Design-Beschreibung des Open-Source Projekts „WebAssign“

1. Allgemeines

Kurzcharakterisierung
WebAssign bietet eine umfassende Plattform für die Durchführung von Übungsveranstaltungen im WWW. Dabei werden Studierende ebenso wie Lehrende unterstützt. Die Erstellung von Aufgaben, das Einsenden von Lösungen, deren Korrektur sowie die Distribution von Ergebnissen erfolgen über das Internet. Den Studierenden werden zeitlich und räumlich flexible und komfortable Arbeitsbedingungen angeboten und den Lehrenden aufwendige administrative Arbeiten und Routinetätigkeiten abgenommen. Die Funktionalität von WebAssign ist auf eine breite Anwendbarkeit und Alltagstauglichkeit ausgelegt.

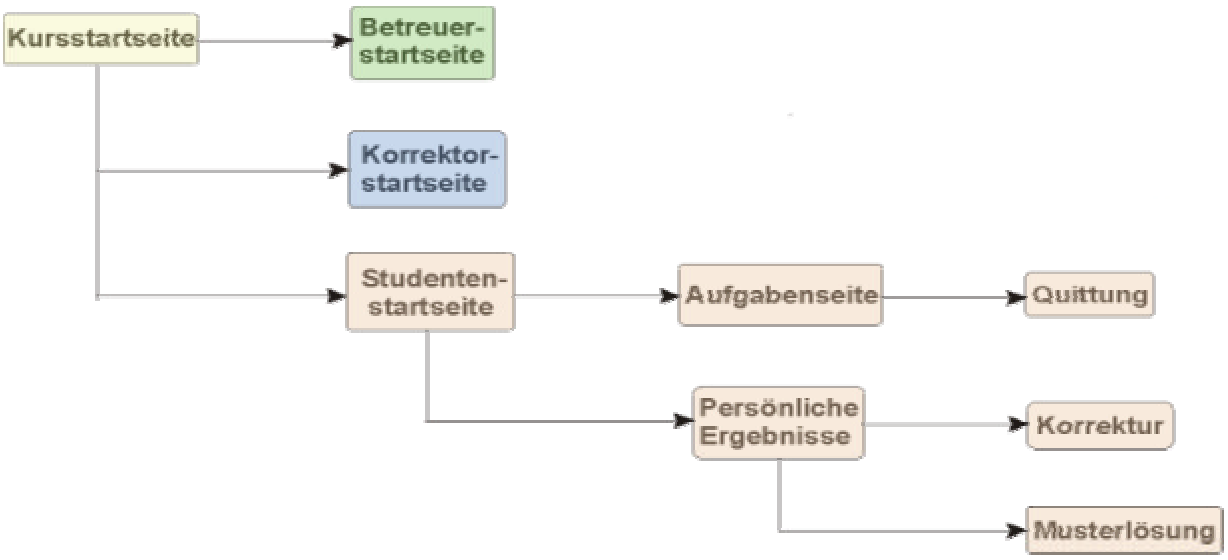
Systemvoraussetzungen
Zum Betrieb von WebAssign ist ein Web-Server erforderlich. Der Server muss in der Lage sein, über eine Servlet-Engine Java-Servlets anzusteuern. Zur Datenverwaltung ist ein relationales Datenbankmanagementsystem erforderlich, dass Treiber für JDBC-Verbindungen anbietet. Dies wären zum Beispiel MySQL oder Informix. Weiterhin ist ein in Java implementierter Server zur Realisierung notwendig.
Eine Mindestanforderung an die Hardwarevoraussetzungen kann nicht angegeben werden, da die Anforderungen von der Frequentierung des Servers abhängen.

Beschreibung der Produktumgebung
Der Nutzer benötigt zur Nutzung und Betrachtung von WebAssign einen Internetbrowser wie z.Bsp. Microsoft Internet Explorer oder Netscape.

Abgrenzung
WebAssign unterstützt nur den Übungsbetrieb. Eine Klausurauswertung bzw. eine Einschreibung zur Klausur wird nicht unterstützt.

2. Produktübersicht

Die folgende Abbildung zeigt die wichtigsten Seiten von WebAssign im Überblick. Jeder Kurs hat eine Einstiegsseite (Kursstartseite), von der aus die übrigen Seiten erreicht werden können.



