Universität Zürich
Institut für Informatik

Dynamic and Distributed
Information Systems

Adaptivität im E-Learning:
Entwicklung eines Ajax-basierten Eintrittstests für den Einsatz in Lernplattformen

Bachelorarbeit 24. Oktober 2007

Anthony Lymer
Zürich, Schweiz

Student-ID: 04-727-871
a.lymer@access.uzh.ch

Betreuerin: **Katharina Reinecke**

Prof. Abraham Bernstein, PhD
Institut für Informatik
Universität Zürich
http://www.ifi.unizh.ch/ddis

Zusammenfassung

Das Ziel dieser Arbeit ist es, einen adaptiven Eintrittstest für ein bereits existierendes Lernsystem zu gestalten. Dabei handelt es sich um das CasIS-Portal, welches das Bearbeiten von Fallstudien elektronisch unterstützt. In dieser Arbeit wird auf den Eintrittstest und dessen Schnittstellen zu diesem Portal eingegangen.

Der Eintrittstest soll dem Benutzer vor der Bearbeitung einer Fallstudie helfen, indem er sein aktuelles Können einschätzt und ihn dementsprechend berät. Nachdem der Test die Kompetenz des Benutzers geprüft hat, werden ihm Lernmaterialien angeboten, damit ihm die Möglichkeit eingeräumt wird, sich optimal auf die Fallstudie vorzubereiten. Der Eintrittstest ist dabei nicht als Hürde zu verstehen, sondern bietet vielmehr eine Option, um Informationen über das eigene Können zu erhalten.

Die Arbeit umfasst sowohl die Entwicklung eines Authoring-Tools um Tests zu erstellen als auch eine Delivery-Engine, welche den Benutzer mit Fragen beliefert und ihm anschliessend Vorbereitungsmodule anbietet.

Abstract

The aim of this thesis is to develop an adaptive assessment test for the CasIS-Portal - an already existing e-learning system, which assists users electronically with solving case studies.

This thesis is concerned with designing and attaching an assessment test to the portal.

The test should give support to a user by estimating his current knowledge and then informing him accordingly.

After a test has been taken and the candidate's competence has been assessed, he will be shown the learning material, such that he has the possibility of preparing himself optimally for the subsequent case study.

The test itself is not to be understood as an obstacle, but rather it provides an opportunity of obtaining information about one's current knowledge level.

The work done comprises not only the development of an authoring-tool to create tests but also a delivery-engine, which presents questions to a candidate and offers him preparation modules.

Inhaltsverzeichnis

1

Einleitung

1.1 Swiss Virtual Campus

Das Projekt Swiss Virtual Campus (SVC) hat das Ziel das Lernen mit digitalen Medien (E-Learning) an Schweizer Hochschulen in höchster Qualität zu fördern. E-Learning soll hier vor allem über das Internet erfolgen und so den Studierenden das selbstständige und zeitunabhängige Lernen ermöglichen. Dabei sollen neuste pädagogische und didaktische Erkenntnisse in die Unterrichtsmethoden einfliessen.

1.1.1 Das Projekt „Cases in Information Systems“

Das CasIS-Projekt (Cases in Information Systems) ist ein Teil des SVC und beschäftigt sich mit der Ausarbeitung und Bereitstellung von qualitativ hochstehenden Fallstudien im Bereich der Wirtschaftsinformatik. Die Bearbeitung der Fallstudien wird durch vier Module unterstützt:

- Eintrittstest: Der Eintrittstest prüft das momentane Wissen des Kandidaten und gibt ihm gegebenenfalls eine Liste mit Vorbereitungsmodulen aus.
- Vorbereitungsmodule: Anhand der Vorbereitungsmodule kann er sein Wissen, das für die Bearbeitung der Fallstudie notwendig ist, erweitern.
- Fallstudien: Bei diesem Modul handelt es sich um die eigentliche Fallstudie, die von dem Kandidaten nach der Vorbereitung gelöst wird.
- Toolbox: Die Toolbox bietet Werkzeuge zur komfortablen Bearbeitung der Fallstudien.

In dieser Arbeit wird auf die Hintergründe und die Umsetzung des Eintrittstests eingegangen.

1.2 Motivation

Der Eintrittstest soll die Fähigkeiten eines Studenten einschätzen und somit akkurate Empfehlungen ausschreiben können. Indem der Studierende Informationen über sein eigenes Können in gewissen Bereichen erhalten kann, soll er optimal auf die Fallstudie vorbereitet werden. Durch die Möglichkeit themenspezifisches Wissen noch vor der Bearbeitung der Fallstudie zu erarbeiten, fühlt sich der Studierende nicht überfordert und soll die Aufgaben erfolgreich lösen können.

1.3 Struktur der Arbeit

Die Arbeit beginnt mit dem Definieren der genauen funktionalen und qualitativen Anforderungen an den Test. Danach werden verwandte Arbeiten untersucht und konkret besprochen, wie die Eigenschaften des Eintrittstests umgesetzt werden sollen. Auf die technischen Aspekte der Umsetzung wird dabei in einem nächsten Kapitel eingegangen. Dabei werden für jede entwickelte Komponente alle verwendeten Werkzeuge und Technologien aufgezählt und beschrieben.

Nachdem detailliert erklärt wurde, wie der Eintrittstest umgesetzt wurde, folgt eine Schlussfolgerung, welche auf die gewonnenen Erkenntnisse eingeht. Die Arbeit schliesst mit einem Ausblick um mögliche zukünftige Veränderungen oder Erweiterungen rund um den Eintrittstest aufzuführen.

Im Anhang befindet sich ausserdem eine Anleitung für die Endbenutzer des CasIS-Portals.

2

Anforderungen an den Eintrittstest

2.1 Funktionale Anforderungen

Adaptivität von Tests Der Eintrittstest soll sich dem geschätzten Können eines Kandidaten anpassen können. Da davon ausgegangen werden kann, dass nicht alle Kandidaten über das gleiche Vorwissen verfügen, ist die Idee eines adaptiven Tests entstanden. Der Vorteil eines solchen Tests besteht darin, dass nicht jeder Benutzer denselben zu Gesicht bekommt. Der Test passt sich ständig den gegebenen Antworten eines Kandidaten an. Einem Kandidaten mit einem sehr grossen und detaillierten Vorwissen werden schwierigere Fragen gestellt, als einem Studierenden mit weniger Kenntnis. Dadurch, dass der Eintrittstest auf die gegebenen Antworten eines Benutzers eingeht, kann in kürzerer Zeit eine Aussage über den Wissensstand gemacht werden, was die Testzeit in der Regel reduziert. Ein Kandidat, der einfache Fragen über ein Thema falsch beantwortet hat, ist sicherlich nicht in der Lage schwierigere Fragen zu beantworten. Somit können die schwierigeren Fragen entfallen. Für den Kandidaten ist die Beantwortung des Tests zudem viel angenehmer, da er weniger unter- oder überfordert ist. Bei einem linearen Test könnten zu einfache Fragen auftreten, die der Benutzer möglicherweise als Fangfragen interpretiert und damit falsche Erkenntnisse über ihn gewonnen werden. Zudem wäre es möglich, dass seine Konzentration nachlässt und er somit Flüchtigkeitsfehler begeht. Auf der anderen Seite können aber auch zu schwierige Fragen auftreten, die ein verzerrtes Testergebnis provozieren, da Kandidaten zu raten beginnen.

Granularität Ein Test muss in mehrere Sektionen verschiedener Granularitäten gegliedert werden können. Dabei können die einzelnen Sektionen wiederum ineinander gekapselt werden, sodass eine Sektion gröberer Granularität eine feinere Sektion beinhaltet. In den jeweiligen Sektionen können sich eine oder mehrere Fragen befinden, wobei solche, die ein detaillierteres Wissen prüfen, einer Sektion feinerer Granularität zugehören.

Vorbedingungen Die Adaptivität des Tests soll anhand von Vorbedingungen umgesetzt werden. Dabei kann die korrekte Beantwortung von vorhergehenden Fragen als Bedingung für die Übermittlung späterer Items verwendet werden. Die Bedingungen werden nur ganzen Sektionen zugewiesen. Ausserdem müssen sie über verschiedene Granularitätsstufen hinweg gesetzt

werden. Es ist somit nur möglich, die Beantwortung eines Items einer gröberen Granularität als Vorbedingung zu definieren.

Fragetypen Die Fragetypen, welche in diesem Eintrittstest eingesetzt werden, sind Single- und Multiple-Choice Fragen. Single-Choice-Items (SC) sind Fragen mit nur einer richtigen Antwortalternative. Bei Multiple-Choice (MC) kann es auch mehrere korrekte Antworten geben - muss es aber nicht. Wie in [Krebs, 2004] erklärt, gibt es mehrere Arten von Multiple-Choice Fragen. Best-Antwort-Typen haben Antworten zugewiesen, bei denen nicht schwarz-weiss bestimmt werden kann, welche davon gänzlich falsch sind. Es muss dabei die beste Antwortmöglichkeit bzw. die besten Antwortmöglichkeiten gewählt werden. Neben den Best-Antwort-Typen existieren auch Richtig/Falsch-Fragen, bei denen die inkorrekten Antwortalternativen vollumfänglich inkorrekt sind und somit die Unterscheidung von richtig und falsch einfacher wird. In einem optimalen Test sollten mehr Best-Antwort-Typen vorkommen.

Da die verschiedenen Fragenarten strukturell gesehen ziemlich ähnlich sind, können in dem CasIS-Eintrittstest alle in [Krebs, 2004] definierten MC-Fragetypen umgesetzt werden.

Verlinkung mit Vorbereitungsmodulen Die Tests müssen mit Lernmaterialien verbunden werden können, sodass der Kandidat nach Abschluss des Tests mit Informationen über die Gebiete, welche die Fallstudie abdeckt, versorgt werden kann. Damit sollen Lücken im momentanen Wissensstand gefüllt werden, um den Benutzer optimal auf die Fallstudie vorzubereiten.

Testerstellung Ein Administrator muss die Möglichkeit haben, Tests zu erstellen. Hierfür wird ein Authoring-Tool entwickelt, welches die oben genannten Anforderungen an die Tests umsetzen kann.

Testauslieferung Die Tests müssen selbstverständlich den Kandidaten, welche ihren Wissensstand prüfen und Empfehlungen ausgeschrieben bekommen wollen, präsentiert werden.

Einbindung ins CasIS-Portal Der Eintrittstest muss in das CasIS-Portal eingebaut werden. Dabei ist zu beachten, dass ein einheitlicher Stil gewählt wird, sodass das komplette Portal als ein Ganzes auftritt.

2.2 Qualitätsanforderungen

Gebrauchstauglichkeit Die einfache Handhabung des Authoring-Tools und des eigentlichen Eintrittstests ist ein wichtiges Kriterium. Deswegen muss eine intuitive und angenehm zu bedienende Benutzerschnittstelle erstellt werden.

Browserunterstützung Da das CasIS-Portal webbasierend ist, muss auf das Rendering der verschiedenen Browser eingegangen werden. Der Eintrittstest und die damit verbundenen Komponenten werden deshalb browserübergreifend programmiert.

Änderbarkeit Der erzeugte Code muss sauber dokumentiert sein, sodass eine allfällige Erweiterung kein Problem darstellt. Ausserdem wird versucht, kohärente Komponenten zu erstellen, die nicht eng gekoppelt sind.

3

Verwandte Arbeiten

Im Bereich des adaptiven Testens wurden Arbeiten über Computer-Adaptive-Testing (CAT) gefunden, bei denen Fragen (Items) computergestützt ausgewählt und einem Kandidaten geliefert werden.

In [Linacre, 2000] werden die Items in verschiedene Schwierigkeitsstufen geordnet, wobei einem Kandidaten ein Item präsentiert wird, und je nach Antwort ein schwierigeres oder ein einfacheres nächstes Item folgt. Ein Item soll dabei einen gewissen Themenbereich abdecken, sodass bei der Beantwortung eine Annahme getroffen werden kann, ob der Kandidat das abgefragte Thema beherrscht. Als erstes Item wird bevorzugt ein etwas einfacheres ausgeliefert und präsentiert, da der Kandidat nicht abgeschreckt werden soll und er eine gewisse Sicherheit aufbauen kann. Bei einer Schwierigkeitsskala von 1 bis 100 könnte dem Kandidaten also zuerst ein Item mit der Schwierigkeit von 30 präsentiert werden. Bei einer richtigen Antwort würde dann ein Item des Schwierigkeitsgrades 40 ausgewählt. Nachdem diese Frage richtig beantwortet wurde, wäre z.B. der Schwierigkeitsgrad 53 an der Reihe. Bei einer inkorrekten Beantwortung dieses Items fällt das Schwierigkeitsniveau wieder. Dies geht solange weiter bis sich der Schwierigkeitsgrad irgendwann einpendelt und die Fähigkeiten des Kandidaten ermittelt wurden. In Abbildung 3.1 ist die Abfolge eines CAT grafisch ersichtlich.

In [Greer and Huang, 1996] werden zusätzlich zu den Schwierigkeitsstufen Granularitäten eingesetzt, indem Bereiche eines Tests in verschiedene Detailstufen gegliedert werden. Diese Bereiche können durch Vorbedingungen verknüpft werden, sodass neue Möglichkeiten entstehen, die Fähigkeiten eines Benutzers zu schätzen. Bei einer inkorrekten Antwort auf ein Item, das als Vorbedingung zu anderen Fragen angesehen werden kann, wird dem System klar, dass der Kandidat die nächsten Items nur mit einer kleinen Wahrscheinlichkeit richtig beantworten wird.

In den obigen beiden Arbeiten wurden für die Realisierung von „Computer-Adaptive-Tests“ Bayes-Netze vorgestellt, welche über die Wahrscheinlichkeitsrechnung versuchen, passende Annahmen über die Fähigkeiten der Kandidaten und die damit verbundene Auslieferung der Items zu machen. In dieser Arbeit kann getrost auf Bayes-Netze verzichtet werden, jedoch werden gewisse Ideen und Konzepte von Computer-Adaptive-Testing verwendet, welche bezüglich den Anforderungen an den Eintrittstest passend erscheinen.

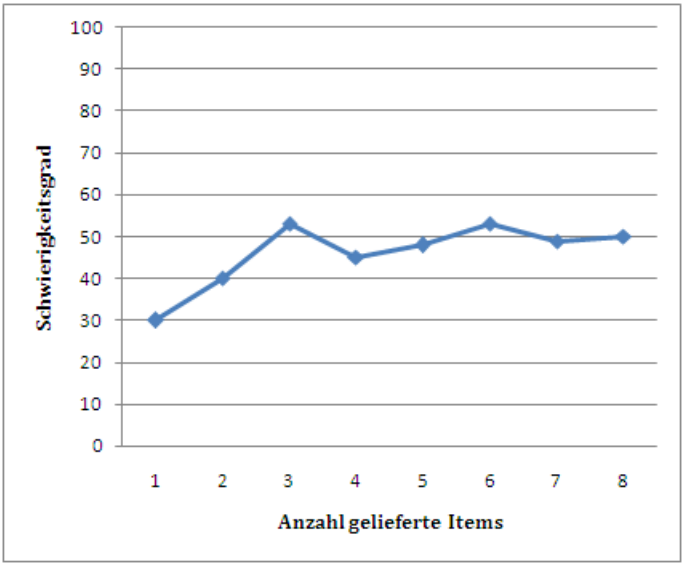

Abbildung 3.1: Abfolge eines Computer-Adaptive-Tests (in Anlehnung an [Linacre, 2000])

4

Umsetzen der Anforderungen

Obwohl der Eintrittstest in dieser Arbeit nicht mit Bayes-Netzen umgesetzt wird, können trotzdem gewisse Eigenschaften eines Computer-Adaptive-Tests extrahiert werden und in die Umsetzung mit einfliessen. So bieten zum Beispiel Granularitäten eine gute Möglichkeit einen Test zu strukturieren und zu ordnen. Durch die Vergabe von Vorbedingungen zwischen den Granularitätsstufen werden adaptive Tests möglich. Den einzelnen Items werden aber keine Schwierigkeitsstufen zugeordnet.

In dieser Arbeit setzt der Autor beim Erstellen des Tests Vorbedingungen, die bei der Auslieferung der Items ausgewertet werden. Diese werden benützt, um entweder weitere Fragen anzuzeigen oder sie zu überspringen. In der IMS QTI-Spezifikation wurde ausserdem ein Element mit dem Namen preCondition entdeckt, welches exakt für die hier benötigten Zwecke verwendet werden kann: Bevor ein Item einem Kandidaten ausgeliefert wird, muss zuerst geprüft werden, ob eine Bedingung erfüllt ist. Die Bedingung setzt sich dabei immer aus der korrekten Beantwortung vorhergehender Fragen zusammen. Auf QTI - ein Standardformat, welches von dem IMS Global Learning Consortium spezifiziert wurde - wird später in dieser Arbeit eingegangen.

