. Hierzu wurde das aktuelle Verzeichnis des Fachbereichs 13 analysiert und anschließend das Datenmodell aufgestellt, welches in Abbildung 6 veranschaulicht wird.

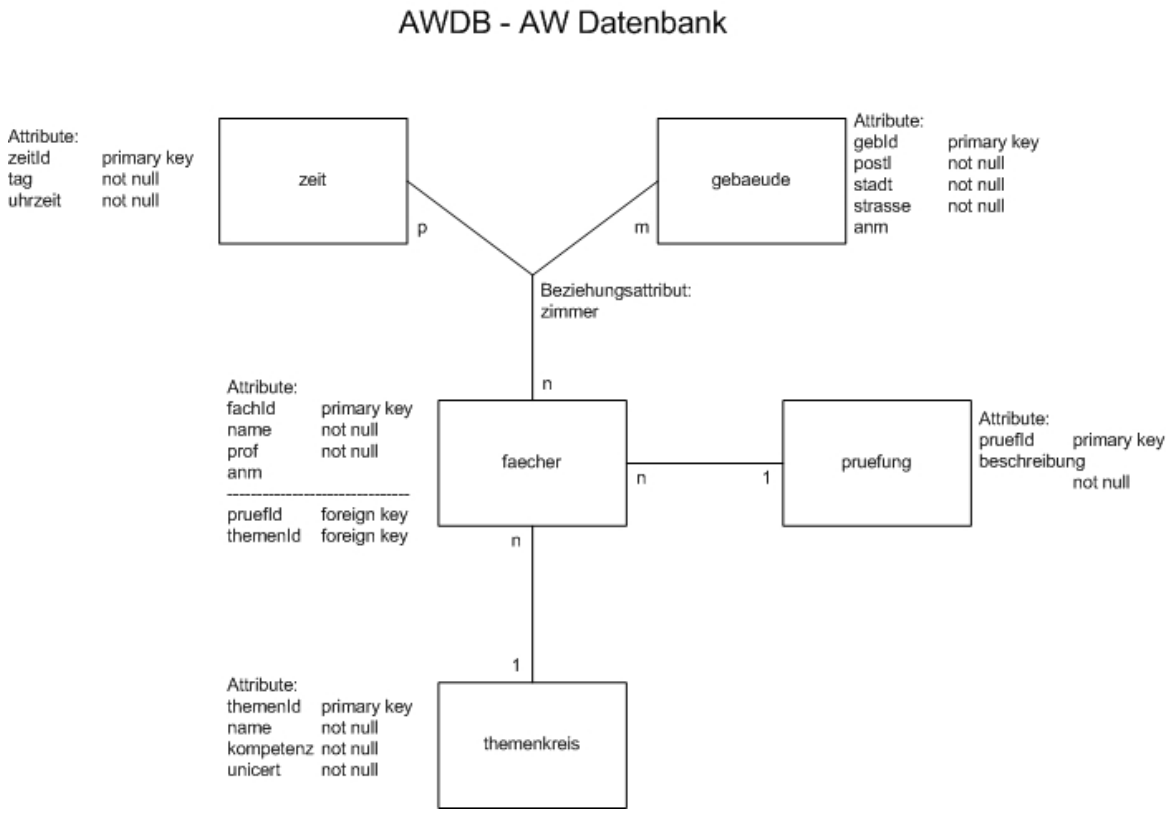

Abbildung 6: Datenbankdesign

Es wurden die Entitäten *faecher, zeit, gebaeude, pruefung und themenkreis* identifiziert. Da ein AW Fach zu verschiedenen Zeiten auch in verschieden Gebäuden stattfinden kann, werden diese drei Entitäten assoziiert. Das Beziehungsattribut Zimmer markiert dann den Raum, der zu diesem Zeitpunkt in jenem Gebäude belegt wird. Im Objektmodell wird diese Dreiecksbeziehung *Termin* genannt (siehe Kapitel 5.3) Die anderen Beziehungen wurden als 1:n identifiziert. Dies ist noch nicht 100% optimal, da noch Abhängigkeiten bestehen, die aber in der ersten Phase des Projekts noch nicht relevant waren. Welche Änderungen diesbezüglich noch vorgenommen werden können, sind im Kapitel 9.2 beschrieben.

8 Konfiguration in der struts-config.xml

Im Struts Framework findet die Konfiguration zentral in einer XML Datei statt. In diesem Kapitel wird beschrieben, wie sie für die Programmierung der AW Auskunft konfiguriert wurde.

8.1 Konfiguration der Formulare

Im oberen Abschnitt der Datei werden die Formulare definiert. Wie bereits in Kapitel 5.2 erwähnt, gibt es zwei Formulare: SucheForm und AnlegenForm. Entsprechend werden diese definiert und einem Namen zugeordnet. Dies ist wichtig, damit das Struts Framework zum einen das Formular einer Action Klasse zuordnen kann, zum anderen das Formular in der View, also JSP, füllen kann.

```
<form-beans>
  <!-- Suche form bean -->
  <form-bean name="sucheForm" type="siab.aw.forms.SucheForm"/>
  <!-- Anlegen form bean -->
  <form-bean name="anlegenForm" type="siab.aw.forms.AnlegenForm"/>
</form-beans>
```