Die Kursstartseite ist die zentrale Anlaufstelle für alle Beteiligten und frei zugänglich. Eine Überprüfung der Zugangsberechtigung findet erst bei den von hier aus erreichbaren Seiten statt. Die Kursstartseite kann an die Erfordernisse des jeweiligen Kurses angepasst werden.

Hat man sich als Student angemeldet, so sieht man auf der Studentenstartseite eine nach Aufgabenheften geordnete Übersicht aller schon bearbeiteten bzw. noch zu bearbeitenden Aufgaben. Für Aufgabenhefte, deren Bearbeitungszeitraum noch nicht begonnen hat, sind die einzelnen Aufgaben noch nicht zugänglich. Zum eigentlichen Aufgabentext gelangt man über den entsprechenden Link. Der Student erhält zur Überprüfung seiner persönlichen Ergebnisse eine Übersicht seiner Korrekturen mit deren Status und den erreichten Punkten, sofern die Korrektur schon abgeschlossen ist. Des weiteren enthält die Übersicht Links zu den einzelnen Musterlösungen. Diese Seite ist nur dem jeweiligen Studenten zugänglich.
Die Studentenstartseite kann ebenfalls an den jeweiligen Kurs angepasst werden.
Ein typischer Ablauf (von Anmeldung bis Ansehen der Musterlösung)aus Sicht eines Studenten ist in der beiliegenden Datei „WebAssign aus Studentensicht“ dargestellt.

Auf der Aufgabenseite befindet sich der eigentliche Aufgabentext. Durch das Eintragen von Lösungen in die dafür vorgesehen Felder wird die Aufgabe bearbeitet und durch Drücken des Einsendeknopfes zum WebAssign-Server geschickt. Die Bearbeitung einer Aufgabe ist jedoch erst endgültig abgeschlossen, wenn das gesamte Heft geschlossen oder das Bearbeitungsende der Aufgabe überschritten ist. Bis dahin kann die Aufgabe beliebig oft eingesandt werden.

Die Betreuerstartseite ist die zentrale Anlaufstelle für Kursbetreuer. Von hier aus kann der Betreuer alle administrativen Funktionen erreichen. Hier befindet sich z.B. eine Aufgabenübersicht, die zu den Konfigurationsseiten für die einzelnen Aufgaben führt. Es existiert eine Einsendungs- und Korrekturübersicht. Diese gibt einen nach Aufgabenheften und Korrektoren geordneten Überblick, wie viele Aufgaben welchen Status haben. Die Betreuerstartseite und deren Folgeseiten sind passwortgeschützt nur für Betreuer zugänglich.

Auf der Korrektorstartseite sieht ein Korrektor die ihm zugeordneten Aufgaben im Überblick. Nach Aufgabenheften geordnet wird angezeigt, wie viele Aufgaben welchen Status haben. Sie ist ebenfalls passwortgeschützt nur für Korrektoren zugänglich.

Hier noch ein typischer Ablauf bei der Arbeit mit WebAssign für den Fall neu erstellter Aufgaben mit manueller Korrektur:

Der Betreuer gestaltet die Aufgabenseiten mit Hilfe eines beliebigen HTML-Editors. Medientypen, Rahmenstrukturen oder zusätzliche Seiten können, unter Berücksichtigung einiger Vorschriften zur Vergabe von Namen für Eingabeelemente und "Absendeknöpfe", frei benutzt werden.
Das Aufgabenmaterial wird ins System geladen. Dieses überprüft automatisch die Aufgabenstruktur und konfiguriert dementsprechend die Datenbank.
Der Betreuer bestimmt die Bearbeitungszeit, während der die Studenten ihre Lösungen einsenden können. Er konfiguriert den Korrekturmodus und gestaltet Schablonen für die Empfangsbestätigungen und Rückmeldungsseiten.
Während der Bearbeitungsphase eines Aufgabenheftes kann ein Student die Aufgabenseiten aufrufen, mit seiner Lösung versehen und diese an das WebAssign-System einsenden. Die Einsendung wird gespeichert und direkt durch eine Quittung vom System bestätigt. Wenn für die Aufgabe eine Vorkorrektur eingestellt wurde, testet das System die Lösungen und teilt die Testergebnisse zusammen mit der Eingangsquittung mit. Wenn nötig kann der Student nun mehrmals seine Lösungen verbessern und diese dann wieder ans System übergeben und zwar solange, bis die Bearbeitungsphase des Aufgabenheftes beendet ist.
Sobald die Bearbeitungsphase für das Aufgabenheft vorbei ist, erstellt das System eine sogenannte Rückmeldungsseite, die die Lösungsbestandteile eines Studenten enthält und ordnet diese einem Korrektor zu.
Der Korrektor bewertet die studentischen Einsendungen online oder offline. Im Online-Modus kann er direkt auf die Rückmeldungsseiten zugreifen, sie mit Anmerkungen versehen und Punkte vergeben. Im Offline-Modus muss er sich die Rückmeldungsseiten zunächst herunterladen, sie bewerten und anschließend zum WebAssign-System heraufladen.
Der Betreuer überprüft die Arbeit des Korrektors und gibt die Korrektur frei.
Der Student erhält eine automatisch generierte E-Mail, dass er auf seine Korrekturen zugreifen kann.