Nachdem im Team rund um das CasIS-Projekt nochmals die Möglichkeiten eines aussagekräftigen und für den Kandidaten komfortablen Test ausgearbeitet wurden, war klar, dass ein Test in Sektionen verschiedener Granularitäten gegliedert werden soll. Anhand mehrerer Granularitäten kann Wissen auf verschiedenen Detailstufen geprüft werden. Um die Idee von Granularitäten zu konkretisieren, dient folgendes Beispiel in Anlehnung an [Greer and Huang, 1996]: Es wird ein Test über Arithmetik erstellt, der Wissen auf mehreren Ebenen prüfen soll. Neben einfachem Addieren, sollen auch schwierigere Fragen, welche die Addition, Division und die Multiplikation in Satzaufgaben kombinieren, gestellt werden, bei denen auch bessere Kandidaten ihre Mühe haben könnten. Die schwierigeren Fragen werden dabei aber nur den Benutzern gestellt, die auch die einfachen Fragen korrekt beantworten. Ein Test wird somit in die Sektion „Addieren“ und die Sektion „Satzaufgaben“ gegliedert, wobei die Sektion „Satzaufgaben“ vom Autor so eingerichtet wird, dass sie dem Kandidaten nur präsentiert wird, wenn er gewisse Fragen der Sektion „Addieren“ richtig beantwortet hat. Die Sektion „Satzaufgaben“ ist also von feinerer Granularität und das Ausliefern an den Kandidaten macht nur Sinn, wenn er die Fragen über das simple Addieren richtig beantwortet hat.

Ein Autor kann den verschiedenen Sektionen ausserdem Vorbereitungsmodule anhängen, die dem Kandidaten anschliessend an den Test angeboten werden. Bei diesen Lernressourcen handelt es sich um Module des „Foundations of Information Systems“ (FOIS). FOIS wurde im Juli 2004 gegründet und ist wie CasIS dem Swiss Virtual Campus zugeordnet. Es besteht aus mehreren Instituten von verschiedenen Schweizer Hochschulen und bietet E-Learning-Kurse im Bereich der

Wirtschaft an [foi, 2007]. Diese Module sollen dem Benutzer helfen gewisse Wissenslücken, die im Laufe des Tests bekannt wurden, zu schliessen. Die einzelnen FOIS-Module decken ein breites Themengebiet ab und sind in verschiedene Lerneinheiten gegliedert. Um eine detailliertere Zuweisung von Lernressourcen zu unterstützen, können nicht nur die gesamten FOIS-Module, sondern auch die einzelnen Lerneinheiten den Sektionen angehängt werden. Ein Autor kann zudem definieren, bei welcher Fehlerrate die Module angezeigt werden sollen. Wenn die gestellten Fragen eher schwierig sind, kann der Autor mit dieser Funktion bestimmen, dass das Vorbereitungsmodul erst bei einer höheren Anzahl an Fehlern empfohlen wird.

Um den Test so kurz wie möglich zu machen, wird bei jeder Antwort geprüft, ob der Benutzer die Fehlerrate bereits über- oder unterschritten hat. Wenn das System merkt, dass die Fehlerrate des Kandidaten unwiderruflich unter oder über der definierten Fehlerrate des Autors ist, wird in die nächste Sektion gesprungen. Der Kandidat wird dabei entweder in eine Sektion feinerer Granularität oder eines anderen Themengebiets geleitet.

Die Abfolge eines Tests ist so definiert, dass ein Benutzer von der gröbsten Sektion in die feineren Sektionen gelangt. Er kann sich auf diese Art und Weise immer weiter in das aktuelle Gebiet einfühlen und wird nicht durch Fragen eines anderen Themas gestört oder verwirrt. Wenn der Test zuerst alle gröbsten Granularitäten und dann jeweils alle feineren Granularitäten abfragen würde, dann wäre der rote Faden nicht unbedingt gegeben. Ein Test könnte mehrere Themen abdecken und da wäre es vor allem in einer feinen Granularität sehr verwirrend, wenn plötzlich eine Frage eines anderen Gebiets gestellt würde.

Um die einzelnen Fragen zu bewerten, kann ihnen eine Gewichtung von 1 bis 3 zugeordnet werden. Ein schwierigeres Item kann somit insgesamt mehr Punkte geben, als ein einfacheres. Konkret hängt die Bewertung von der zugewiesenen Gewichtung und der Anzahl korrekter Antworten eines Kandidaten ab. Bei einer korrekten Antwort bekommt der Kandidat

$$\frac{1}{Anzahl\ korrekte\ Antworten\ eines\ Items} * x$$

Punkte, wobei x die Gewichtung beschreibt. Bei einer falschen Antwort werden ihm analog die Punkte abgezogen. Für ein Item ist die minimale Punktzahl jedoch nach unten begrenzt, sodass auch bei ausschliesslich falschen Antworten eine Punktzahl von null resultiert. Aus dieser Formel geht heraus, dass die korrekte Beantwortung einer Frage mit einer höheren Gewichtung, für das Testergebnis bedeutend ist.

Diese Bewertung wird jeweils auf jeder Sektion durchgeführt und anschliessend wird die Fehlerrate des Kandidaten mit der für die Vorbereitungsmodule definierten Fehlerrate verglichen. Die Fehlerrate selbst wird über folgende Formel berechnet:

$$1 - \frac{Aktuelle\ Punktzahl\ in\ entsprechender\ Sektion}{Maximal\ mögliche\ Punktzahl\ in\ entsprechenderSektion}$$

Wenn die Fehlerrate des Benutzers höher ist, dann wird ihm das zugehörige FOIS-Modul angezeigt und entsprechend signalisiert. Bei einem tieferen Fehlerquotienten werden die Vorbereitungsmodule ebenfalls angezeigt. Hierbei soll aber klar ersichtlich sein, dass er gemäss dem Test über genügend Wissen hinsichtlich dem geprüften Thema verfügt. Bei Sektionen, die aufgrund nicht erfüllter Vorbedingungen ausgelassen wurden, ist die Anzahl korrekt beantworteter Items folglich null, weshalb die Vorbereitungsmodule bei allen übersprungenen Sektionen angezeigt werden.

Um den Kandidaten nicht unter Druck zu bringen, erfährt er nicht, wie viele Fragen er falsch beantwortet hat. Der Test ist ausschliesslich für den Kandidaten gedacht und soll keinen Wettbewerb darstellen und Vergleiche zu anderen Kandidaten ziehen. Deswegen wird auch von einer Highscore abgesehen. Ausserdem soll der Eintrittstest dem Benutzer keine Hürde sein, sondern als optionale Hilfe dienen.

Da mit verschiedenen Granularitäten Kandidaten mit hohem oder niedrigem Vorwissen geprüft werden können, werden viel gezieltere Testfragen möglich als bei einem simplen linearen Test. Ausserdem kann schneller auf das Wissen des Kandidaten rückgeschlossen werden, weshalb die Kandidaten nicht immer alle Fragen beantworten müssen.

5

Implementierung

In diesem Abschnitt wird auf die vorliegende Architektur und die in dieser Arbeit verwendeten Mittel und Technologien eingegangen. Ausserdem werden die entwickelten Komponenten im Detail vorgestellt.

5.1 Der QTI-Standard

Der „IMS Question & Test Interoperabilty"-Standard spezifiziert ein Datenmodell, das die Struktur von Tests und Fragen definiert. Das Datenmodell wird anhand von UML beschrieben und kann über das systemunabhängige XML-Format[1] implementiert werden. QTI hat sich seit den Anfängen in 1999 weit verbreitet und wird in mehreren Lernplattformen wie zum Beispiel WebCT und OLAT verwendet [Piendl, 2005, ola, 2007, qti, 2002a].

5.1.1 Gründe für Standardisierung

Der grosse Vorteil eines Standards ist die damit verbundene Interoperabilität von Systemen. Wenn ein Autor mit einem Test auf mehreren Systemen Kandidaten prüfen will, muss er ohne Standards die Fragen und Antworten jedes Mal neu eingeben. Wenn die Items, in den verschiedenen Systemen, aber alle die gleiche Struktur aufweisen, dann ist es möglich, diese Tests zu exportieren und zu importieren. Für einen Testautor nimmt der Aufwand also stark ab, wenn er auf mehreren Plattformen aktiv ist. Neben dem eigenen Vorteil, der für einen Ersteller von Tests besteht, sind natürlich auch Kollegen erfreut, wenn sie von jemandem einen fertigen Test bekommen, der nur noch importiert werden muss.

5.1.2 QTI-Versionen

Der QTI-Standard befindet sich momentan in der Version 2.1, welche als „Public Draft Specification Version 2" erschienen ist. Ausser bei ernsthafteren Problemen in der Umsetzung wird an der Funktionalität der QTI-Elemente mit höchster Wahrscheinlichkeit nichts mehr verändert, obwohl die Version noch nicht als „final" deklariert wurde [qti, 2007b]. Die Version 2.1 ist eine

[1] XML ist ein simples, flexibles und systemunabhängiges Textformat, das oft für den Transfer von Daten eingesetzt wird [xml, 2007]

Weiterentwicklung der Version 2.0, welche Änderungen und Verbesserungen der Version 1.2 auf Item-Ebene mit sich brachte [qti, 2007a]. Version 2.1 passte die Spezifikation der Version 2.0 dann in Bezug auf Sektionen und Tests an. Eine grössere strukturelle Änderung von QTI 1.2 auf QTI 2.1 ist, dass die Items nun als einzelne XML-Dokumente gespeichert werden [qti, 2002b]. Ausserdem wurde in QTI 2.1 die Möglichkeit zur Erstellung eines adaptiven Tests mitberücksichtigt. Dies kann entweder über Verzweigungen oder die im CasIS-Eintrittstest verwendeten Vorbedingungen realisiert werden.

5.1.3 In dieser Arbeit umgesetzt

QTI ist in dieser Arbeit nicht vollumfänglich implementiert, da eine Import-Funktion nicht spezifiziert wurde und die Betreiber des CasIS-Portals nur gewisse Fragetypen verwenden werden. Um aber für mögliche spätere Erweiterungen vorbereitet zu sein, wurde doch ein Teil bereits in QTI umgesetzt. Die Spezifikation beinhaltet viele Fragetypen verschiedener Komplexität. Neben Multiple-Choice-Items existieren z.B. Items, bei denen auf einem Bild eine Position gewählt werden muss. Wie in Kapitel 2.1 genauer ausgeführt, werden in dieser Arbeit die gängigen Fragetypen Single-Choice und Multiple-Choice verwendet. Zur Strukturierung eines Tests werden Sektionen verwendet, welche über Vorbedingungen gesteuert werden. Das System mit weiteren Fragetypen zu erweitern ist aber jederzeit möglich, weil Java-Klassen für die vollumfängliche Implementierung von QTI bereitstehen. Da die einzelnen Items, die von diesem System generiert werden, dem Standard entsprechen, steht einer zukünftigen Export-Funktion, anhand des IMS Content-Packaging-Standards[2], nichts im Wege. Die vom CasIS-Portal erzeugten Fragen und Antwortmöglichkeiten sind also theoretisch in jedes System importierbar, welches auf den QTI-Standard der Version 2.0 oder 2.1 setzt. Die Implementierung des Standards im CasIS-Portal kann somit als Basis-Implementierung der QTI-Spezifikation angesehen werden.

5.1.4 Referenzimplementierung R2Q2

Das R2Q2-Projekt stellt Web-Services zur Verfügung, mit denen QTI-Items angezeigt und verarbeitet werden können [r2q, 2007]. R2Q2 kann zur Überprüfung der Richtigkeit von erstellten QTI-Items verwendet werden, sodass Implementierer sicherstellen können, dass sie den Standard richtig verwenden. „To help these stakeholders [early adopters] we have provided a Web client to which they can submit questions and see the rendered version“ [r2q, 2007]. Neben der Online-Demonstration, bei der eigene QTI-Items heraufgeladen werden können, ist es auch möglich, den Source-Code herunterzuladen und R2Q2 auf dem eigenen Webserver laufen zu lassen. In dieser Arbeit wurde R2Q2 jedoch lediglich als Referenz verwendet um die eigens entwickelte Delivery-Engine, also die Komponente, welche die Items an den Kandidaten ausliefert, auf ihre Richtigkeit zu prüfen.[3]

5.1.5 Übersicht über eine Lernplattform

Um eine Übersicht über eine Lernplattform zu bekommen, dient Abbildung 5.1, welche von der QTI-Webseite stammt und minimal ergänzt wurde. Sie soll einen Überblick über das zu implementierende System verschaffen, wobei gewisse abgebildete Komponenten im CasIS-Portal verschmelzen. Im Folgenden werden kurz die Systeme und Akteure beschrieben:

[2] Mit IMS Content-Packaging kann Lernmaterial gepackt und exportiert bzw. importiert werden. [ims, 2007]

[3] Die Engine von R2Q2 benützt andere Namensräume für die XML-Dateien. Die Items die mit CasIS erstellt werden, können aber sonst problemlos von der Engine angezeigt werden.

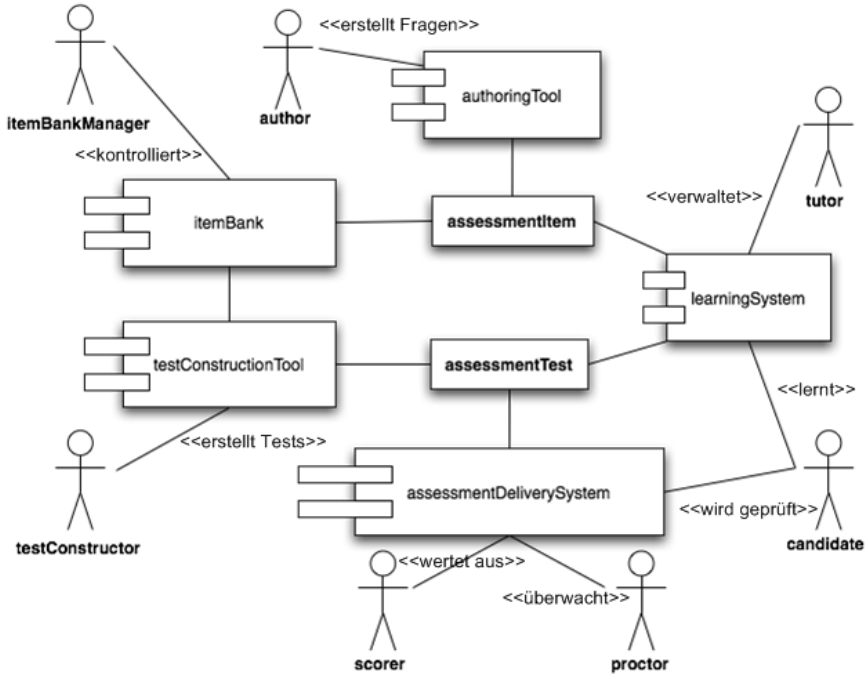

Abbildung 5.1: Übersicht über eine Lernplattform (in Anlehnung an [qti, 2006])

- AssessmentItem: Ein Item besteht aus der Frage und dessen möglichen Antworten.
- AssessmentTest: Ein Test besteht aus mehreren Items.
- AuthoringTool: Das Authoring-Tool bietet die Möglichkeit Items zu erstellen und zu editieren.
- ItemBank: Die Item-Bank ist eine Sammlung aus allen Items, die erstellt wurden.
- Testconstruction-Tool: Um aus den vom Authoring-Tool erstellten Items einen Test zu gestalten, wird ein Testkonstruktionstool verwendet.
- AssessmentDeliverySystem: Dieses System beliefert den Kandidaten mit Fragen und verarbeitet deren Antworten.
- LearningSystem: Das Learning-System beinhaltet die Tests und Lerninhalte, welche dem Lernenden präsentiert werden (in dieser Arbeit handelt es sich dabei um das gesamte CasIS-Portal).
- Author: Der Autor übernimmt die Aufgabe des Erstellens und Editieren von Items.
- ItemBankManager: Der Item-Bank-Manager kontrolliert die Sammlungen von Items in der Item-Bank.
- TestConstructor: Dieser Akteur generiert Tests anhand der Item-Sammlung.
- Scorer: Der Scorer bekommt die Ergebnisse von der Delivery-Engine und wertet sie aus. Dabei kann es sich entweder, wie in dieser Arbeit, um ein System oder um einen Menschen handeln.

- Proctor: Dieser Akteur überwacht den Auslieferungsprozess der Delivery-Engine, ist aber nicht mit dem Bewerten der Antworten von Kandidaten beschäftigt.
- Tutor: Der Tutor verwaltet oder unterstützt den Lernprozess eines Kandidaten.
- Kandidat: Ein Kandidat wird geprüft und beantwortet die Items, die von der Delivery-Engine geliefert werden.

Wie die Systeme in dieser Arbeit umgesetzt worden sind, ist in Kapitel 5.2 genauer beschrieben.

5.2 Architektur

Dieser Abschnitt beschäftigt sich mit den für den Eintrittstest verwendeten Technologien. Abbildung 5.2 bietet eine Übersicht über die Hauptkomponenten und deren Zusammenspiel untereinander.