8.2 Konfiguration der Forwardings

Forwardings erleichtern die Definition und Konfiguration des Aktionsflusses in einer Anwendung. Das sie hier zentral konfiguriert werden, müssen sie nicht statisch in den Actionklassen angegeben werden, sondern die Actionklasse verweist einfach auf den Namen der JSP. Soll die Umlenken in jeder Actionklasse angegeben werden können, muss sie hier global definiert werden. Soll die Weiterleitung nur für eine bestimmte Actionklasse gelten, muss sie im Actionmapping definiert werden.

```
<global-forwards>
  <forward name="index" path="/aw/jsp/index.jsp"/>
  <forward name="errors" path="/aw/jsp/errors.jsp"/>
  <forward name="aendern" path="/aw/jsp/aendern.jsp"/>
  <forward name="loeschen" path="/aw/jsp/loeschen.jsp"/>
</global-forwards>
```

8.3 Konfiguration der Action Klassen

Hier werden die Action Klassen in das Struts Framework eingebunden. hierzu wird der (URL) Pfad angegeben und die Klasse mit vollqualifizierten Pfad. Der Gültigkeitsbereich wird mit dem Zusatz *scope* bestimmt. Die dazugehörige Form muss ebenso angegeben werden, wie auch lokale Forwardings.

```
<action-mappings>
  <action    path="/aw/jsp/anzeigen"
             type="siab.aw.actions.AnzeigenAction"
             scope="request"
             input="/aw/jsp/index.jsp">
  </action>
  <action    path="/aw/jsp/suche"
```

```
        type="siab.aw.actions.SucheAction"
        name="sucheForm"
        scope="request"
        input="/aw/jsp/suche.jsp">
    <forward name="found" path="/aw/jsp/suche.jsp"/>
</action>
<action    path="/aw/jsp/anlegen"
          type="siab.aw.actions.AnlegenAction"
          name="anlegenForm"
          scope="request"
          input="/aw/jsp/anlegen.jsp">
    <forward name="end" path="/aw/jsp/index.jsp"/>
    <forward name="next" path="/aw/jsp/anlegen.jsp"/>
</action>
<action    path="/aw/jsp/aendern"
          type="siab.aw.actions.AendernAction"
          name="anlegenForm"
          scope="request"
          input="/aw/jsp/aendern.jsp">
    <forward name="end" path="/aw/jsp/index.jsp"/>
    <forward name="next" path="/aw/jsp/suchen.jsp"/>
</action>
<action    path="/aw/jsp/loeschen"
        type="siab.aw.actions.LoeschenAction"
        name="anlegenForm"
        scope="request"
```

```
            input="/aw/jsp/loeschen.jsp">
        <forward name="end" path="/aw/jsp/index.jsp"/>
        <forward name="next" path="/aw/jsp/suchen.jsp"/>
    </action>
</action-mappings>
```

9 Erweiterungsmöglichkeiten

Hier noch ein paar Vorschläge, wie die AW Auskunft noch erweitert, bzw. verbessert werden könnte:

9.1 Internationalisierung

Aus Zeitgründen wurde kaum auf eine Internationalisierung geachtet. Diese müsste durch die Konfiguration der *ApplicationRessources.properties* umgesetzt werden

9.2 Verbesserung des Datenbankdesigns

Wie bereits erwähnt gibt es im Datenmodell noch Abhängigkeiten. Diese sollten in eine KBNF übereinstimmende Form gebracht werden. Z.B. soll es möglich sein, Kompetenzen unabhängig vom Thema anzulegen, oder Professoren unabhängig von den AW Fächern. Dies betrifft v.a. den nächsten Punkt.

9.3 Anlegen, Ändern und Löschen der Suchkriterien

Es sollen nicht nur AW Fächer, sondern auch die mit den AW Fächern verbundenen Daten wie Thema, Gebäude oder Professoren angelegt, geändert oder gelöscht werden können. Hierzu wurden bereits die Form Klassen und JSPs vorbereitet. Die Funktionalitäten sind allerdings noch auskommentiert.

9.4 Programmierung von eigenen Tags

Für die Iterierung der AW Fächer oder die ID Zuweisung der Hyperlinks für das Ändern oder Löschen wurde mit Skriptlets umgesetzt.

9.5 Volltextsuche

Es wurde noch keine Volltextsuche eingebaut. Dies kann hinzugefügt werden. Hierzu wurde bereits in der SucheForm ein entsprechendes Feld definiert. Die SucheAction müsste entsprechend geändert werden, und das AWFachModel erweitert werden.

9.6 Performancesteigerung der Datenbankzugriffe

Es wird im Moment für jedes gefundene AW Fach ein eigenes SQL Statement abgesetzt. Dies kann sich negativ auf die Performance auswirken. Daher liegt hier noch verbesserungsbedarf.

9.7 Erweiterung zur "AW Fach Anmeldung"

Die größte Erweiterung wäre das Hinzufügen des Use Cases "AW Fach anmelden". Es soll sich der betreffende User für ein AW Fach, das er gefunden hat, auch gleich anmelden können. Hierzu muss die View entprechend angepasst werden. Die Anmeldung kann als eine Art "Warenkorb" in der Session gehalten werden. Nach einer Bestätigung der Anmeldung wird in der User Tabelle Einträge für die angemeldeten AW Fächer des Users hinzugefügt. Dies erfordert also auch eine Änderung des User Konzepts. Anschließend könnte z.B. die Bestätigung als PDF File abgespeichert werden und/oder eine E-Mail versendet werden.