3. Grundsätzliche Design-Entscheidungen

WebAssign basiert auf einer Servlet-Architektur. Dadurch können Java-Programme über einen Web-Server aufgerufen werden. Servlets sind Java-Programme, die auf der Seite des Servers ausgeführt werden.
WebAssign besitzt ein zentrales Servlet, dass alle Anfragen an den WebApplicationServer weiterleitet.

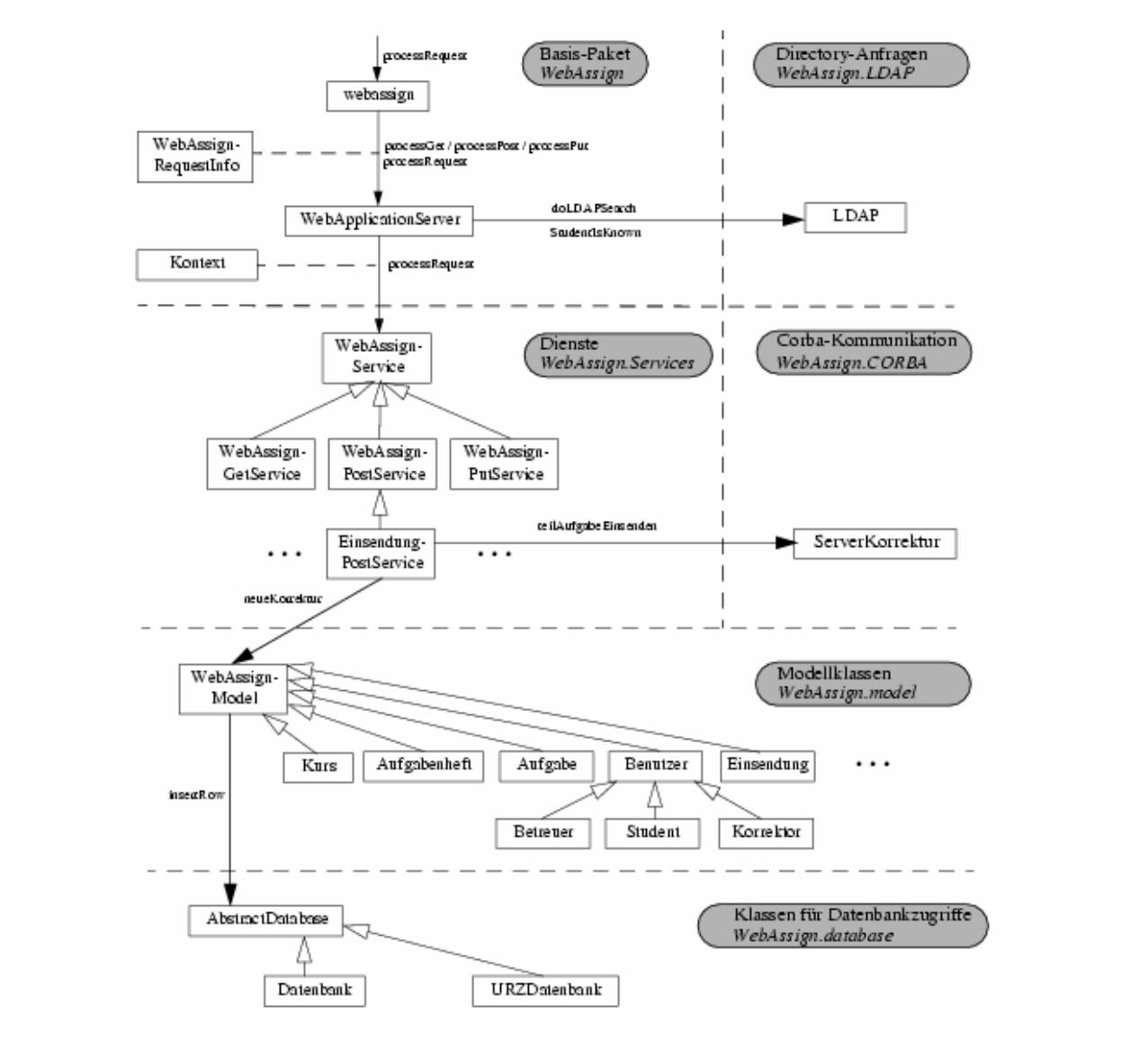
4. Paket- und Klassenstruktur

Es war uns leider nicht möglich, ein Reverse Engineering von WebAssign durchzuführen. Trotz zahlreicher Versuche war die heruntergeladene Datei immer fehlerhaft bzw. die Ordner leer.
Die mdl-Datei zeigte zwar im linken Rahmen die importierten Ordner und alle Elemente, aber keine Diagramme an. Die Option „Automatic Resize (off)“ konnte nicht ausgewählt werden.
Beim Öffnen der mdl-Datei zu Hause kam die Meldung, dass die petal-Versionen nicht übereinstimmen.
Und im Windowspool konnte Rational Rose nicht gestartet werden.
Die vorliegende Beschreibung der Paket- und Klassenstruktur beruht auf Dokumenten, die auf www-pi3.fernuni-hagen.de veröffentlicht sind.

Es existiert eine Javadoc-Dokumentation zu WebAssign.
Diese ist über www.campussource.de/software/webassign oder über die angegebene Website der Fernuniversität Hagen zu erreichen.

Die Paketstruktur
Das über WebServer bzw. Servlet-Engine aufgerufene Servlet „webassign“ leitet den Request -- alle Angaben darüber sind im Objekt WebAssignRequestInfo gekapselt -- weiter an den WebApplicationServer. Dort wird die URL analysiert, der aufgerufene Dienst extrahiert und die Zugangsdaten des Users geprüft, gegebenenfalls