5.2.1 Struts2 MVC-Framework

Da der Eintrittstest in das CasIS-Portal eingebunden ist, welches das Struts-Framework der Version 2 verwendet, wird auch in dieser Arbeit von Struts Gebrauch gemacht. Hierbei handelt es sich um ein Framework das auf J2EE setzt und eine Webapplikation in Model (Modell), Controller (Steuerung) und View (Präsentation) teilt. Diese Unterteilung wird als MVC-Pattern bezeichnet und ist ein sehr verbreitetes Architekturmuster. Das Modell beschäftigt sich hierbei lediglich um die Daten und hat keine Kenntnisse wie diese in der Präsentation-Komponente dargestellt werden. Die Präsentation befasst sich mit dem Design und der grafischen Benutzerschnittstelle. Woher die Daten kommen und wie sie aufbereitet wurden, ist für die Präsentation irrelevant. Die Steuerung stellt eine Schnittstelle zwischen Präsentation und Modell dar. Bei Struts sind dies Servlets, welche HTTP-Requests entgegen nehmen (gegebenenfalls mit dem Modell kommunizieren, um Daten aufzubereiten) und der Präsentation-Komponente das Resultat des Requests liefern. Wie in [str, 2007] beschrieben, könnte theoretisch eine Webapplikation nur aus JavaServer Pages[4] bestehen, welche alle drei Komponenten vereinen. Je grösser aber eine Applikation wird, desto schwieriger wird es sie zu pflegen, wenn keine Trennung stattfindet: „Web applications based on JavaServer Pages sometimes commingle database code, page design code, and control flow code. In practice, we find that unless these concerns are separated, larger applications become difficult to maintain“ [str, 2007]. Für die Präsentation-Komponente kommen statt JSP, Apache Velocity Templates[5] zum Einsatz. Die Überlegung dahinter ist, dass somit kein Java-Code in der View-Komponente vorkommen kann, was die View-Komponente von dem Controller trennt. Struts wird in dieser Arbeit allerdings nur verwendet um das Authoring-Tool und den Eintrittstest selbst zu laden (siehe Abbildung 5.2). Da in den einzelnen Komponenten des Eintrittstests Ajax zum Einsatz kommt, werden diese Requests über „Direct Web Remoting“ abgehandelt.

Als Webserver wird Apache Tomcat 5.5[6] verwendet, der als Web-Container für Java-Code fungiert.

[4] JavaServer Pages bieten die Möglichkeit dynamische Webseiten zu erstellen [jsp, 2007]
[5] Velocity bietet eine einfache aber mächtige Template-Sprache um Java-Objekte zu referenzieren [vel, 2007]
[6] Apache Tomcat ist ein Servlet-Container [tom, 2007]

5.2.2 Ajax

Ajax bietet die Möglichkeit einen Browser mit dem Server kommunizieren zu lassen, ohne dass die Seite neu geladen werden muss. Dabei werden die Anfragen direkt von JavaScript aus getätigt und verarbeitet. Der Begriff Ajax wurde als Abkürzung für „Asynchronous JavaScript + XML“ bekannt. Im Internet wurde über diesen Term rege diskutiert, wobei viele der Meinung waren, dass es sich nicht um ein Akronym handle, da neben XML noch andere Technologien zum Einsatz kommen können [Shiflett, 2007]. Daher wird in dieser Arbeit der Begriff nicht in Grossbuchstaben („AJAX“) geschrieben.

Die verwendete Ajax-Technologie wird durch die ExtJS-Bibliothek, DWR und das Transfer-Format JSON unterstützt (siehe unten).

5.2.3 Direct Web Remoting (DWR)

Damit die Kommunikation von JavaScript zum Server simpler zu handhaben ist, wird, wie oben beschrieben, Direct Web Remoting (DWR) eingesetzt. Auf der Webseite von DWR wird das Produkt folgendermassen angepriesen: „Easy Ajax for Java. Export your Java code to a browser and include the results in your pages“ [dwr, 2007]. DWR kann sehr schnell und unkompliziert aufgesetzt werden und erleichtert die Arbeit mit Ajax extrem, da man sich nicht mehr um den eigentlichen Request kümmern muss. Damit DWR funktioniert, werden während der Laufzeit dynamisch JavaScript-Dateien generiert, welche den Code mit den Ajax-Requests beinhalten. Als Rückgabe-Wert der Aufrufe auf der Client-Seite können Java-Objekte, die von dem Server erstellt werden, dienen. So ist es zum Beispiel möglich Java-Beans automatisch in JSON (siehe Abschnitt 5.2.7) zu konvertieren und damit leicht auf der Client-Seite arbeiten zu können. Um den Java-Code mit JavaScript zu verbinden, wird eine XML-Datei verwendet. In dieser Datei kann auch eine White- oder Black-List definiert werden, die besagt, welche Methodenaufrufe erlaubt bzw. unerlaubt sind. Da die Anfragen nicht mehr selbst geschrieben werden müssen, entsteht das Gefühl, dass man direkt von JavaScript aus Java-Methoden aufrufen kann, was die Entwicklung mit DWR sehr komfortabel macht.

5.2.4 JAXB-Framework

Da auf den QTI-Standard gesetzt wird, müssen die Items in dem XML-Dateiformat gespeichert werden. Dabei muss das von dem „IMS Global Learning Consortium“ erstellte XML-Schema eingehalten werden. Um das XML-Schema korrekt umzusetzen, wird eine Technologie zur XML-Daten-Bindung genutzt. Ein performantes Framework hierfür ist JAXB (Java Architecture for XML Binding), welches aus einem XML-Schema Java-Code erzeugt [jax, 2007]. Das XML-Schema zum QTI-Standard ist auf der offiziellen QTI-Webseite zu finden und kann mit dem Bindingkompilierer xjc in Javaklassen kompiliert werden. Mit diesen Klassen kann nun auf bequeme Art und Weise ein dem XML-Schema konformes Dokument erstellt werden.

Im Falle einer zukünftigen vollumfänglichen Implementierung des QTI-Standards ist mit JAXB und Xindice (siehe unten) bereits ein grosser Schritt getan, da die technischen Rahmenbedingungen hiermit gesetzt wurden.

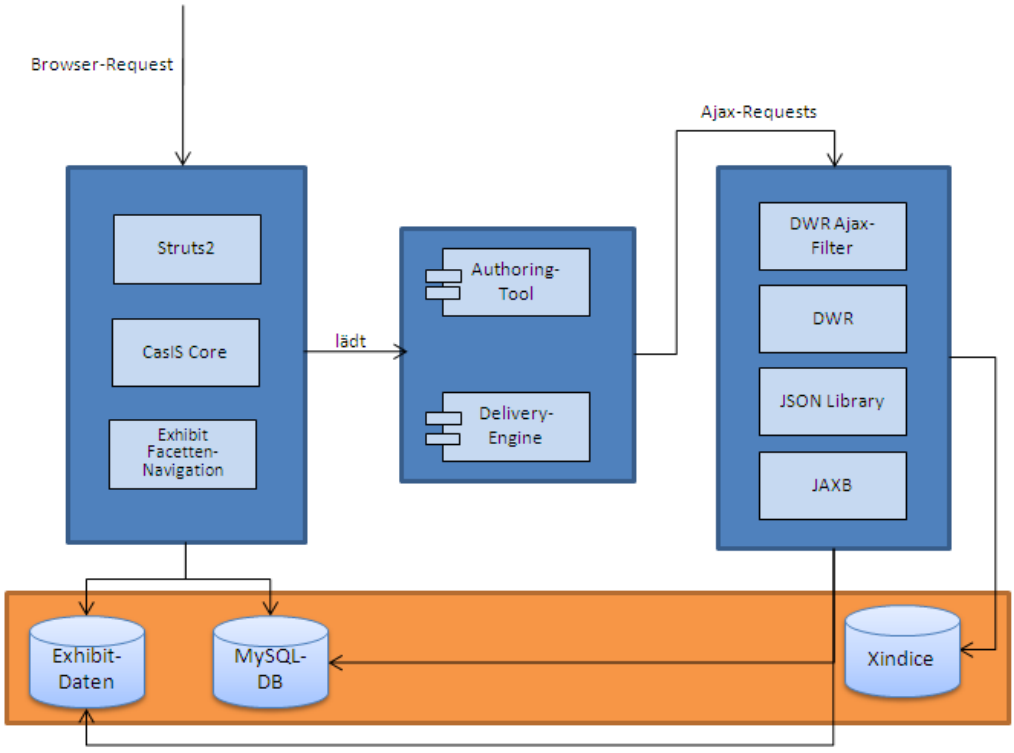

Abbildung 5.2: Architektur

5.2.5 Xindice

Da in jedem Test für jede Frage ein XML-Dokument erstellt werden muss, kann davon ausgegangen werden, dass langfristig gesehen sehr viele Dateien entstehen. Um mit den XML-Daten effektiv arbeiten zu können, wird auf Xindice, eine native XML-Datenbank, zurückgegriffen. Xindice wurde von der „Apache Software Foundation“ entwickelt und bietet die Möglichkeit XML-Dokumente zu speichern und zu manipulieren. Apache Xindice speichert diese Dokumente in „Collections“ und unterstützt die Ausführung von XPath- und XUpdate-Anfragen. Ausserdem können Indizes erstellt werden, um die Performanz zu steigern [xin, 2007]. In dieser Arbeit wurde eine native XML-Datenbank gewählt, damit die XML-Dokumente nicht andauernd in eine relationale Struktur umgewandelt werden müssen.

5.2.6 Die Ajax-Bibliothek ExtJS

Um die Benutzerschnittstelle optimal zu gestalten, wird Ajax eingesetzt. Mittlerweile gibt es sehr viele verschiedene Ajax-Frameworks um Requests zu tätigen oder dynamische UI-Elemente einfach einzufügen. In dieser Arbeit wird die Bibliothek ExtJS verwendet, welche zuerst bloss eine Erweiterung der YUI-Library[7] war. ExtJS ist weit verbreitet und wird von Firmen wie IBM oder Adobe auf ihren Webseiten benützt [ext, 2006]. Die Entscheidung fiel auf ExtJS, weil es eine sehr umfangreiche Tree-View-Komponente besitzt, welche ein grosser Bestandteil der Benutzerschnittstelle des Authoring-Tools darstellt. Die JavaScript-Bibliothek Dojo[8] bietet zwar auch eine solche Komponente an, diese scheint aber noch nicht allzu ausgereift zu sein und kann optisch auf keinste Weise mit der von ExtJS mithalten. Ausserdem können auch sehr einfach Layout- und Formularelemente eingefügt werden, welche ebenfalls im Authoring-Tool Verwendung finden.

5.2.7 JSON

„JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate“ [jso, 2007]. JSON kann statt XML für den Datenaustausch zwischen Client und Server verwendet werden. Dieses Format ist aber im Gegensatz zu XML kompakter, sodass weniger Last transferiert werden muss. Ein grosser Vorteil besteht ausserdem darin, dass JavaScript das Format nativ versteht und somit auf der Client-Seite kein Parsen notwendig ist. Um JSON in Java zu parsen und zu generieren wird die offizielle Bibliothek von json.org benützt. In dieser Arbeit wird dieses Format erstens aus Performanzgründen und zweitens aufgrund von DWR verwendet. DWR bietet wie bereits beschrieben die Möglichkeit Java-Beans in JSON zu konvertieren.

5.2.8 Exhibit

Exhibit ist ein vom MIT entwickeltes Tool, um Daten zu präsentieren. Dabei hat der Benutzer Filter- und Sortieroptionen um die Informationen so darzustellen, wie er sie benötigt. Ex-

[7] YUI ist eine von Yahoo! entwickelte JavaScript-Bibliothek [yui, 2007]
[8] http://dojotoolkit.org/

hibit basiert auf JavaScript und verwendet eine oder mehrere JSON-Dateien um Informationen zu speichern. Um die Daten in der JSON-Datei anzuzeigen, muss eine HTML-Seite erstellt werden, welche anhand von CSS designtechnisch verändert werden kann [exh, 2007]. Exhibit wird im CasIS-Portal überall dort verwendet, wo Daten angezeigt und sortiert werden können. In Abschnitt 5.4.2 wird genauer auf die Verwendung von Exhibit in dieser Arbeit eingegangen.

5.3 Authoring-Tool

Wie in Abschnitt 5.1.5 beschrieben, ist ein Authoring-Tool für die Erstellung und Editierung von Items verantwortlich. Das Authoring-Tool in dieser Arbeit ist aber zusätzlich in der Lage, aus diesen Items Tests zu erstellen.

5.3.1 Layout

Wie schon im Abschnitt 5.2 besprochen, wird für die Benutzerschnittstelle die ExtJS-Library verwendet. Das Authoring-Tool sollte vom Aussehen und Gefühl her möglichst nah an eine Desktop-Applikation kommen, da eine simple Webseite den gewünschten Funktionsumfang nur sehr schwierig abdecken könnte. Es gibt genug Beispiele (unter anderem Google Docs[9] oder Zoho Writer[10]) die zeigen, dass webbasierte Applikationen sich nicht mehr gross von Desktop-Anwendung unterscheiden müssen. Ein plausibler Grund Ajax zu verwenden ist in [Mahemoff, 2007] folgendermassen beschrieben: „Ajax aims to keep the benefits of the Web, but without sacrificing usability“.

Die ExtJS-Bibliothek bietet eine wunderbare Möglichkeit, um auf einfache Weise ein ansehnliches Layout zu gestalten. Im HTML-Code müssen lediglich -Elemente mit einer ID hinzugefügt werden, welche ExtJS über JavaScript den Bereichen Norden, Süden, Westen, Osten und Zentrum zuweist. In dieser Arbeit ist das Layout so aufgeteilt, dass auf der linken Seite ein Container mit einer Tree-View-Komponente und rechts ein Formular mit den Eigenschaften der Tests und Items ersichtlich ist. Die Baumstruktur dient der Übersicht aller Tests und deren Sektionen und Items. Es werden nur die Tests dargestellt, die auch dem angemeldeten Administrator gehören.

Da jeder Test, jede Sektion und jedes Item Eigenschaften haben, die editierbar sein sollen, eignet sich dieses Layout optimal. Auf der linken Seite ist die überschaubare Tree-View-Komponente, mit der die einzelnen Testelemente ausgewählt werden können und rechts werden die Eigenschaften des Ausgewählten über gängige UI-Elemente wie Textfelder oder Dropdown-Menüs bearbeitet. Neben dem Editierbereich rechts existieren noch andere Möglichkeiten einen Test zu editieren. Die Reihenfolge der Items oder Sektionen kann auf simpelste Weise mit „Drag and Drop“ verändert werden. Um Elemente zu erstellen oder zu löschen, steht ausserdem auf jedem Element ein Kontextmenü zur Verfügung, welches über die rechte Maustaste aufgerufen werden kann.

Ausserdem wird dasselbe Farbschema wie im gesamten CasIS-Portal verwendet, um ein einheitliches Auftreten zu erzielen.

[9] http://docs.google.com/
[10] http://writer.zoho.com/

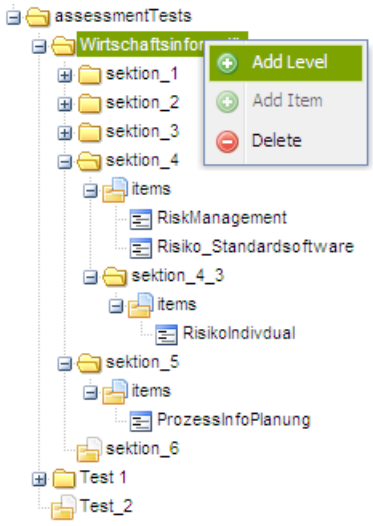

Abbildung 5.3: Tree-View-Komponente des Authoring-Tools

5.3.2 Tree View

Die Information für die Baumstruktur wird über das Datenaustauschformat JSON geladen. Bei jedem Klick auf einen Knoten wird dabei ein Ajax-Request ausgeführt, welchen der Server entgegennimmt und dabei eine Anfrage auf die Xindice-Datenbank tätigt. Anhand der Informationen dieser Anfragen wird eine JSON-Antwort generiert.

In der Tree-View-Komponente wird die in Kapitel 4 definierte Teststruktur abgebildet. Dies bedeutet, dass Tests in Sektionen verschiedener Granularitäten gegliedert werden. Jede Sektion kann zusätzlich als Elternknoten einer anderen Sektion dienen, sodass Granularitätshierarchien definiert werden können. Wenn eine Sektion Items aufweist, dann erscheint zusätzlich ein Ordner mit dem Namen „items“. Dieser Ordner wird auf der gleichen Ebene wie die Sektionen feinerer Granularität angezeigt. Für eine bildliche Vorstellung des eben Erklärten dient Abbildung 5.3. Auf dem Bild ist der Test „Wirtschaftsinformatik“ mit mehreren Sektionen sichtbar. „sektion_4“ besitzt neben eigenen Items noch eine Sektion feinerer Granularität, was bedeutet, dass auf deren Items Vorbedingungen gesetzt werden können. Die feinere Granularität ist stets ein Kindknoten der gröberen Granularität. Die Tiefe der Granularitäten kann theoretisch unendlich sein, was bedeutet, dass dem Autor volle Flexibilität gewährleistet wird.

Um die verschiedenen Testelemente zu unterscheiden, werden verschiedene Symbole verwendet. Eine Sektion, welche Items oder andere Sektionen beinhaltet, wird mit einem Ordner gekennzeichnet. Eine noch leere Sektion besitzt als Icon einen Ordner mit einem weissen Dokument darin. Für den Item-Ordner wird ein ähnliches Symbol verwendet, wie bei einer leeren Sektion, allerdings ist das abgebildete Dokument blau, was signalisieren soll, dass es Elemente darin hat. Die einzelnen Items selbst unterscheiden sich optisch von den anderen Knoten darin, dass sie als Symbol keinen Ordner aufweisen.

Für den „Drag and Drop“-Mechanismus und das Kontextmenü müssen mehrere Event-Handler registriert werden. Da beim „Drag and Drop“ nicht jeder Knoten als Drop-Ziel dient und dies schon während dem Ziehen eines Knotens dem Benutzer signalisiert werden soll, wird während der Aktion andauernd geprüft, ob ein Element über einer zuläs-

sigen Drop-Zone ist. Ausserdem muss vor dem effektiven Drop eine Anfrage an den Server ausgeführt werden, welche die Reihenfolge der Items oder der Sektionen in der XML-Datenbank aktualisiert. Nur wenn die Anfrage erfolgreich ausgeführt wurde, darf der Knoten endgültig fallen gelassen werden. Die „Drag and Drop"-Aktion ist nur bei Items der selben Sektion oder bei Sektionen des gleichen Elternknotens zulässig. Dies wurde so eingeschränkt, dass ein Benutzer nicht aus Versehen sehr komplexe Strukturen erzeugt und somit den Überblick über seinen Test verliert.

Für das Kontextmenü wird ebenfalls ein Event-Handler registriert, der bei einem Rechtsklick auf einen Knoten reagiert.[11] Das Kontextmenü soll das einfache Erstellen oder Löschen von Testelementen ermöglichen. Abhängig von dem geklicktem Knoten können entweder Tests, Sektionen oder Items erstellt werden. Um dies zu verdeutlichen: Beim Klick auf den Wurzelknoten können nur Tests erzeugt werden, da jede Sektion einem Test zugehörig ist und somit das Erstellen einer solchen von dem Wurzelknoten unmöglich ist. Beim Rechtsklick auf eine Sektion hingegen können entweder weitere Sektionen oder Items hinzugefügt werden.

Bevor der Knoten erzeugt wird, erscheint ein Dialogfenster, das nachfragt, wie das neue Testelement heissen soll. Erst wenn ein Name eingegeben und bestätigt wurde, wird eine Anfrage an den Server gesendet. Bei der Erzeugung von Sektionen feinerer Granularitäten, werden automatisch Vorbedingungen bezüglich der korrekten Beantwortung der Items der gröberen Granularität gesetzt.

Beim Löschen eines Elements wird der Autor ebenfalls durch einen Dialog zur Bestätigung aufgefordert.

5.3.3 Editierbereich

Der Editierbereich ermöglicht die detailliertere Bearbeitung von Testelementen, das Verwalten von Vorbedingungen und die Zuweisung von Vorbereitungsmodulen. Diese drei Bearbeitungsmöglichkeiten (Sichten) sind als Toolbar-Schaltflächen im Editierfenster angezeigt (siehe Abbildung 5.4). Die Darstellung des Editierbereichs ist von dem geklickten Knoten in der Tree-View-Komponente und der aktuellen Sicht abhängig.

Bearbeitung von Testelementen: Wenn es sich beim ausgewählten Knoten um einen Test oder eine Sektion handelt, können, wie in Abbildung 5.4 ersichtlich, Titel, Autorenname, Fallstudie und eine Beschreibung eingegeben werden. Im Falle mehrerer Benutzer mit dem selben Admin-Zugang, kann der Autorenname als Unterscheidungsmerkmal herangezogen werden. Durch den Namen wird also klar, welcher Benutzer den Test tatsächlich erstellt hat. Die Zuweisung einer Fallstudie geschieht über ein Dropdown-Menü, welches alle möglichen Module auflistet.

Bei einer Sektion kann lediglich der Titel editiert werden. Dies dient in erster Linie dem Autor, der über die Angabe sinnvoller Titel eine bessere Übersicht über seine Testelemente bekommen kann.

Abbildung 5.5 macht ersichtlich, dass ein Item über mehr editierbare Eigenschaften verfügt als ein Test. Hierzu gehört neben dem Titel, die Gewichtung, die in Kapitel 4 angesprochen wurde. Ausserdem müssen selbstverständlich Fragen und Antworten eingegeben werden

[11] Bei gewissen Browsern wird der „oncontextmenu"-Event verhindert, weshalb mit gedrückter CTRL-Taste und Linksklick das Kontextmenü ebenfalls ersichtlich wird.

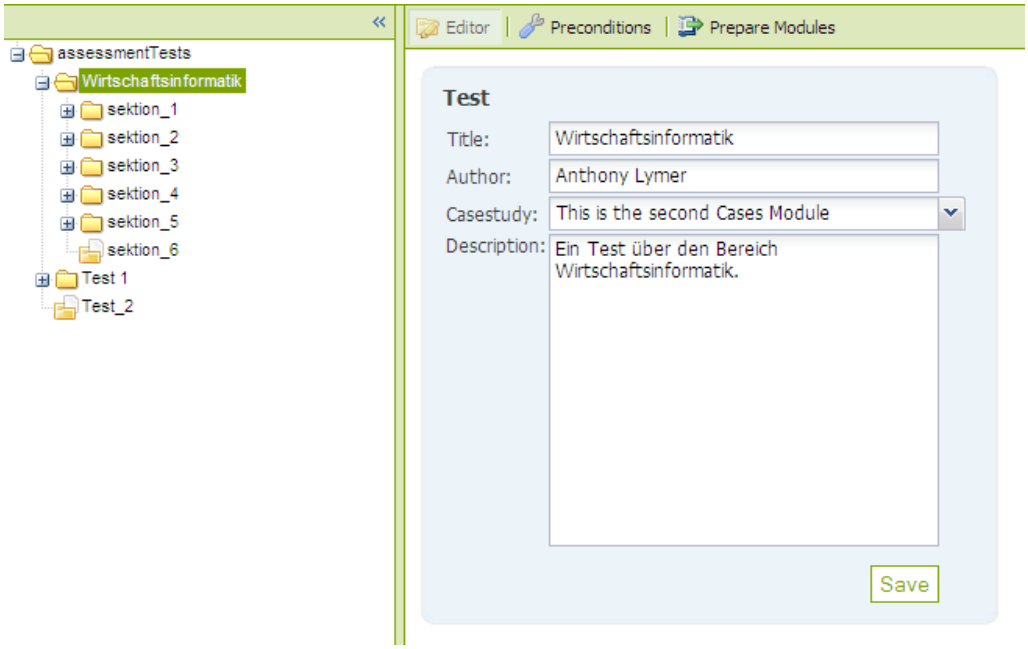

Abbildung 5.4: Bearbeitung eines Tests im Editor

können. Der Typ des Items kann über das Dropdown-Menü „Itemtype“ angegeben werden, wobei Single- und Multiple-Choice zur Auswahl stehen. Da der Autor selbst bestimmen kann, wie viele Antworten er möchte, können über intuitive Schaltflächen Antwortalternativen hinzugefügt und gelöscht werden. Um zu bestimmen, ob die jeweiligen Antworten korrekt sind, wird die Checkbox neben dem Textfeld verwendet. Bei einem Single-Choice-Item ist die Angabe von mehreren richtigen Antworten jedoch nicht möglich, da dies der Natur dieses Fragetyps widersprechen würde.

Alle Formulare werden erst gespeichert, wenn der Benutzer auf „Save“ klickt.

Verwalten von Vorbedingungen: Vorbedingungen können ausschliesslich Sektionen zugewiesen werden. Um eine Auswahl aller möglichen Vorbedingungen zu bekommen, werden die Items der Sektion mit der gröberen Granularität aufgelistet (Elternknoten). Nun können mit einem Klick die gewünschten Items, die als Vorbedingung dienen sollen, ausgewählt werden. Um dem Benutzer einen gewissen Komfort zu bieten, können zudem über eine Schaltfläche alle Items selektiert bzw. deselektiert werden. Es muss aber zwingend mindestens ein Item ausgewählt werden, da eine Unterteilung in Sektionen verschiedener Granularitäten ohne Vorbedingung keinen grossen Sinn machen würde.

Zuweisung von Lernressourcen: Wie in Kapitel 4 bereits genauer erläutert, sollen gewisse Wissenslücken über die FOIS-Module gefüllt werden. Diese Lernressourcen können jeder Sektion über eine Auflistung zugewiesen werden. Dabei handelt es sich um zwei Dropdown-Menüs, wobei das zweite abhängig vom Inhalt des ersten ist. Das eine Menü listet alle Vorbereitungsmodule auf und das andere gliedert die ausgewählten Module in einzelne Lerneinheiten. Dem Autor wird ausserdem die Möglichkeit eingeräumt, eine Feh-

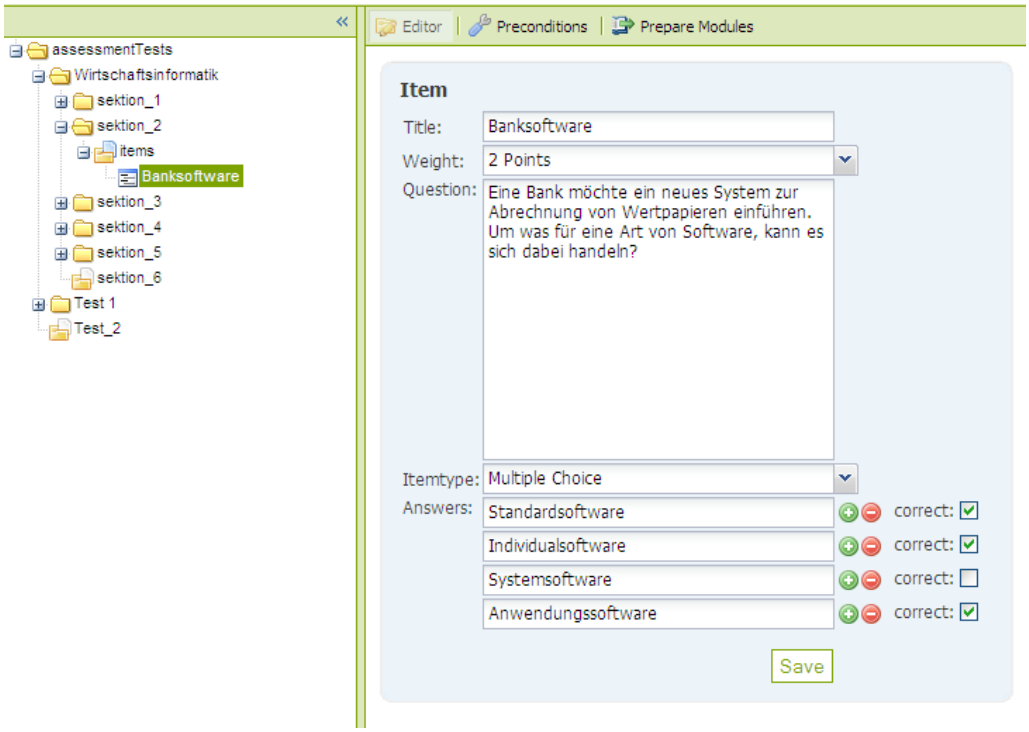

Abbildung 5.5: Bearbeitung eines Items im Editor

lerrate zu definieren. Diese bestimmt, ab welcher Anzahl falsch beantworteter Fragen das Vorbereitungsmodul angezeigt werden soll. Hierfür sind drei Standardwerte vorgesehen, die ebenfalls über ein Dropdown-Menü ausgewählt werden können.

5.3.4 Umsetzung

Wie schon erwähnt, wird Ajax in Verbindung mit JSON verwendet, da dies eine performante Lösung darstellt. Bei jeder Benutzerinteraktion, welche die Datenbank verändern soll, wird serverseitig Java-Code ausgeführt.

Auf der Serverseite wurden zwei Klassen erstellt, welche die Anfragen des Browsers entgegennehmen. Die Klassen wurden nach Funktionen aufgeteilt; so deckt eine Klasse alle Aktionen der Tree-View-Komponente ab und die andere alle Aktionen in dem Editierbereich. Beide verwenden das JAXB-Framework, welche das Marshalling bzw. Unmarshalling der XML-Dokumente durchführt. Um die Daten in die Datenbank zu speichern, wurde das Data-Access-Object-Entwurfsmuster (DAO) verwendet. Es besteht also ein Java-Interface, welches die Methoden vorgibt, die verwendet werden müssen. Die implementierenden Klassen können demnach beliebig ausgewechselt werden, sofern sie die Schnittstelle korrekt umsetzen. Das DAO-Pattern hat den Vorteil, dass damit die zugrunde liegende Datenbank einfacher geändert werden kann. Bei einem Datenbankwechsel muss bloss die implementierende Klasse getauscht werden, wobei die zwei oben beschriebenen Klassen gleich bleiben können. Das aktuell implementierte DAO verwaltet alle Zugriffe auf Xindice.

Neben den bereits erwähnten Klassen sind zudem Java-Beans im Einsatz. Wie in Abschnitt

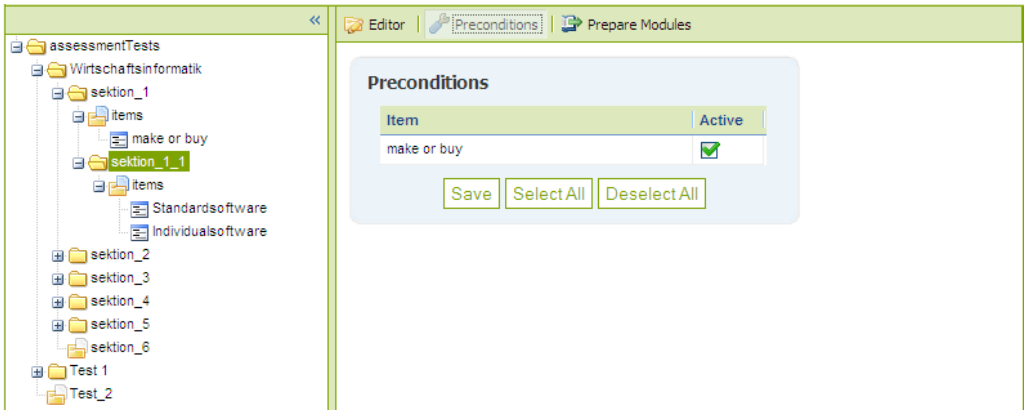

Abbildung 5.6: Definieren von Vorbedingungen

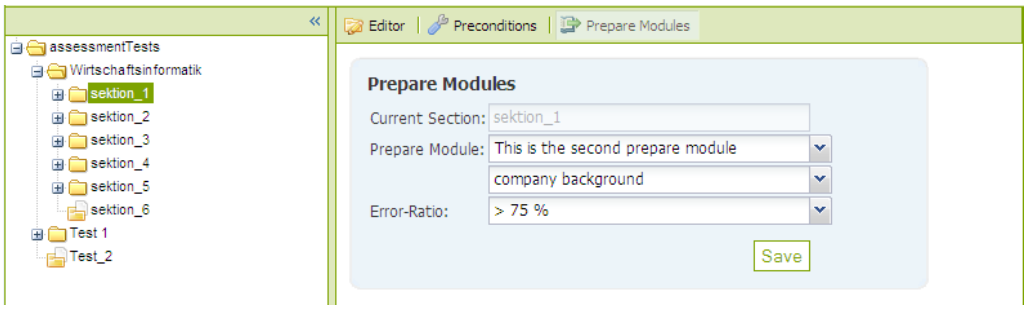

Abbildung 5.7: Zuweisung von Vorbereitungsmodulen

5.2.3 beschrieben, kann DWR Java-Beans in JSON umwandeln, sodass auf simpelste Art und Weise von JavaScript darauf zugegriffen werden kann. Bei Anfragen, die mehr Informationen transferieren, wie zum Beispiel ein ganzes Item, wurden Java-Beans erstellt. Bei den restlichen Ajax-Anfragen wird direkt ein JSON-String erzeugt, welchen der Browser ohne Konvertierung von DWR erhält.

DWR wurde so eingerichtet, dass jeweils die benötigten Methoden in eine White-List aufgenommen wurden. Vom Browser können somit nur zugelassene Funktionen aufgerufen werden.

Eine weitere Option von DWR ist das Filtern von Ajax-Requests. Da bei längerer Abwesenheit die Session ablaufen kann und der Benutzer demnach keine Änderungen mehr vornehmen darf, wird bei jedem Ajax-Aufruf geprüft, ob der Benutzer noch angemeldet ist. Mit DWR hat man die Möglichkeit in der Konfigurationsdatei eigene Ajax-Filter hinzuzufügen. Für das oben beschriebene Session-Problem wurde ein Filter entwickelt, der bei einer abgelaufenen Session eine Exception wirft. Obwohl dieser Code von der Serverseite ausgeführt wird, kann DWR von der Client-Seite die Exception abfangen und abwickeln. Dabei muss ein Error-Handler registriert werden, welcher bei jedem Fehler aufgerufen wird. Falls dieser Fehler eine Exception aufgrund der abgelaufenen Session ist, wird der Benutzer auf die Login-Seite des CasIS-Portals umgeleitet. Um den Benutzer aber nicht zu verwirren, wird vor der Weiterleitung ein kleiner Dialog mit einem Fortschrittsbalken angezeigt, der darauf hinweist, dass sich der Autor erneut einloggen muss.