über einen Aufruf an einen Directory-Server (Paket WebAssign.LDAP). Das Kontext-Objekt hält die Attribute des gerade agierenden Benutzers vor. WebAssignService stellt die Basisfunktionalität für alle Serviceklassen zur Verfügung, etwa den Aufbau des Antwort-Datenstroms, die Expansion der darin enthaltenen Variablen und das Setzen von Content-type und -length. Die Services gliedern sich auf in Get-, Post- und Put-Services, entsprechend der Art des ursprünglichen HTTP-Requests.
Unter den Serviceklassen kommt "EinsendungPostService" eine besondere Bedeutung zu: Neben der Speicherung der eingesandten Daten in die DB wird hier die automatische Korrektur einer Aufgabe angestoßen. Hiefür wird eine CORBA-Kommunikation (Paket WebAssign.CORBA) mit dem registrierten Korrekturmodul aufgebaut.

Die Paketübersicht mit den zentralen Klassen und deren Abhängigkeiten:



Domänenklassenmodell

1. Universitäre Institutionen

Eine WebAssign-Installation können verschiedene Veranstalter, auch aus unterschiedlichen Universitäten, unabhängig voneinander nutzen. Veranstalter sind in der Regel Lehrgebiete. Man kann aber auch größere (Institut, Fachbereich) oder kleinere Einheiten (ein einzelner Dozent) als Veranstalter realisieren. Dabei ordnet sich jeder Veranstalter einer Universität zu. 2. Veranstaltungen Zentraler Gegenstand von WebAssign sind Übungsveranstaltungen, die in der Regel zu einem Kurs bzw. einer Vorlesung gehören. Jeder Kurs ist einem Veranstalter zugeordnet und ist durch eine Kursnummer und eine Versionsnummer identifiziert. Kursnummern kann jeder Veranstalter frei wählen.