Da die Initialisierung des JAXB-Framework eine Weile dauert, wurde geschaut, dass diese in dem Moment geschieht, bei dem es den Benutzer am wenigsten stört. Der beste Moment ist sicherlich beim Laden des gesamten Authoring-Tools. Während dem Starten wird eine Ajax-Anfrage durchgeführt, welche prüft, ob JAXB bereits initialisiert ist. Sollte dies nicht der Fall sein, wird, wie bei der Weiterleitung zur Login-Seite, ein Dialog mit einem Fortschrittsbalken angezeigt und JAXB geladen. Dieser Dialog wird aber nur in sehr seltenen Fällen aufgerufen, da das JAXB-Objekt als Singleton nur einmal erstellt wird und danach für alle Clients dasselbe ist.

Wie schon in Abschnitt 5.2.5 beschrieben, werden die Tests in einer nativen XML-Datenbank hinterlegt. Ein Test wird dabei in eine Collection gespeichert, welche wiederum eine Collection für alle Items beinhaltet. In den jeweiligen Collections befinden sich die XML-Dokumente.

Damit die einzelnen Items in der Datenbank eindeutig voneinander unterschieden werden können, wurden „Universal Unique Identifiers“ (UUID) eingesetzt. Eine UUID wird anhand eines Zeitstempels und der physikalischen Netzwerkadresse des erzeugenden Computers erstellt. Die Wahrscheinlichkeit einer Kollision geht gegen null, wobei davon ausgegangen werden kann, dass bis ins Jahre 3400 keine ID zweimal vorkommt. [P. Leach, 2005].

Ein wichtiger Bestandteil der Umsetzung war das Thema Sicherheit. Neben der in DWR definierten White-List wurde das Authoring-Tool auf Cross-Site-Scripting-Attacken (XSS) geprüft. Mit XSS lässt sich fremder JavaScript-Code auf einer Webseite ausführen, was unter Umständen gravierende Folgen haben kann. Eine Möglichkeit von Cross-Site-Scripting ist das Auslesen von Cookies einer anderen Person, was bei sensitiven Daten alles andere als erwünschenswert ist [Anh Nguyen-Tuong, 2005]. Auf das Authoring-Tool haben zwar nur Administratoren Zugriff, dennoch darf die Sicherheit nicht vernachlässigt werden. Falls ein Angreifer irgendwie Zugriff erlangen kann, so soll er möglichst wenig Schaden anrichten können. Um dies zu gewährleisten, werden alle HTML-Tags in HTML-Entities umgewandelt, damit bei der Anzeige von gespeicherten Daten z.B. <script> in <script> umgewandelt und somit vom Browser nicht als Script-Code interpretiert wird.

Auf die Gefahren von SQL-Injektionen wird später eingegangen, da dies erst bei der Einbindung des Authoring-Tools in das CasIS-Portal relevant ist.

5.4 Einbindung ins CasIS-Portal

Das CasIS-Portal ist in vier verschiedene Module aufgeteilt. Die Basis des Portals ist bereits programmiert und muss nun durch den Einbau des Eintrittstests erweitert werden. In Abbildung 5.8 sind die Module „Entry Test“ (Eintrittstests), „Preparation“ (Vorbereitungsmodule), „Cases“ (Fallstudien), „Toolbox“ (Werkzeuge) ersichtlich. Dabei wird wie folgt vorgegangen: Ein Benutzer gelangt auf das Portal und kann sich einen Test aussuchen. Nach Absolvierung des Tests klickt er auf „Prepare“ und gelangt zu den Vorbereitungsmodulen. Wenn er sich sicher genug fühlt, kann er über den Link „Cases“ die gewünschte Fallstudie bearbeiten. In der Toolbox befinden sich anschliessend entsprechende Werkzeuge, um die Fallstudie bequem lösen zu können.

Die Benutzer des Portals sind in zwei Rollen eingeteilt. Ein Rolleninhaber ist der User, der als Kandidat einen Test löst, danach sich vorbereitet und schliesslich die Fallstudie mit geeigneten Werkzeugen bearbeitet. Die andere Rolle bezieht sich auf den Administrator, der Tests erstellen kann und Lernmaterial anbietet.

Da ein Benutzer keine Tests erstellen soll, wird das Authoring-Tool einzig in dem Admin-Bereich angezeigt. Neben dem Tool selbst hat ein Administrator auch die Möglichkeit, eine Übersicht über alle Tests zu gewinnen. Für diese Funktion ist das in Abschnitt 5.2.8 beschriebene Exhibit optimal. Dank Exhibit ist das Sortieren oder Filtern nach bestimmten Kriterien kinderleicht und der Administrator kann sich sehr schnell einen Überblick über alle existierenden Tests verschaffen.

Für das Authoring-Tool bestand bereits ein Velocity-Template, welches nur noch mit dem richtigen Inhalt gefüllt werden musste. Um die Daten in Exhibit anzuzeigen, wurde die zugrunde liegende JSON-Datei bearbeitet und erweitert.

Für den eigentlichen Eintrittstest bietet Exhibit allerdings nicht die Funktionen, die notwendig sind. Der Test soll nicht lediglich Daten anzeigen, sondern muss zudem eine höhere Interaktivität - nämlich das Beantworten der Items - zulassen.

Um auf das Authoring-Tool zu gelangen, muss, wie in Abbildung 5.8 ersichtlich, die bereits vorhandene Navigation des CasIS-Portals verwendet werden. Bei einem Klick auf „Edit“ wird das in Abschnitt 5.3 beschriebene Tool geladen.

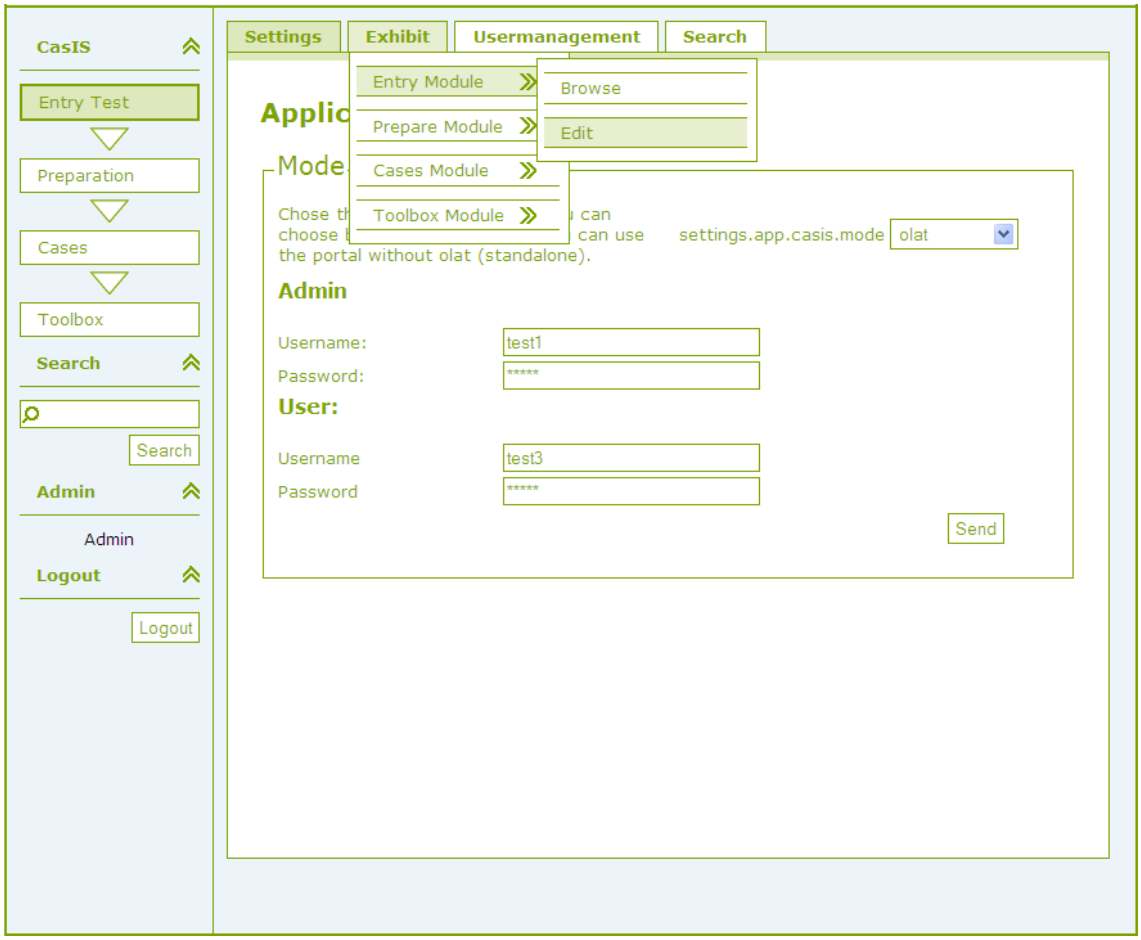

Abbildung 5.8: Navigation im CasIS-Portal um das Authoring-Tool zu starten

5.4.1 MySQL-Datenbank

Das CasIS-Portal besitzt bereits eine MySQL-Datenbank in der wichtige Userdaten wie Benutzername und Passwort gespeichert sind.

Wie in Kapitel 4 beschrieben, können den einzelnen Sektionen Vorbereitungsmodule zugewiesen werden, die abhängig von der Fehlerrate eines Kandidaten angezeigt werden. Um diese Verlinkung zu speichern, wird auf die MySQL-Datenbank zurückgegriffen, welches für dieses Unterfangen mit Tabellen erweitert wurde.

Ausserdem muss gespeichert werden, welcher Autor welchen Test erstellt hat und wie die Fallstudien damit verbunden sind. Obwohl diese Daten bereits in den JSON-Dateien von Exhibit vorhanden sind, werden sie zusätzlich in der MySQL-Datenbank abgelegt. Diese Entscheidung gründet vor allem auf der sehr bequemen Anfragesprache von MySQL. Zudem kann somit bei häufigen Zugriffen und grösseren Datenmengen ein Performanzvorteil resultieren.

Wie für die Datenbank-Anfragen der Xindice-Datenbank wird auch für die Kommunikation zu MySQL das DAO-Pattern verwendet.

Mit der Verwendung einer SQL-Datenbank entstehen gewisse Sicherheitsrisiken, welche fatale Folgen haben können. Anhand von SQL-Injektionen kann schädlicher SQL-Code als Parameter für eine Anfrage verwendet werden. Wenn bei folgendem Code der Parameter entrytestId so manipuliert wird, dass zum Beispiel „' 1'; DROP TABLE entrytest_casestudy_linker;“ an den Server geschickt wird, wäre die Folge, dass die Tabelle entrytest_casestudy_linker gelöscht würde.

```
String entrytestId =((String[]) parameters.get("entrytestId"))[0];
String sql = "SELECT casestudy_id" +
             "FROM entrytest_casestudy_linker" +
             "WHERE entrytest_id = " + entrytestId;
```

Um diese Angriffe zu verhindern, können „Prepared Statements“ verwendet werden, welche die einzelnen Parameter zuerst prüfen, bevor die Anfrage ausgeführt wird. Wie folgendes Zitat bekräftigt, kann die Datenbank-Logik von den übergebenden Daten getrennt werden. „Prepared statements can help increase security by separating SQL logic from the data being supplied“ [Fisk, 2007].

5.4.2 Exhibit

Exhibit wird im CasIS-Portal überall für die Präsentation von sortierbaren Daten verwendet. Das Tool kommt in allen vier CasIS-Modulen zum Einsatz, um eine klare Übersicht zu gewährleisten. Um eine Vorstellung von Exhibit zu bekommen, dienen die Abbildungen 5.9 und 5.10. Auf diese Seite gelangt man, wenn auf Browse geklickt wird (Abbildung 5.8). Es gibt zwei verschiedene Ansichten; die eine zeigt mit weniger Information tabellarisch alle Tests an, während die andere eine Detail-Sicht bietet. In der Detail-Sicht sind, wie Abbildung 5.10 zeigt, zusätzlich noch eine Beschreibung und das Datum, wann der Test erstellt wurde, sichtbar.

Um die Daten zu sortieren, muss lediglich auf den Tabellenkopf geklickt werden. Die Filterungsfunktion wird in den zwei Boxen auf der rechten Seite angeboten. In Abbildung 5.10 zum Beispiel, wurde ein Filter auf die Fallstudien angewendet. Es werden deswegen nur

die Fallstudien mit dem Titel „This is the second Cases Module“ angezeigt. Durch diese Funktion können also schnell Informationen über einen bestimmten Test gewonnen werden. Neben dem Filtern auf Fallstudien kann zusätzlich nach Autoren gefiltert werden. Die beiden Optionen lassen sich theoretisch auch vereinen.

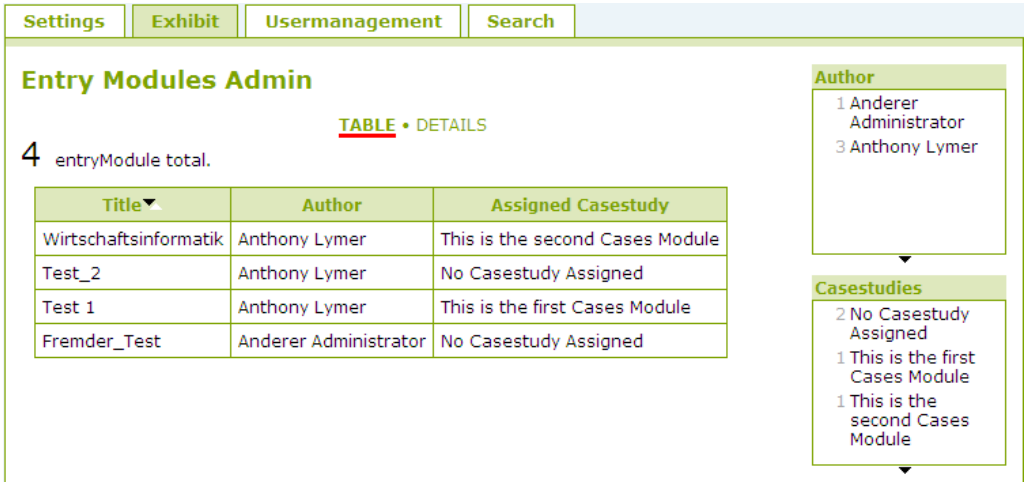

Abbildung 5.9: Exhibit in der tabellarischen Sicht

Abbildung 5.10: Exhibit in der detaillierten Sicht

Das von Exhibit verwendete Datenmodell basiert auf JSON, weshalb alle relevanten Daten in JavaScript gespeichert sind. Die einzelnen Tests befinden sich dabei jeweils in einem Array.

Das Design der präsentierenden Seite wird über „Cascading Style Sheets“ (CSS[12]) manipuliert. Die hier benützte CSS-Datei wurde bereits mit dem CasIS-Portal definiert. Es musste also lediglich festgelegt werden, welche Daten präsentiert werden sollen.

In den Auflistungen, die in den Abbildungen 5.9 und 5.10 ersichtlich sind, werden nicht nur die eigenen Tests angezeigt, sondern auch die Tests der anderen Administratoren. Dies hat

[12]http://www.w3.org/Style/CSS/

den Vorteil, dass festgestellt werden kann, ob ein Autor bereits einen Test für eine Fallstudie entwickelt hat.

Für Exhibt existierte bereits ein eigenes DAO, welches mit den benötigten Methoden erweitert wurde.

5.4.3 Auswählen eines Tests

Im CasIS-Portal kann ein Administrator auswählen, welche Module einem Benutzer angezeigt werden sollen. In dem Admin-Bereich gibt es die Möglichkeit die einzelnen Fallstudien auszuwählen, sodass sie für die Benutzer aufgeschaltet sind. Ein Test steht immer mit genau einer Fallstudie in Beziehung, wobei gleiches für die Gegenrichtung gilt. Das Aufschalten von Modulen beeinflusst alle CasIS-Benutzer gleichermassen: Es gibt keinen Benutzer, dem andere oder zusätzliche Eintrittstests bzw. Fallstudien präsentiert werden. Falls dies erwünscht wäre, müsste eine neue Instanz des Portals gestartet werden. Da jedem Test eine Fallstudie zugewiesen ist, genügt es, wenn diese selektiert wird. Der richtige Test wird dann automatisch für den Benutzer sichtbar.

5.5 Delivery-Engine

Ein weiterer wichtiger Teil dieser Arbeit ist die Entwicklung einer Delivery-Engine. Hierbei handelt es sich, wie in Abschnitt 5.1.5 beschrieben, um ein System, welches einen Kandidaten mit Fragen beliefert. Während dieses Vorgangs prüft es ausserdem ob Vorbedingungen erfüllt sind und handelt dementsprechend.

5.5.1 Ausgangspunkt Exhibit

Um auf einen Test zu gelangen, dient erneut die Navigation von Exhibit. Ein Kandidat gelangt nach der Anmeldung am System auf die Startseite und bekommt eine Übersicht über alle möglichen Tests. Diese werden ausschliesslich über die Detail-Sicht von Exhibit präsentiert. Der Beweggrund dahinter ist, dass nur wenige Daten präsentiert werden und eine Tabelle mit nur zwei Spalten optisch nicht gerade glänzt.