3. Lehrmaterial

Das Lehrmaterial zu einem WebAssign-Kurs besteht aus weitgehend frei gestaltbaren HTML- Seiten, die ggfs. um Bilder, Applets usw. ergänzt werden. Alle Materialien werden in der Tabelle *kursressource* gespeichert. Jeder Bestandteil -- HTML-Seite, Bild, Applet usw. -- ist einem Kurs zugeordnet und durch seinen Namen identifiziert. Damit sind die Materialien verschiedener Kurs voneinander unabhängig. Zum Teil müssen vorgegebene Namen verwendet werden (s.u.), insbesondere für die HTML-Seiten von Aufgaben und Musterlösungen. Metadaten werden in den Tabellen *aufgabenheft*, *aufgabe*, *teilaufgabe* und *eingabefeld* gespeichert, wodurch die im Folgenden beschriebene Struktur realisiert wird. Metadaten werden zum Teil automatisch erzeugt -- durch Analyse der heraufgeladenen Materialien. Sie können aber auch von Hand bearbeitet werden. Zu einem Kurs gibt es ein oder mehrere Aufgabenhefte. Zu jedem Aufgabenheft gehören ein oder mehrere Aufgaben. Je Aufgabenheft werden in der Tabelle *aufgabenheft*

Bearbeitungsbeginn und -ende festgelegt. Während dieses Zeitraums können die zugehörigen Aufgaben von den Studierenden bearbeitet werden. Zu jeder Aufgabe gehören ein Aufgabentext und in der Regel eine Musterlösung sowie Hinweise für den Korrektor. Hinzu kommt eine Korrekturseitenschablone, in die nach Einsendeschluss die Lösungsbestandteile eines Studierenden eingetragen werden (Die dadurch entstehende Korrekturseite geht an den Korrektor und schließlich zurück zum Studierenden). Der Aufgabentext steht auf einer HTML-Seite mit dem Namen aufgabe<n>.<m>.html, wobei n die Nummer des Aufgabenhefts und m die Nummer der Aufgabe ist. Analog stehen Musterlösungen auf der Seite muster<n>.<m>.html. Die Seiten für Korrekturhinweise und die Korrekturseitenschablonen werden mit hinweise<n>.<m>.html bzw. korrektur<n>.<m>.html bezeichnet.
Metadaten über die Anzahl von Aufgabenheften und Aufgaben können von Hand bearbeitet oder durch Heraufladen von Aufgabenseiten automatisch erzeugt werden. Zu jeder Aufgabe können in der Tabelle *aufgabe* ein sprechender Name für die Aufgabe sowie Angaben zur Korrekturart (von Hand, automatisch, beides oder keine) eingetragen werden. Bei automatischer Korrektur muss diese durch den Namen eines Korrekturmoduls und einen Konfigurations-String für den Korrekturmodul präzisiert werden. Bei Bedarf kann eine Aufgabe in Teilaufgaben zerlegt werden, die von den Studierenden jeweils getrennt eingeschickt werden können. Eine ungeteilte Aufgabe hat genau eine Teilaufgabe. Für jede Teilaufgabe enthält die Aufgabenseite eine HTML-Form, ein oder mehrere Eingabefelder sowie einen Submit-Knopf zum Einschicken. Metadaten über die Anzahl von Teilaufgaben und Eingabefeldern werden durch Analyse der Aufgabenseite gewonnen. Von Hand können für jede Teilaufgabe erreichbare Punkte und Angaben zur automatischen Korrektur in der Tabelle *teilaufgabe* eingetragen werden. Zu jeder Teilaufgabe gehört auch eine Quittungsschablone. Diese Schablone legt die HTML-Seite fest, die ein Studierender nach einer Einsendung zurückbekommt. Hier kann z.B. seine gerade eingeschickte Lösung noch einmal wiederholt oder das Ergebnis der automatischen Vorkorrektur angezeigt werden. Die HTML-Seite für die Quittungsschablone hat wieder einen vorgegebenen Namen, der neben Aufgabenheft und Aufgabe auch die zugehörige Teilaufgabe identifiziert. Beispiele: quittung3.2.a.html oder quittung1.7.c.html.
Eigene Tabellen für Musterlösungen, Korrekturhinweise, Korrekturseiten- und Quittungsschablonen werden nicht benötigt. Die entsprechenden HTML-Seiten befinden sich in der Tabelle *kursressource* und können über ihre Namen zugeordnet werden. Auch Informationen darüber, welche Eingabefelder oder automatisch erzeugten Korrekturergebnisse in eine Korrekturseiten- oder Quittungsschablone einzutragen sind, werden nicht explizit als Metadaten gespeichert. Dies ergibt sich aus der Verwendung von bestimmten Variablen im HTML-Code der Schablone.