Nachdem sich der Kandidat für einen Test entschieden hat, wird über Struts eine neue Seite geladen. Bevor der Benutzer den Test zu Gesicht bekommt, werden alle zu liefernden Items in die MySQL-Datenbank gespeichert. Auf der Seite angekommen, kann der Benutzer nun die Fragen beantworten. Dem Kandidaten werden lediglich die Fragen und deren Antwortalternativen präsentiert. Er soll nicht mitbekommen, wie viele Punkte er bei einer Frage gerade gemacht hat oder was im Hintergrund exakt abläuft. Die Antworten können je nach Fragetyp über Checkboxen oder Radiobuttons ausgewählt werden. Checkboxen unterscheiden sich von Radiobuttons in der Hinsicht, dass mehrere Antworten angegeben werden können. Wenn bei einem Radiobutton neu selektiert wird, verschwindet automatisch die vorherige Selektion.

Über eine Schaltfläche können die Antworten abgegeben werden. Dabei wird sofort die nächste Frage geladen und dem Benutzer präsentiert. Nach der Beantwortung der letzten Frage wird darauf hingewiesen, dass der Test nun abgeschlossen ist und die Vorbereitung

auf die Fallstudien beginnen kann. Über die Navigation des CasIS-Portals besteht die Möglichkeit auf die zugewiesenen FOIS-Module zu gelangen.

Abbildung 5.11 zeigt ein Beispiel einer vom System ausgelieferten Frage.

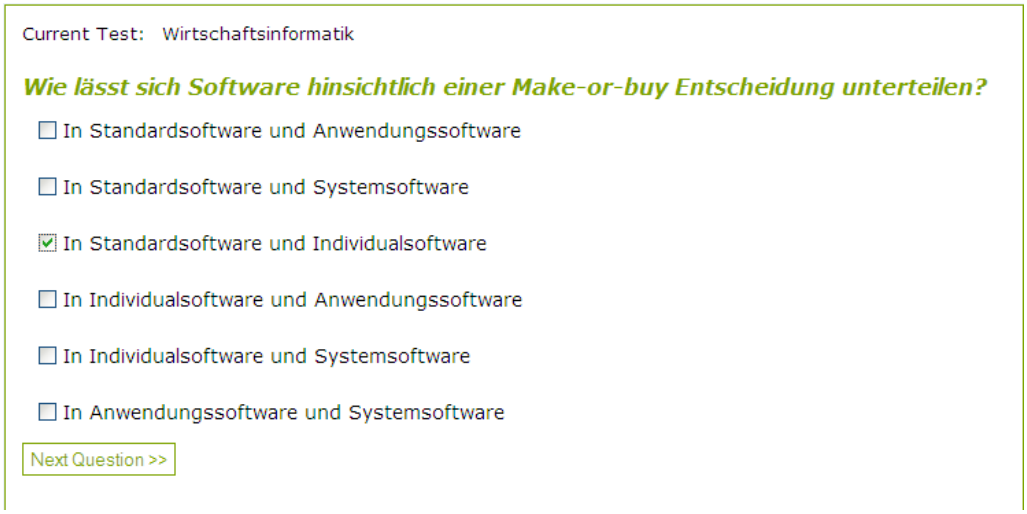

Abbildung 5.11: Die Delivery-Engine mit dem aktuell gelieferten Item

5.5.2 Hintergründe der Bewertung

Der Eintrittstest passt sich den gegebenen Antworten an. Erstens wird über Vorbedingungen geprüft, ob ein Item geliefert werden kann. Zweitens wird geprüft, ob bereits genügend inkorrekte Antworten des Benutzers gegeben wurden, um zu entscheiden, dass ein Vorbereitungsmodul empfohlen werden soll. Wie in Kapitel 4 erklärt, kann jeder Sektion ein Vorbereitungsmodul zugewiesen werden. Bei dieser Zuweisung wird zudem eine Fehlerrate bestimmt, ab welcher die Module empfohlen werden sollen. Der Algorithmus geht dabei folgendermassen vor: Die Fehlerrate des Benutzers wird bei jeder präsentierten Sektion initial als 100% angenommen. Bei einer korrekten Beantwortung von Fragen innerhalb dieser Sektion, nimmt die Fehlerrate ab. Wenn nun die Fehlerrate tiefer ist, als diejenige, die von dem Autor bestimmt wurde, kann in die nächste Sektion gesprungen werden, da der Benutzer bereits genügend Wissen bewiesen hat. Auf der anderen Seite wird geprüft, ob der Benutzer schon zu viele falsche Antworten abgegeben hat, dass er auch bei der korrekten Beantwortung von nachfolgenden Items nicht mehr unter die vom Autor definierte Fehlerrate kommen kann.

Die Fehlerrate eines Benutzers wird dabei, wie schon in Kapitel 4 erwähnt, folgendermassen berechnet:

$$1 - \frac{Aktuelle\ Punktzahl\ in\ entsprechender\ Sektion}{Maximal\ mögliche\ Punktzahl\ in\ entsprechender\ Sektion}$$

Durch dieses Vorgehen kann ein Test um mehrere Items verkürzt werden, sodass der Benutzer nicht Fragen beantworten muss, die keinen Einfluss mehr auf die Vorbereitungsmodule haben.

Die Evaluation der Vorbedingung geschieht ebenfalls jeweils vor der Auslieferung an den Kandidaten. Diese Bedingungen beinhalten dabei immer die korrekte Beantwortung der Items der gröberen Granularität. Falls eine Sektion eine Vorbedingung hat, wird also geprüft ob die Fragen, welche zur Erfüllung der Bedingung korrekt sein müssen, richtig beantwortet wurden.

5.5.3 Technische Umsetzung

Für die Benutzerschnittstelle kommt - wie beim Authoring-Tool - Ajax zum Einsatz. Einem Kandidaten wird somit eine angenehme Benutzeroberfläche geboten, die nicht nach jeder Frage neu geladen werden muss. Für die Kommunikation von Browser zu Server wird erneut auf DWR gesetzt, um die Ajax-Anfragen bequem zu verwalten. Wie in Abschnitt 5.3.4 angesprochen, wurden bereits Java-Beans erstellt, die DWR konvertieren kann. Da die Delivery-Engine auf dieselben Daten zurückgreift wie das Authoring-Tool, können die vorhandenen Beans erneut Anwendung finden.

Um die Fragen den Benutzern zu liefern und deren Antworten zu speichern, wird wiederum von der MySQL-Datenbank Gebrauch gemacht. Beim Starten eines Tests, müssen die zugehörigen Items in der Datenbank abgelegt werden. Diese Speicherung wird nötig, um einem Benutzer die Möglichkeit zu bieten, den Test zu einem späteren Zeitpunkt fortzuführen. Bei der Beantwortung der Fragen wird das eben gelieferte Item aus der Datenbank entfernt, sodass das nächste dem Benutzer präsentiert werden kann. Somit kann jederzeit - also auch nachdem sich der Benutzer einmal abgemeldet hat - das aktuelle Item wieder gefunden und geliefert werden. Ein Benutzer kann zu einem Zeitpunkt nur einem Test zugewiesen sein, weshalb nach der Testauswahl die Datenbank mit diesen Informationen erweitert wird. Ausserdem werden die Antworten des Benutzers gespeichert und bewertet, womit anschliessend die Fehlerrate für das Empfehlen der Vorbereitungsmodule berechnet wird.

Der Test endet, wenn kein einziges Item mehr in der Datenbank ist.

5.5.4 Vorbereitungsmodule

Alle existierenden Vorbereitungsmodule sind in einer JSON-Datei von Exhibit gespeichert. Da jeder Kandidat andere Testergebnisse erzielt, können nicht für jeden dieselben Daten gewählt werden. Damit jeder Benutzer andere Empfehlungen bekommt, wird deswegen aus der vorhandenen JSON-Datei dynamisch die passenden Vorbereitungsmodule geladen. Exhibit verwendet also als Input ein von der Quelldatei verändertes JSON-Array um die Informationen anzuzeigen.

Der Benutzer bekommt über ein weiteres Attribut signalisiert, ob ihm ein gewisses Modul empfohlen wird. Bei der Generierung des Arrays wird bei jedem Modul geprüft, ob die vom Autor definierte Fehlerrate höher ist, als die des Benutzers. In einem positiven Fall wird dabei jeweils das Attribut gesetzt, das beschreibt ob der Benutzer ein Vorbereitungsmodul benötigt.

Dieses gesetzte Attribut ermöglicht in Exhibit die Filterung und Sortierung nach empfohlenen bzw. nicht empfohlenen Vorbereitungsmodulen. Dem Benutzer wird somit schnell ersichtlich, welche Module für ihn von hohem Nutzen sind und welche er als zweitrangig beachten kann.

Um sich ein Vorbereitungsmodul anzusehen, muss in die Detail-Sicht von Exhibit gewechselt werden, auf der sich ein Link zu der entsprechenden Lerneinheit befindet. Mit einem Klick auf den Link wird das gewünschte Lernmaterial angezeigt. Damit der Benutzer trotzdem auf dem CasIS-Portal bleiben kann, wird das FOIS-Modul in einem neuen Fenster geladen.

Wenn ein Benutzer sich abmeldet und zu einem späteren Zeitpunkt seine Empfehlungen erneut anschauen möchte, ist dies problemlos möglich. Weil dem Benutzer jeweils nur ein Test zugeordnet ist und die beantworteten Fragen gespeichert bleiben, können die zuletzt empfohlenen Vorbereitungsmodule umgehend wieder angeschaut werden. Falls sich der Benutzer erneut bei dem Test versuchen will, werden die alten Informationen über die Beantwortung gelöscht, womit neue Empfehlungen erstellt werden können.

Bei einem Testabbruch seitens des Benutzers, werden ihm nur die Vorbereitungsmodule aufgelistet, bei denen schon Klarheit über sein Können herrscht. Angenommen ein Kandidat absolviert einen Test mit mehreren Sektionen, er beantwortet aber bloss eine und lässt die anderen aus. Wenn er sich dann die Vorbereitungsmodule anschauen will, wird ihm bloss eines aufgelistet, da über die Fähigkeiten bezüglich Themen der anderen Sektionen nichts bekannt ist.

Nachdem die Vorbereitungsmodule durchgearbeitet wurden, steht dem Lösen der Fallstudie nichts mehr im Wege. Wie in Abbildung 5.8 ersichtlich genügt ein Klick, um auf die Fallstudie zu gelangen. Der Student, der nun alle zu bearbeitenden Fallstudien sieht, kann sich nun die gewünschte aussuchen und ist mit den, vom Eintrittstest empfohlenen, Vorbereitungsmodulen optimal auf die Fallstudie vorbereitet. Über entsprechende Werkzeuge der Toolbox - das vierte Modul im CasIS-Portal - kann die Fallstudie nun mit dem erweiterten Hintergrundwissen souverän gelöst werden.

6

Ausblick

Falls die Administratoren des CasIS-Portals zusätzlich auf anderen Lernsystemen aktiv sein werden, wäre es vorstellbar, dass der Wunsch nach einer Exportfunktion aufkommt. Durch den verwendeten QTI-Standard steht einer solchen Funktion im Grunde genommen nichts im Wege. Die erstellten Tests müssten lediglich mit einem Standard gepackt werden, sodass sie in ein anderes System importiert werden können.

Der in dieser Arbeit entwickelte Eintrittstest umfasst lediglich Multiple-Choice und Single-Choice Fragetypen. Für komplexere Anforderungen müsste er also erweitert werden. Diese Erweiterungen würden unter anderem mit der vollumfänglichen Implementation von QTI erreicht. Da die technischen Rahmenbedingungen für ein solches Vorgehen bereits bestehen, wäre dieses Vorhaben problemlos durchführbar. Wenn QTI vollständig umgesetzt wird, wäre eine Funktion zur Importierung von Tests sinnvoll. Bis es aber soweit ist, lohnt sich eine Importfunktion nur halbwegs, da es wahrscheinlich ist, dass ein anderes Lernsystem zusätzliche Elemente verwendet.

Nachdem das CasIS-Portal evaluiert wurde, wäre es denkbar, dass noch kleinere Veränderungen vorkommen können. Eine Anpassung des Eintrittstests sollte aber kein grösseres Problem darstellen, da versucht wurde die Software in verschiedene kohärente Komponenten zu unterteilen, welche unabhängig von anderen Komponenten geändert werden können.

7

Schlussfolgerung

In dieser Arbeit wurde ein Eintrittstest für eine bereits bestehende Lernplattform entwickelt. Hier sollen nun die wichtigsten Erkenntnisse aufgelistet werden.

Technologie: Mit DWR wird die Programmierung mit der Ajax-Technologie ungemein vereinfacht, wobei die ExtJS-Library ein sehr ansehnliches und umfangreiches Framework für solche Anwendungen bietet. Für ähnliche Arbeiten sind diese zwei Produkte also auf jeden Fall zu empfehlen. IMS QTI kann in gewissen Fällen sehr sinnvoll sein, weshalb es sich durchaus gelohnt hat die Tests auf diesem Standard aufzubauen.

Didaktische Überlegungen: Die Adaptivität des Tests bringt einen sehr grossen Vorteil mit sich. Obwohl die Adaptivität nicht auf komplexen Wahrscheinlichkeitsrechnungen beruht, konnten plausible Möglichkeiten gefunden werden, um einen adaptiven Test umzusetzen. Die Testfragen werden dadurch optimal auf die heterogenen Kandidaten zugeschnitten, was ihnen ein positives Testerlebnis bieten soll.

Benutzerschnittstelle: Da ein Autor bei der Erstellung eines Tests durchaus komplexe Strukturen bilden kann, wurde mit der intuitiven Benutzerschnittstelle dank ExtJS eine schöne Lösung gefunden.

Den in Kapitel 2 definierten Anforderungen ist Rechnung getragen worden; so wurde dank ExtJS ein ansehnliches Authoring-Tool entwickelt und die Delivery-Engine fliessend in das CasIS-Portal eingebaut.

A

Anleitung zur Erstellung eines Eintrittstests

In diesem Teil der Arbeit wird das Authoring-Tool und dessen einzelne Funktionen genau beschrieben. Es wird detailliert aufgezeigt, wie es zu benützen ist und auf was geachtet werden muss. Dabei soll es als Anleitung und Stütze bei allfälligen Problemstellungen dienen. Ausserdem wird gezeigt, wie ein erstellter Test in dem CasIS-Portal aufgeschaltet werden kann.

A.1 Das CasIS-Portal

In diesem Abschnitt wird kurz auf das gesamte CasIS-Portal eingegangen. Dabei werden die Funktionen anhand von Abbildungen beschrieben.

A.1.1 Benutzeranmeldung

Nachdem Sie die Internetadresse des CasIS-Portals eingegeben haben, erscheint eine Anmeldemaske (Abbildung A.1). Um auf das CasIS-Portal zu gelangen, müssen Sie sich zuerst bei dem System mit ihren Zugangsdaten anmelden.

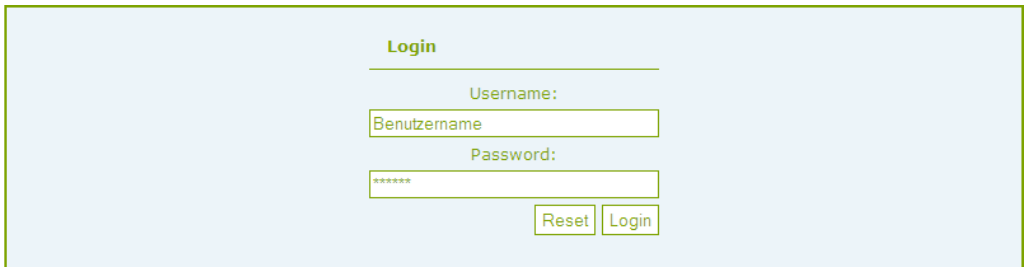

Abbildung A.1: Die Anmeldemaske des CasIS-Portals

A.1.2 Überblick über das Portal

Das CasIS-Portal beinhaltet vier Module. In Abbildung A.2 wurden diese mit der Nummer 1 markiert. Diese Module werden einem normalen Benutzer angezeigt, nachdem er auf der Login-Seite seine Daten eingegeben hat. Dabei werden automatisch die aufgeschalteten Tests präsentiert (siehe Abschnitt A.3.1).

Wenn Sie als Administrator angemeldet sind, haben Sie zusätzlich die Möglichkeit in den Administratorenbereich zu gelangen. Dafür müssen Sie auf den mit 2 angeschriebenen Hyperlink klicken.

Um sich aus dem CasIS-Portal abzumelden, können Sie die mit der Nummer 3 beschrifteten Schaltfläche betätigen.

Abbildung A.2: Die Startseite des CasIS-Portals

A.2 Authoring-Tool

Dieser Abschnitt erklärt alle Funktionen des Authoring-Tools und beschreibt, wie sie angewendet werden.

A.2.1 Starten des Authoring-Tools

Um das Authoring-Tool zu starten, gehen Sie auf den in Abbildung A.3 mit 1 beschrifteten Bereich. Nachdem Sie mit dem Mauszeiger dort angelangt sind, öffnet sich ein kleines Menü. Wenn Sie nun auf „EntryTest“ gehen und anschliessend auf „Browse“ (Bereich 2 in Abbildung A.3) drücken, wird Ihnen eine Übersicht über alle erstellten Tests geboten (siehe

Abbildung A.4). Hierbei werden auch die Tests der anderen Autoren angezeigt. Dadurch können Sie prüfen, ob bereits ein anderer Administrator einen Test für eine bestimmte Fallstudie geschrieben hat. In dieser Übersicht haben Sie die Möglichkeit die Informationen tabellarisch oder im Detail anzeigen zu lassen. Ausserdem können Sie nach Fallstudien und Autoren filtern.

Um auf das Authoring-Tool zu gelangen, müssen Sie auf „Edit“ (Bereich 3 in Abbildung A.3) klicken. Nachdem sich die Seite neu geladen hat, sollten Sie das in Abbildung A.5 abgebildete Tool sehen. Nun können Sie damit anfangen Tests, Sektionen und Items zu erstellen, editieren oder zu löschen.

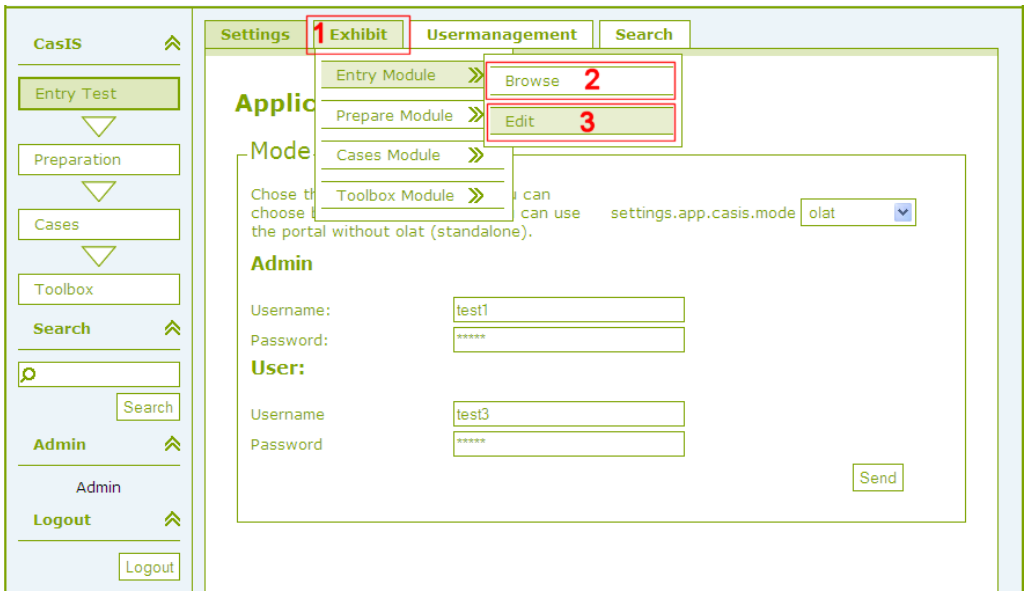

Abbildung A.3: Navigation um auf das Authoring-Tool zu gelangen

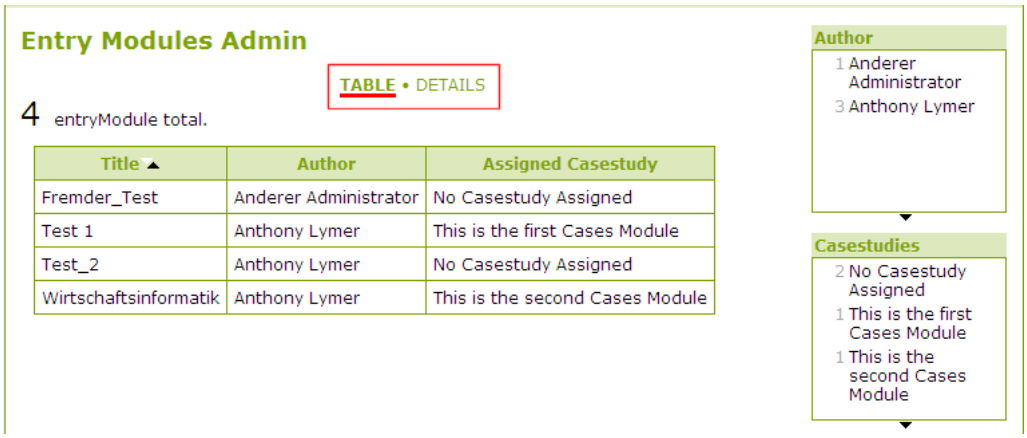

Abbildung A.4: Übersicht über alle Tests

A.2.2 Erklärung der Komponenten des Authoring-Tools

Das Authoring-Tool ist in drei Bereiche aufgeteilt. Die einzelnen Bereiche sind in Abbildung A.5 rot umrandet und nummeriert. Im Folgenden werden die Bereiche erklärt:

Bereich 1 (Tree-View): Der erste Bereich wird verwendet um eine Übersicht der erstellten Tests, Sektionen und Items zu bekommen. Die Tree-View-Komponente ist dabei so aufgebaut, dass zuerst Tests (siehe 1.1 in Abbildung A.5), dann Sektionen (siehe 1.2 in Abbildung A.5) und zuletzt Items oder weitere Sektionen aufgelistet werden (siehe 1.3 in Abbildung A.5). Die Möglichkeit Sektionen ineinander zu kapseln, gründet auf der Idee, dass verschiedene Granularitäten verwenden werden können. Hierbei beinhalten die Sektionen gröbster Granularität die feineren Sektionen, wobei Fragen, die ein detaillierteres Wissen prüfen, von feinerer Granularität sind. Um die Idee von Granularitäten näher zu beschreiben, kann Abbildung A.5 herangezogen werden. Darauf ist eine Sektion mit dem Item „make or buy“ und einer Untersektion ersichtlich. Die Untersektion beinhaltet Fragen über Standardsoftware bzw. Individualsoftware. Es kann angenommen werden, dass Kenntnis über Standard- und Individualsoftware ein detaillierteres Wissen benötigen, als Kenntnis über Make-or-Buy. Deswegen befinden sich die beiden Items in einer Sektion feinerer Granularität, als das Item „make or buy“.

In diesem Bereich können zudem über ein Kontextmenü Testelemente erstellt und gelöscht werden, was später noch genau erklärt wird.

Bereich 2 (Editierbereich): Der zweite Bereich ist dazu da die Eigenschaften der Testelemente über Formulare zu editieren. Damit Ihre Einstellungen gespeichert werden, müssen Sie bei den Formularen jeweils auf „Save“ klicken.

Bereich 3 (Toolbar): Über den Bereich drei, kann die Ansicht des zweiten Bereichs geändert werden.

Der Editor wird verwendet, um die eigentlichen Testelemente zu bearbeiten.

Mit der „Precondition“-Schaltfläche können den einzelnen Sektionen Vorbedingungen zugewiesen werden. Diese werden verwendet, um während des Tests bestimmte Sektionen nur unter gewissen Umständen anzuzeigen.

Die Vorbedingungen beziehen sich immer auf Items der gröberen Sektion. Beim Test in Abbildung A.5 z.B. hat die Sektion „sektion_1_1“ eine Vorbedingung auf das Item „make or buy“, was bedeutet, dass sie einem Kandidaten nur präsentiert wird, wenn die Frage „make or buy“ richtig beantwortet wurde.

Neben Vorbedingungen können den Sektionen Lernmaterialien zugewiesen werden. Dabei handelt es sich um die FOIS-Module, welche Informationen im Bereich der Wirtschaft anbieten.

Die einzelnen Funktionen werden weiter unten im Detail erklärt.

A.2.3 Bedienung der Tree-View Komponente

Um Kinder eines Knotens anzuzeigen, müssen Sie, wie bei Tree-View-Elementen gängig, auf das Plus-Zeichen (+) klicken. Nach dem Aufklappen ändert sich das Plus in ein Minus, welches Sie dazu benützen können, um die Kinder wieder zu verstecken.

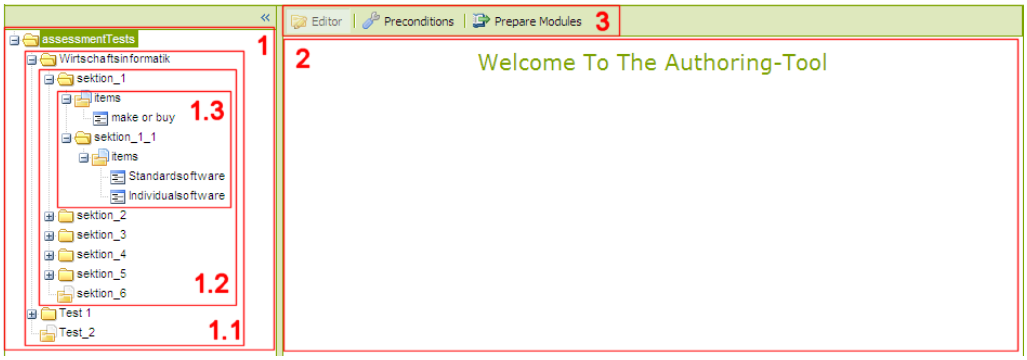

Abbildung A.5: Die Startseite des Authoring-Tools

A.2.4 Inaktivität

Aus Sicherheitsgründen werden Sie bei längerer Inaktivität aus dem CasIS-Portal abgemeldet. Dies kann vorkommen, wenn Sie mehrere Minuten nichts in dem Portal oder im Authoring-Tool anklicken. Damit Sie mit Ihrer Arbeit weiterfahren können, müssen Sie sich wieder anmelden. Daten gehen dabei nur in jenem Fall verloren, wenn Sie vor der Inaktivität nicht gespeichert haben.

A.2.5 Erstellen von Testelementen

Um einen Test zu erstellen, klicken Sie mit der rechten Maustaste auf den Knoten „assessmentTest“. Dabei geht das in Abbildung A.6 dargestellte Kontextmenü auf. Sollte kein Menü erscheinen, versuchen Sie es mit der linken Maustaste nochmals, während Sie die CTRL-Taste auf der Tastatur gedrückt halten.

Wenn Sie auf „Add Test“ drücken, erscheint ein Dialog, der Sie nach dem Titel des Tests fragt. Dieser Dialog ist in Abbildung A.7 ersichtlich. Um zu bestätigen, klicken Sie auf „OK“, womit der Test erstellt wird.

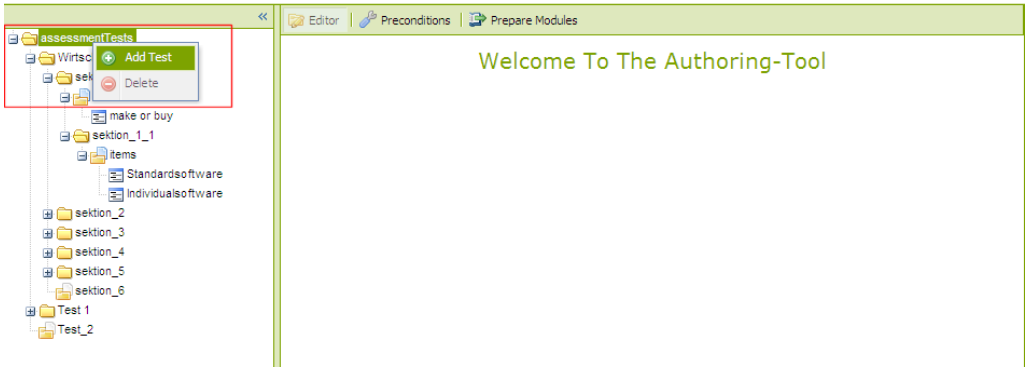

Abbildung A.6: Kontextmenü um einen Test zu erstellen

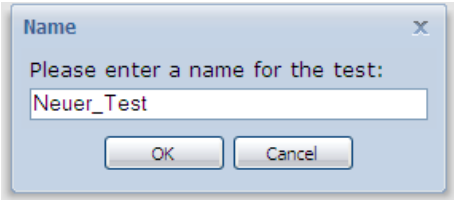

Abbildung A.7: Dialog um den Titel des Tests zu bestimmen

Eine Sektion wird sehr ähnlich wie ein Test erzeugt. Da Sektionen entweder in einem Test oder einer anderen Sektion erstellt werden können, muss das Kontextmenü auch auf diesen Testelementen aufgerufen werden. Sie können also mit der rechten Maustaste auf einen Test oder eine Sektion gehen und dann analog, wie oben vorgehen.

Um ein Item zu erzeugen, können Sie wieder gleich vorgehen. Items können aber lediglich in Sektionen erstellt werden.

A.2.6 Bearbeiten von Testelementen

Um die Testelemente zu bearbeiten, müssen Sie, wie in Abbildung A.8 (Markierung 2) ersichtlich, in der Toolbar den „Editor“ aktivieren.

Test-Editor Um einen Test zu bearbeiten müssen Sie ihn in der Tree-View-Komponente auswählen (Markierung 1 in Abbildung A.8), was zur Folge hat, dass der Test-Editor (Markierung 3 in Abbildung A.8) erscheint. In diesem Editor können Sie nun den Titel, den Autorennamen, die Fallstudie und eine Beschreibung angeben.

Ein Test steht immer mit genau einer Fallstudie in Beziehung, wobei gleiches für die Gegenrichtung gilt. Deswegen kann diese Verknüpfung zwischen den beiden Modulen hier stattfinden.

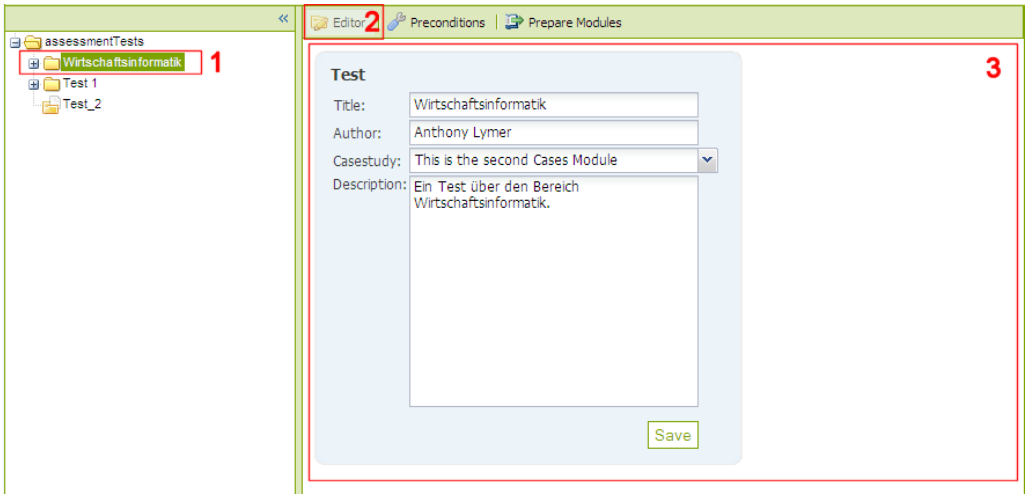

Abbildung A.8: Bearbeiten eines Tests

Section-Editor Um eine Sektion zu bearbeiten, müssen Sie in der Tree-View-Komponente die gewünschte auswählen. Bei einer Sektion kann jedoch, wie in Abbildung A.9 ersichtlich, nur ein Titel angegeben werden. Dieser Titel dient Ihnen lediglich um die Übersicht des Tests aufrechtzuerhalten. Bei der Auslieferung an einen Kandidaten erscheint dieser Titel nicht.

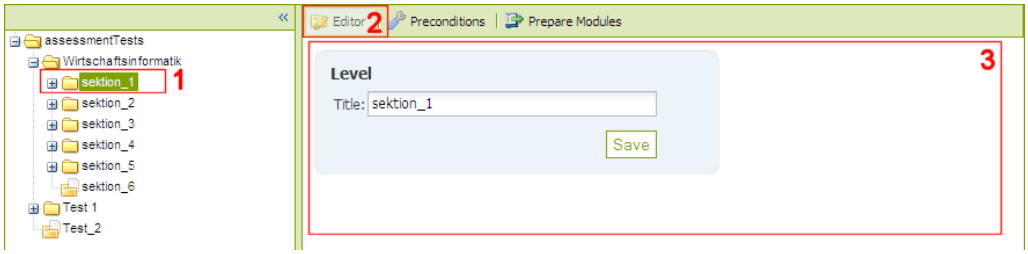

Abbildung A.9: Bearbeiten einer Sektion

Item-Editor Für die Bearbeitung von Items wird wieder gleich vorgegangen wie bei dem Test und der Sektion; in der Tree-View-Komponente wird das gewünschte Testelement ausgewählt. Sofern in der Toolbar der Editor aktiv ist, erscheint das in Abbildung A.10 ersichtliche Formular.

Die wichtigsten Eigenschaften eines Items sind naturgemäss die Frage selbst und dessen Antwortalternativen. Wenn Sie Hilfe im Verfassen guter Multiple-Choice-Items benötigen, können die Tipps in [Krebs, 2004] herangezogen werden.

Neben den Fragen und Antworten können Sie jedem Item zusätzlich eine Gewichtung von eins bis drei Punkten zuweisen. Wenn ein Benutzer die Frage vollständig korrekt beantwortet, erhält er dabei, die von Ihnen definierte Punktzahl. Andernfalls werden ihm gewisse Punkte abgezogen. Einer schwierigeren Frage sollten Sie natürlich eine höhere Gewichtung zuweisen.

Ausserdem können Sie bestimmen um welchen Fragetyp es sich bei dem Item handeln soll. Wenn Sie Multiple-Choice wählen, kann ein Kandidat bei dem Test eine oder mehrere Antworten abgeben. Bei Single-Choice kann bloss eine Antwort, als die korrekte, gewählt werden.

A.2.7 Ändern der Reihenfolge von Testelementen

Die einzelnen Items werden im fertigen Test in der Reihenfolge präsentiert, wie sie in der Tree-View-Komponente aufgelistet sind. Es wird dabei von der gröberen Sektion immer in die feinere Sektion gewandert. Beispiel: Bei der in Abbildung A.5 ersichtlichen Teststruktur, würde zuerst das Item „make or buy“ geliefert und anschliessend „Standardsoftware“ und „Individualsoftware“.

Um die Reihenfolge von Sektionen oder Items zu ändern, können Sie ganz einfach über „Drag and Drop“ die Elemente verschieben. In Abbildung A.11 sehen Sie ein Beispiel einer solchen Aktion.

Es ist aber nur möglich die Reihenfolge der Sektionen und Items der gleichen Hierarchiestufe zu verändern.

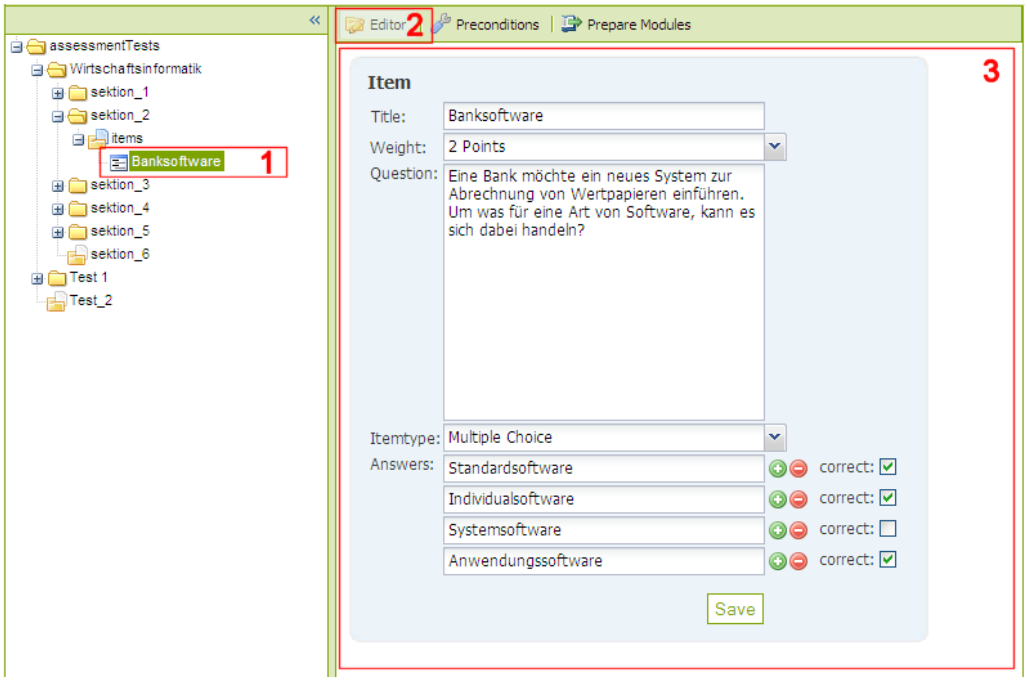

Abbildung A.10: Bearbeiten eines Items

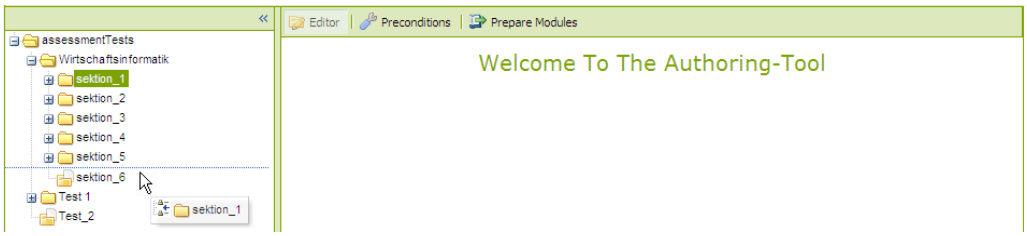

Abbildung A.11: Reihenfolge verändern mittels „Drag and Drop“

A.2.8 Löschen von Testelementen

Wenn Sie Tests, Sektionen oder Items löschen wollen, können Sie erneut von dem Kontextmenü (ähnlich dem in Abbildung A.6) Gebrauch machen. Gehen Sie auf das Testelement, das Sie löschen wollen und rufen Sie das Menü auf. Darauf können Sie nun auf „Delete“ klicken, um den Löschvorgang zu starten. Bevor das Testelement entfernt wird, werden Sie über einen Dialog (Abbildung A.12) nochmals gefragt, ob Sie das Löschen fortführen wollen. Bei einer Zusage werden das Element und gegebenenfalls alle zugehörigen Sektionen und Items endgültig gelöscht.

A.2.9 Zuweisen von Vorbedingungen

Vorbedingungen bieten die Möglichkeit Sektionen nur anzeigen zu lassen, wenn eine Bedingung erfüllt wurde. Die Bedingung beinhaltet dabei immer das korrekte Beantworten

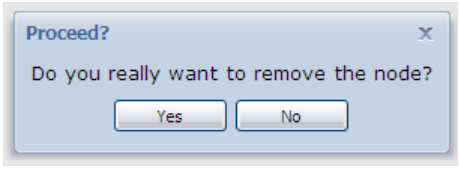

Abbildung A.12: Dialog vor dem Löschen des Testelements

von Fragen einer gröberen Granularität. Ein Beispiel einer zugewiesenen Vorbedingung ist in Abbildung A.13 ersichtlich. Hier wurde „sektion_1_1“ eine Vorbedingung zugewiesen, die besagt, dass die Sektion dem Kandidaten nur präsentiert wird, wenn das Item „make or buy“ richtig beantwortet wurde.

Um Vorbedingungen zu definieren, müssen Sie auf die entsprechende Sektion klicken und in der Toolbar „Precondition“ anwählen. Sie können beliebig viele Items der gröberen Granularität als Vorbedingung definieren, indem Sie über die Checkboxen das Gewünschte selektieren. Es muss aber mindestens eines als Vorbedingung definiert sein.

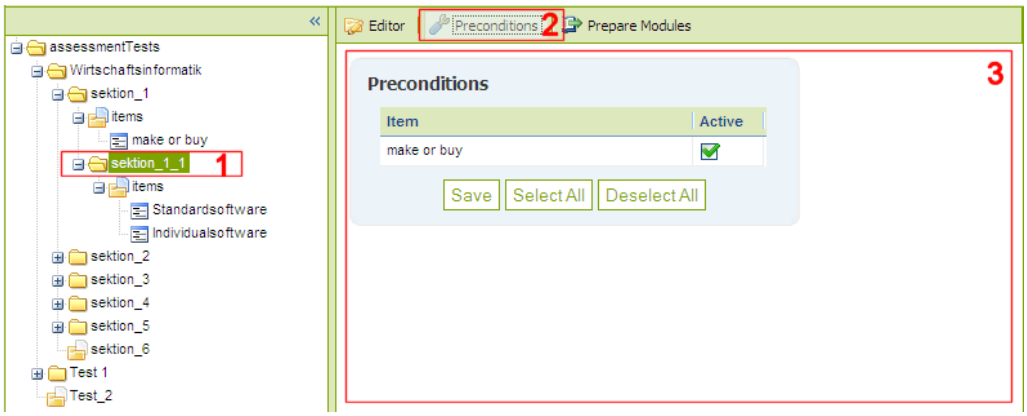

Abbildung A.13: Definieren von Vorbedingungen

A.2.10 Zuweisen von Lernmaterial

Da ein Kandidat nach dem Test mit Lernmaterial beliefert werden soll, muss im Authoring-Tool die Möglichkeit geboten werden, die passenden Vorbereitungsmodule dem Test zuzuweisen. Die Zuweisung der Module ist dabei nur bei Sektionen möglich.

Vorbereitungsmodule decken jeweils ein grösseres Themengebiet ab, wobei sie in Lerneinheiten gegliedert sind und somit eine präzisere Zuweisung möglich wird. Um Vorbereitungsmodule zuzuweisen, müssen Sie die gewünschte Sektion selektieren und über die Toolbar „Prepare Modules“ anwählen. Wie Sie in Abbildung A.14 sehen können, wurde der Sektion „sektion_1_1“ das Vorbereitungsmodul „This is the first prepare module“ zugewiesen, wobei auf die Lerneinheit „company background“ verwiesen wird. Die erste Combobox stellt dabei immer das Vorbereitungsmodul dar, während die zweite eine Lerneinheit des ausgewählten Moduls zeigt.

Ausserdem können Sie bestimmen, ab welcher Fehlerrate das Modul einem Testkandidaten empfohlen werden soll. Sie haben dabei die Möglichkeit zwischen drei vordefinierten Werten zu wählen.

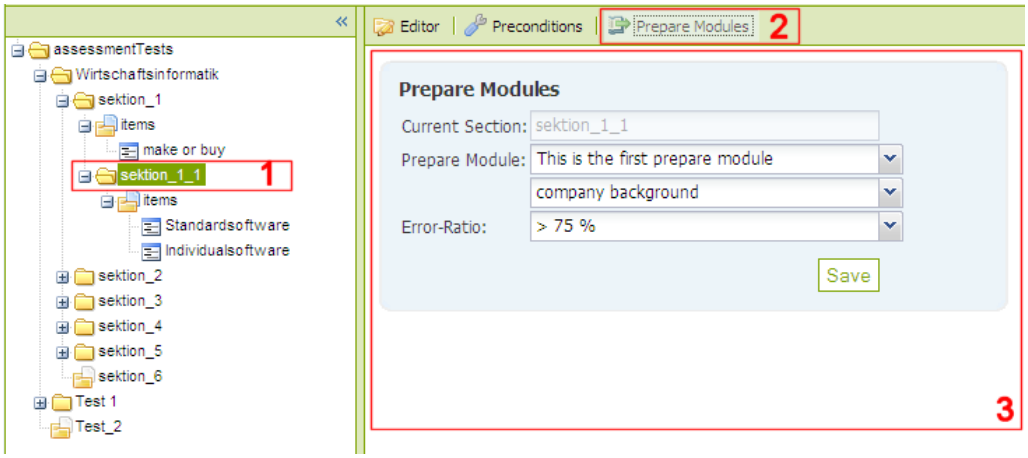

Abbildung A.14: Zuweisen von Vorbereitungsmodulen

A.3 Integrierung des Tests

A.3.1 Aufschalten

Wie schon oben beschrieben existiert für jeden Test genau eine Fallstudie und für jede Fallstudie genau ein Test. Im Abschnitt über das Authoring-Tool wurde erklärt, wie die beiden Module verknüpft werden können.

Um einen Test den Benutzern freizuschalten, müssen Sie in dem Admin-Bereich auf die Fallstudien gehen. Wie Sie in diesen Bereich gelangen, können Sie in der Abbildung A.3 sehen. Dabei muss statt „Entry Test“ „Cases“ und danach „Browse“ gewählt werden. Dort angekommen, können Sie nun in die Detail-Sicht (ähnlich wie in Abbildung A.4) gehen und bei der entsprechenden Fallstudie auf „Select“ klicken. Dabei wird neben der gewählten Fallstudie zusätzlich der Eintrittstest freigeschaltet.

A.3.2 Überprüfen

Da Sie als Administrator auch alle Möglichkeiten haben, die ein Benutzer hat, kann überprüft werden, ob das Aufschalten des Eintrittstests geklappt hat. Dabei wählen Sie beim CasIS-Portal einfach „Entry Test“ aus, was Ihnen eine Übersicht über alle aufgeschalteten Tests gibt. Nun sollten Sie Ihren Test, der zur eben aufgeschalteten Fallstudie gehört, sehen.

Falls Sie den Test nun kontrollieren wollen, können Sie ihn starten. Achten Sie dabei aber darauf, dass nicht alle von Ihnen erstellten Items vorkommen müssen, da unter Umständen die definierten Vorbedingungen dies korrekterweise verhindern.

Abbildungsverzeichnis

Literaturverzeichnis

[qti, 2002a] (2002a). Ims question & test interoperability: An overview. http://www.imsglobal.org/question/qtiv1p2/imsqti_oviewv1p2.html (Stand 23.10.07).

[qti, 2002b] (2002b). Ims question & test interoperability: Asi best practice & implementation guide. http://www.imsglobal.org/question/qtiv1p2/imsqti_asi_bestv1p2.html (Stand 23.10.07).

[ext, 2006] (2006). Extjs - javascript library. http://www.extjs.com/ (Stand 07.10.07).

[qti, 2006] (2006). Ims question and test interoperability overview - version 2.1 public draft (revision 2) specification. http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_oviewv2p1pd2.html (Stand 23.10.07).

[tom, 2007] (2007). Apache tomcat. http://tomcat.apache.org/ (Stand 07.10.07).

[vel, 2007] (2007). The apache velocity project. http://velocity.apache.org/engine/index.html (Stand 07.10.07).

[xin, 2007] (2007). Apache xindice. http://xml.apache.org/xindice/ (Stand 07.10.07).

[ims, 2007] (2007). Content packaging specification. http://www.imsglobal.org/content/packaging (Stand 07.10.2007).

[dwr, 2007] (2007). Dwr - easy ajax for java. http://getahead.org/dwr (Stand 07.10.07).

[exh, 2007] (2007). Exhibit 2.0. http://simile.mit.edu/exhibit/ (Stand 20.10.07).

[xml, 2007] (2007). Extensible markup language (xml). http://www.w3.org/XML/ (Stand 07.10.07).

[foi, 2007] (2007). Fois - foundations of information systems. http://www.fois.ch/ (Stand 07.10.07).

[qti, 2007a] (2007a). Ims question & test interoperability specification. http://www.imsglobal.org/question/ (Stand 23.10.07).

[jso, 2007] (2007). Introducing json.
http:/json.org (Stand 07.10.07).

[jsp, 2007] (2007). Javaserver pages technology.
http://java.sun.com/products/jsp/ (Stand 07.10.07).

[jax, 2007] (2007). Jaxb reference implementation.
http://jaxb.dev.java.net/ (07.10.07).

[ola, 2007] (2007). Olat an der universität zürich.
http://www.id.uzh.ch/dl/elearning/olatunizh.html (Stand 23.10.07).

[qti, 2007b] (2007b). Qti 2.1: Ready for implementation.
http://qtitools.caret.cam.ac.uk/index.php?option=com_content&task=view&id=14&Itemid=9 (Stand 07.10.07).

[r2q, 2007] (2007). R2q2: Rendering and response processing services for qtiv2 questions.
http://www.r2q2.ecs.soton.ac.uk/overview/index.htm (Stand 07.10.07).

[str, 2007] (2007). Struts - the apache software foundation.
http://struts.apache.org/ (Stand 23.10.07).

[yui, 2007] (2007). The yahoo! user interface library (yui).
http://developer.yahoo.com/yui/ (Stand 23.10.07).

[Anh Nguyen-Tuong, 2005] Anh Nguyen-Tuong, Salvatore Guarnieri, D. G. J. S. D. E. (2005). Automatically hardening web applications using precise tainting. http://www.cs.virginia.edu/papers/infosec05.pdf (Stand 07.10.2007).

[Fisk, 2007] Fisk, H. (2007). Prepared statements.
http://dev.mysql.com/tech-resources/articles/4.1/prepared-statements.html (Stand 14.10.2007).

[Greer and Huang, 1996] Greer, J. A. C. J. E. and Huang, S. X. (1996). Adaptive assessment using granularity hierarchies and bayesian nets.

[Krebs, 2004] Krebs, R. (2004). Anleitung zur herstellung von mc-fragen und mc-prüfungen für die ärztliche ausbildung. http://www.iml.unibe.ch/fileadmin/iml-website/aae/pdf/MC_Anleitung.pdf (Stand 07.10.2007).

[Linacre, 2000] Linacre, J. M. (2000). Computer-adaptive testing: A methodology whose time has come. http://www.rasch.org/memo69.pdf (Stand 07.10.07).

[Mahemoff, 2007] Mahemoff, M. (2007). *Ajax Design Patterns*. O'Reilly.

[P. Leach, 2005] P. Leach, M. Mealling, R. S. (2005). A universally unique identifier (uuid) urn namespace. http://www.greenbytes.de/tech/webdav/rfc4122.pdf (Stand 07.10.2007).

[Piendl, 2005] Piendl, D. T. (2005). E-learning mit webct an der eth zürich.
http://www.id.ethz.ch/events/idforum/archiv/01072005/E-Learning_mit__an_der_ETH.pdf (Stand 23.10.07).

[Shiflett, 2007] Shiflett, C. (2007). Ajax is not an acronym.
http://shiflett.org/blog/2007/apr/ajax-is-not-an-acronym (Stand 23.10.07).