4. Personen

Bei den WebAssign-Benutzern werden drei Gruppen unterschieden -- Studenten, Betreuer und Korrektoren. Zu jeder der drei Gruppen gehört eine gleichnamige Tabelle. Alle Benutzer sind einem Veranstalter zugeordnet. Darüber hinaus werden Studenten durch ihre Matrikelnummer, Betreuer und Korrektoren durch ihren Namen identifiziert. Jeder Veranstalter kann damit Benutzer unabhängig von anderen Veranstaltern verwalten. Welcher Student welchen Kurs belegt hat, wird in der Tabelle *belegung* festgehalten. Analog dienen die Tabellen *betreuung* und *korrigierung* dazu, die Zuordnung von Betreuern und Korrektoren zu Kursen festzuhalten.

5. Veranstaltungsbezogene Studentendaten

Ein zentraler Bereich der WebAssign-Datenbank sind die Tabellen, die festhalten, wer wann was eingesendet hat und wie diese Einsendungen bewertet worden sind. Derartige Informationen werden analog zur Struktur des Aufgabenmaterials auf verschiedenen Ebenen festgehalten.
Auf der Ebene des Aufgabenhefts wird festgehalten, ob und wann ein Student ein bestimmtes Aufgabenheft geschlossen hat. Hierzu dient die Tabelle *hefteinsendung*. Auf der Aufgabenebene werden zum einen ggfs. angefallene Ergebnisse einer automatischen Korrektur (der ganzen Aufgabe) in der Tabelle *autokorrektur* gespeichert. Zum anderen wird je Aufgabe und Student eine Korrekturseite in der Tabelle *korrektur* gespeichert. Neben dem Inhalt der Seite (HTML-Code) werden der aktuelle Zustand (neu, unkorrigiert, beim Korrektor, korrigiert oder freigegeben) sowie die erreichte Punktzahl festgehalten.
Auf Teilaufgabenebene werden analog zur ganzen Aufgabe die Ergebnisse der automatischen Vorkorrektur in der Tabelle *vorkorrektur* gespeichert. Hier können allerdings keine Punktzahlen eingetragen werden. Das eigentliche Vorkorrekturergebnis zu einer Teilaufgabe ist ein String -- in der Regel HTML-Code. Das Schlüsselfeld *name* bekommt dabei den Wert 'LEER'. Darüber hinaus wird die Tabelle *vorkorrektur* auch dazu verwendet, bei der Vorkorrektur oder direkt beim Einsenden einer Teilaufgabe anfallende Bilder oder sonstige Dokumente zu speichern. In diesen Fällen werden im Feld

name der (Datei-) Name des Bildes bzw. Dokuments und im Feld *inhalt* dessen Inhalt eingetragen. Prinzipiell kann die Tabelle *autokorrektur* analog verwendet werden. Allerdings hat sich dort bisher noch keine sinnvolle erwendung für Bilder oder sonstige Dokumente ergeben, sodass das Feld *name* in der Tabelle *autokorrektur* immer den Wert 'LEER' enthält. Wie oft ein Student eine bestimmte Teilaufgabe eingesendet hat, wird in der Tabelle *einsendungszaehler* festgehalten.
Auf der untersten Ebene der Eingabefelder wird in der Tabelle *einsendung* die von einem Studenten zuletzt eingesandte Lösung, genauer: der im entsprechenden Eingabefeld eingetragene Lösungsbestandteil gespeichert. Dabei wird auch das Datum der letzten Einsendung vermerkt.

Domänenklassenmodell: