

Pflichtenheft der SF GmbH

SEDA-WI-Proj-B

SF GmbH

Denis Hamann	Matr.-Nr. 1684873
Anna Kupfer	Matr.-Nr. 1491515
Hannes Stadler	Matr.-Nr. 1692114
Christian Hindelang	Matr.-Nr. 1685285
Mario Serno	Matr.-Nr. 1687104

Otto-Friedrich-Universität Bamberg

Version: v1.1 - 9. November 2012

Inhaltsverzeichnis

1	Einleitung	3
1.1	Definitionen	3
2	Zielbestimmung	4
2.1	Musskriterien	4
2.2	Wunschkriterien	5
2.3	Abgrenzungskriterien	5
3	Produkteinsatz	6
3.1	Anwendungsbereiche	6
3.2	Zielgruppen	6
3.3	Betriebsbedingungen	6
4	Produktumgebung	7
4.1	Software	7
4.2	Hardware	8
4.3	Orgware	8
4.4	Produkt-Schnittstellen	9
4.5	Zukünftige Entwicklungen	9
5	Produktfunktionen	10
5.1	Prozesse mit Dateneingabe	10
5.2	Listen	14
5.3	Berichte	16
6	Produktdaten	17
7	Produktleistung	20
8	Benutzeroberfläche	23
8.1	Bildschirmlayout	23
8.2	Drucklayout	23
8.3	Tastaturbelegung	25
8.4	Dialogstruktur	25
9	Qualitätszielbestimmung	33

10 Testszenarien - Christian Hindelang	34
11 Entwicklungsumgebung	36
11.1 Software	36
11.1.1 Programmierstil	37
11.2 Hardware	38
11.3 Orgware	38
11.4 Produkt-Schnittstellen	38
12 Ergänzungen	39
12.1 Jar- Registry Extension	39
13 Glossar	40

Abbildungsverzeichnis

1	SERM der Daten-Architektur	18
2	Lehrstuhlplan	24
3	Dozentenstundenplan	24
4	Stundenplananzeige, rollenunabhängig	25
5	GUI Startseite, rollenunabhängig	27
6	Startseite für authentifizierte Benutzer der Hausverwaltung	28
7	Konfliktlösung der Raumanfragen	28
8	Verwaltung der Lehrstühle	29
9	Verwaltung der Nutzer	29
10	Verwaltung der Räume	30
11	Startseite für Dozenten	31
12	Verwaltung der Lehrveranstaltungen durch die Dozenten	31
13	Raumanfragen durch die Dozenten	32

1 Einleitung

(Anna Kupfer)

Die Vorliegende Arbeit enthält die Gesamtheit an notwendigen Spezifikationen die uns von der Otto-Friedrich-Universität Bamberg übermittelt wurden um ein Pflichtenheft zur Softwareentwicklung (Upgrade zum UnivIS 1.0, welches sich aktuell schon uni-weit im Einsatz befindet) einer Verwaltungsanwendung für Lehrveranstaltungen und Räumen für einen bestimmten Uni-Standort zu erarbeiten. In diesen Vorgaben, dem Lastenheft, wurden Ergebnisse aus eigens angeführten Ermittlungen zu den individuellen Anforderungen durch Fragebögen, Selbstaufschreibungen sowie Feldbeobachtungen zur aktuellen und gewünschten Einsatzsituation des UnivIS 1.0 und 2.0 zusammengestellt [vgl. [Bal09]]. Nach der ausführlichen Sichtung aller Unterlagen und einer ersten Vorstudie der Anforderungen konnte vorliegendes Pflichtenheft erstellt werden. In diesem finden Sie jegliche fachliche Definitionen und Anforderungen, die im Zusammenhang mit der angeforderten Software und den gewünschten Funktionen und Leistungen stehen.

Aus diesem Grund werden zunächst die unterschiedlichen Zielbestimmungen, der präzise Produkteinsatz sowie die gesamte Umgebung (Soft-, Hard-, Orgware und Produkt-Schnittstellen) erläutert. Dem folgen detaillierte Spezifikationen zu den Funktionen, Daten und Leistungen. Um erste Eindrücke zu vermitteln werden Anforderungen an die Benutzeroberfläche sowie erste Prototyp-User-Interfaces vorgestellt. Das Pflichtenheft schließt mit qualitätsbezogenen Zielbestimmungen, globalen Testszeanrien und genauen Angaben zur Entwicklungsumgebung. Weitere Ergänzungen sowie ein Glossar dienen der Komplettierung und Vermittlung einer besseren Verständlichkeit des vorliegende Dokuments.

1.1 Definitionen

In diesem Abschnitt werden Abkürzungen und Begrifflichkeiten erläutert, die im Pflichtenheft verwendet werden.

- | | |
|-------|---|
| /X0/ | Eine derartige Kennzeichnung wird im Pflichtenheft für die Kennzeichnung von Software-Merkmalen verwendet. Anstelle des „X“ steht die Abkürzung des jeweiligen Merkmals. „Z“ kennzeichnet Zielbestimmungen, „F“ Funktionen, „D“ Daten, „L“ Leistungen, „B“ die Benutzeroberfläche, „Q“ qualitative Bestimmungen und „T“ Testszenarien. Anstelle der „0“ steht die Nummer des jeweiligen Merkmals. |
| /XW0/ | Diese Kennzeichnung entspricht der wie bei „/X0/“ beschrieben, das zusätzliche „W“ signalisiert allerdings, dass es sich um ein wünschenswertes Merkmal handelt, dass je nach Entwicklungsaufwand und Dringlichkeit ggf. nicht in der ersten Version der Software implementiert sein wird. |

2 Zielbestimmung

(Mario Serno)

Im Rahmen dieses Kapitels werden die Ziele beschrieben, welche durch den Einsatz des Produktes erreicht werden sollen. Sie gliedern sich in Muss-, Wunsch- und Abgrenzungskriterien. Musskriterien beschreiben Ziele die das Produkt ohne Einschränkung erfüllen muss. Wishkriterien stellen Leistungen dar, welche für den Einsatzzweck nicht unbedingt erforderlich sind, die das Produkt aber dennoch bieten soll. Abgrenzungskriterien beschreiben Ziele, deren Erreichung bewusst nicht angestrebt wird. Sie dienen lediglich zur Eindämmung der Komplexität [vgl. [Bal96]] .

Da sich die Anwendungsfunktionen der Nutzer deutlich voneinander unterscheiden, ergeben sich nutzerspezifische Zielsetzungen. Die Nutzer des Systems sind Dozenten, die Universitäts- und Hausverwaltung sowie Studenten.

2.1 Musskriterien

- /ZM10/ Die Dozenten sollen die Lehrveranstaltungen verwalten können.
- /ZM20/ Die Universitäts- und Hausverwaltung soll die Raumdaten verwalten können.
- /ZM30/ Die Universitäts- und Hausverwaltung soll die Benutzerdaten aller Nutzer pflegen können.
- /ZM40/ Die Universitäts- und Hausverwaltung soll den Raumbedarf der Lehrstühle und die Raumplanung koordinieren können.
- /ZM50/ Die Universitäts- und Hausverwaltung sowie die Dozenten sollen sich für die spezifischen Aufgaben im System authentifizieren.
- /ZM60/ Alle Nutzer des Systems sollen individuelle Wochenpläne (für Räume und Lehrveranstaltungen) für den Universitätsstandort Erba erstellen können (Ausgabe i. F. einer PDF).
- /ZM70/ Es soll möglich sein, das System um die restlichen Universitätsstandorte, Lehrveranstaltungen und Angehörige der Universität nach einem Evaluationszeitraum erweitern zu können.
- /ZM80/ Der Nutzer soll einen Überblick über aktuell laufende und demnächst startende Lehrveranstaltungen erhalten können (Live-Ticker).
- /ZM90/ Alle Nutzer sollen grundlegende Informationen zu Räumen und Lehrveranstaltungen erhalten können.

/ZM100/ Das System soll intuitiv bedienbar sein.

2.2 Wunschkriterien

/ZW10/ Alle Nutzer des Systems sollten organisatorische Änderungen kurzfristig erkennen können.

/ZW20/ Für Dozenten sollte ein personalisierter Live-Ticker zur Verfügung stehen.

/ZW30/ Für Studenten sollte ein personalisierter Login-Bereich bereitgestellt werden um die Sammlung der Lehrveranstaltungen persistent speichern zu können sowie einen personalisierten Live-Ticker zu integrieren.

2.3 Abgrenzungskriterien

/ZA10/ Das Produkt soll zunächst lediglich von Universitätsmitarbeitern und Studenten am Standort Erba eingesetzt werden. Ein hochschulweiter Einsatz ist aktuell nicht vorgesehen.

/ZA20/ Ein Zugriff auf das Systems via Internet vom Heimrechner der Nutzer aus ist nicht vorgesehen.

3 Produkteinsatz

(Mario Serno)

Der geplante Produkteinsatz hat wesentliche Auswirkungen auf die funktionale Mächtigkeit und auf die Qualitätsmerkmale [vgl. [Bal96]]. Nachfolgend werden deshalb der Anwendungsbereich, die Zielgruppen und Betriebsbedingungen beschrieben.

3.1 Anwendungsbereiche

/P10/ Das Produkt wird intern an der Otto-Friedrich-Universität Bamberg eingesetzt.

3.2 Zielgruppen

/P20/ Zielgruppen des Produktes sind Lehrstühle, deren Mitarbeiter, die Hausverwaltungsmitarbeiter und Studenten der der Otto-Friedrich-Universität Bamberg am Standort Erba.

3.3 Betriebsbedingungen

/P30/ Das Produkt wird auf den Clients der Hausverwaltung, Lehrstühle und PC-Pools ausgeführt.

/P40/ Das System ist so zu konzipieren, dass die einzelnen Anwendungsinstanzen über eine gemeinsame Datenbasis (PostgreSQL-Server) kommunizieren können.

/P50/ Die tägliche Betriebszeit des Produktes erstreckt sich Montag bis Freitags jeweils von 08:00 bis 20:00 Uhr. Eine Nutzung am Wochenende ist nicht vorgesehen.

/P60/ Zu Beginn jedes Semesters wird die Lauffähigkeit des Produktes durch einen Administrator über einen Zeitraum von drei Wochen beobachtet. Nach Ablauf dieses Zeitraums ist ein unbeaufsichtigter Betrieb vorgesehen.

4 Produktumgebung

(Denis Hamann)

Im Folgenden wird die Umgebung beschrieben, in der die finale Software eingesetzt werden soll. Die Produktumgebung umfasst die Basismaschinen und die Systemumgebung, getrennt nach Software, Hardware und Produktschnittstellen. Zusätzlich werden im Bereich Orgware die zugehörigen organisatorischen Randbedingungen beschrieben.

4.1 Software

Die Software beschreibt die vorzuhaltende Software-Umgebung um eine problemlose Ausführung des Programms sicherzustellen.

- /PU10/ Als Betriebssystem kommt Windows 7 (Professional, x64 - 64bit) zum Einsatz. Dieses hat die neusten Updates vorzuweisen und eine gängige sowie aktuelle Antiviren Lösung installiert zu haben.
- /PU20/ Im Bereich der Laufzeitumgebung wird auf die Java Virtual Machine Version 7 von Oracle gesetzt. Hierzu ist das entsprechende Java Runtime Environment (JRE) in Version 7 vorzuhalten. Es muss sichergestellt werden, dass die entsprechenden Klassenpfade, sowie Systempfade gesetzt sind, sodass der Prozess java als auch javaw im CLI bekannt ist. Darüber hinaus müssen .jar-Dateien mit der JRE verknüpft sein um einen Start der Anwendung per Doppelklick zu gewährleisten. Entsprechende Einstellungen sind notfalls in der Registry vorzunehmen [vgl. Kapitel 12.1].
- /PU30/ Die Datenbank welche in Verbindung mit der Software eingesetzt wird, stellt PostgreSQL in Version 9.2.1 dar. Es muss sichergestellt werden, dass die betreffenden PCs welche die neue UnivIS Software verwenden sollen für die Datenbank freigeschaltet sind (Freigabe der jeweiligen IPs).
- /PU40/ Zusätzlich soll sichergestellt werden, dass entsprechende Netzwerk-Einstellungen (Firewalls, Router) eine ordnungsgemäße Verbindung zwischen Anwendungs-PC und Datenbank erlauben.
- /PU50/ Das Programm wird auf die Swing-Komponenten von Java setzten, daher ist es notwendig sicher zustellen, dass eine entsprechende Shell vorhanden ist. Als Oberfläche wird auf die Standard Fensteroberfläche von Windows 7 (shell: explorer.exe) gesetzt. Kiosk-Systeme sind als Benutzeroberfläche nicht vorgesehen.

4.2 Hardware

/PU60/ Die Software soll auf IBM-Kompatiblen Computern der Intel x86 Architektur ausgeführt werden. Um sicher zu gehen, dass die Anwendung ausreichend schnell ausgeführt wird, werden nachfolgende Systemvoraussetzungen empfohlen:

CPU: 1Ghz

RAM: 1GB

HDD: 100MB

Peripherie: Maus & Tastatur

Zusätzlich muss sichergestellt werden, dass ein entsprechender Netzwerkanschluss für die Verbindung zum Datenbankserver vorhanden ist.

/PU70/ Entsprechende Netzwerkhardware (Router, Switch, Patchpanel, Firewall, Netzkabel) zur Verbindung der Computer mit dem Datenbankserver werden vorausgesetzt.

4.3 Orgware

/PU80/ Für die Entwicklung der Software wird vorausgesetzt, dass jeweils ein involvierter Mitarbeiter der unterschiedlichen Organisationsbereiche der Universität zur Beantwortung von fachspezifischen Fragen zur Verfügung steht. Darüber hinaus ist für Arbeiten vor Ort ein entsprechender Platz zu stellen.

/PU90/ Benutzerhandbücher werden, sofern benötigt, in elektronischer Form (PDF) ausgeliefert. Sie beschreiben die grundlegenden Funktionen der Software.

/PU100/ Der Auftraggeber stellt sicher, dass dem Auftragsnehmer entsprechende Informationen zur Verfügung gestellt werden, um eine ordnungsgemäße Autorisierung und Authentifizierung der Benutzer sicherzustellen.

/PU110/ Die Datenbank (PostgreSQL) muss auf einem entsprechenden Server installiert sein und als Dienst laufen. Es muss sicher gestellt werden, dass entsprechende Router und Firewalls konfiguriert sind und eine Verbindung in beide Richtungen möglich ist, um eine Persistierung der Daten zu ermöglichen. Die Bereitstellung, sowie entsprechende Verfügbarkeit stellt der Auftraggeber sicher

/PU120/ Es wird davon ausgegangen, dass die Hausverwaltung, sowie die Dozenten entsprechende Informationen zu Räumen und Lehrveranstaltungen korrekt und zeitnah in das System einpflegen um eine sinnvolle Nutzung zu ermöglichen.

4.4 Produkt-Schnittstellen

/PU130/ Das Produkt beinhaltet lediglich eine Schnittstelle zur Datenbank wie in Kapitel 4.1 beschrieben. Über diese findet die Persistierung der Daten, sowie die Abfrage komplexer Anfragen statt. Schnittstellen mit der zukünftig geplanten Weboberfläche, sowie Mobilen Anwendung erfolgt nicht über die Software direkt sondern über die gemeinsame Datenbank. Die dort vorliegende Datenstruktur ist allen weiteren Anwendungen bekannt und ermöglicht so eine reibungslose Interaktion zwischen den Systemen. Parallele Sitzungen der Software werden ebenfalls über die Datenbank synchron gehalten. D.h. trägt Benutzer1 an Workstation1 eine neue Lehrveranstaltung ein, so ist diese auch in Workstation2 bei Benutzer2 bei entsprechender Einstellung zu sehen. Eine direkte Kommunikation zwischen den einzelnen Instanzen der Software findet nicht statt.

4.5 Zukünftige Entwicklungen

/PU140/ Wie bereits im Lastenheft angemerkt ist vorgesehen die Software später über den Standort hinaus zu verwenden. Neben der Ausweitung der Standorte sollen die Zugriffsmöglichkeiten auf das System um ein Webinterface, sowie eine Mobile Anwendung erweitert werden. Diese Entwicklungen werden in der erstellten Software berücksichtigt und entsprechend umgesetzt.

/PU150/ Um sicherzustellen, dass die spätere Erweiterung problemlos möglich ist, erhält der Auftraggeber entsprechend eine Dokumentation des Quellcodes, sowie der Datenbank bzw. dessen konkrete Relationen. Für das Ziel das bestehende UnivIS zu einem späteren Zeitpunkt zu ersetzen sollte dementsprechend ein Change-Management Ansatz gefahren werden, in diesem unter anderem auch die Schulung der späteren Benutzer erfolgen sollte.

5 Produktfunktionen

(Hannes Stadler)

5.1 Prozesse mit Dateneingabe

/F01/ **Prozess mit Dateneingabe:** Benutzer-Authentifizierung

Akteur: Alle

Beschreibung: Alle Benutzer des Systems haben die Möglichkeit sich mittels einer Login-Funktion am System anzumelden. Hierdurch werden Sie einer der möglichen Benutzergruppen zugeordnet. Nicht angemeldete Benutzer gelten als Mitglieder der Benutzergruppe „Studenten“.

Eingabevalidierung: Nutzer müssen eine gültige Nutzerkennung mit dazu passendem Passwort eingeben. Diese werden von der Verwaltung festgelegt. Es wäre wünschenswert, wenn Studenten sich in naher Zukunft auch am System anmelden könnten [siehe /FW61/] und hierzu ihre übliche Hochschulkennung (baXXXXXX) und Passwort verwenden könnten. Im Falle von falschen Zugangsdaten erhält der Benutzer eine Meldung, die ihn über den Fehlversuch informiert. Es gibt nur eine Meldung, sodass für ihn nicht ersichtlich ist, ob das Passwort oder die Nutzerkennung ungültig ist.

/F10/ **Prozess mit Dateneingabe:** Verwaltung von Raumdaten

Akteur: Verwaltung

Beschreibung: Mitglieder der Verwaltung können Räume erstellen, bearbeiten und löschen.

Eingabevalidierung: Es können beliebig viele Räume erstellt, gelöscht oder bearbeitet werden. Bei der Erstellung darf die Raumnummer (siehe Datenobjekt /D10/) nicht schon im System vorhanden sein (die systeminterne Raum ID ist für Nutzer nicht selbst wählbar, dies gilt auch im Folgenden), anderenfalls wird der Nutzer mit einer Fehlermeldung darauf hingewiesen. Die übrigen Attribute eines Raums [siehe /D10/] werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen. So kann eine „Raumkapazität“ bspw. nur einer ganzzahligen positiven Zahl entsprechen.

/F20/ Prozess mit Dateneingabe: Verwalten von Lehrveranstaltungen**Akteur:** Dozenten

Beschreibung: Dozenten können Lehrveranstaltungen, die sie oder andere Mitarbeiter ihres Lehrstuhls in einem Semester halten eintragen, bearbeiten und löschen. (Diese sind nur für angehörige ihres Lehrstuhls sichtbar und bearbeitbar. Für Studenten werden sie erst sichtbar und verwendbar, wenn für sie ein Raum gebucht wurde und dies bestätigt wurde.)

Eingabevalidierung: Es können beliebig viele Lehrveranstaltungen erstellt, gelöscht oder bearbeitet werden. Doppelerstellungen sind möglich, falls dies in speziellen Fällen von Dozenten benötigt wird (durch die systeminterne ID ist ein fehlerfreier Systembetrieb gewährleistet). Die übrigen Attribute einer Lehrveranstaltung [siehe /D50/] werden sofern möglich nach Logik und passender Größe überprüft. Bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/FW21/ Prozess mit Dateneingabe: Verwalten von Live-Ticker Meldungen**Akteur:** Dozenten und Verwaltung

Beschreibung: Dozenten und Mitglieder der Verwaltung können Meldungen erstellen, die Studenten im Live-Ticker zu sehen bekommen. Es lässt sich ein Zeitraum eingeben, in dem die Meldung sichtbar ist. Wünschenswert wäre hierbei, dass Studenten über die Authentifizierung personalisierte Meldungen (zu einer Lehrveranstaltung, einem Raum oder einem Lehrstuhl) angezeigt bekommen.

Eingabevalidierung: Es sollten beliebig viele Live-Ticker Meldungen erstellt, gelöscht oder bearbeitet werden können. Die übrigen Attribute einer Ticker-Meldung [siehe /DW80/] werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellungsvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

/F30/ Prozess mit Dateneingabe: Anfragen von Räumen**Akteur:** Dozenten

Beschreibung: Dozenten können ihre Lehrveranstaltung für ein Semester zu einer bestimmten Zeit an einem bestimmten Werktag in einem Raum eintragen. Eine Zeiteinheit beträgt dabei 2 Stunden, kann immer nur voll belegt werden und die Zählung der Einheiten geht von 8 Uhr morgens bis 20 Uhr abends (womit ein Tag aus 6 Einheiten besteht). Buchungen müssen seitens der Verwaltung bestätigt werden [siehe /F40/].

Eingabevalidierung: Es sind Anfragen für bereits eingetragene Lehrveranstaltungen möglich. Eine Anfrage umfasst genau eine Zeiteinheit zu einem Raum an einem Tag. Es sind beliebig viele Buchungen für eine Lehrveranstaltung möglich. Eine Buchung darf sich nicht mit einer bereits von der Verwaltung bestätigten Buchung im gleichen Raum zur gleichen Zeit am gleichen Tag überschneiden, sonst wird der Vorgang abgebrochen und dem Nutzer eine entsprechende Fehlermeldung gezeigt. Die übrigen Attribute einer Buchung [siehe /D60/] werden, sofern vom Dozenten angegeben und generell möglich, nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Dozent darauf hingewiesen.

/F40/ Prozess mit Dateneingabe: Verwaltung von Raumbelagungen inklusive Konfliktlösung**Akteur:** Verwaltung

Beschreibung: Mitglieder der Verwaltung können die Belegungswünsche zu Räumen seitens der Dozenten bestätigen, abweisen oder umlegen. Im Falle eines Konflikts, bzw. einer Ablehnung können sie dem Dozenten explizite Vorschläge machen (und hierzu Erläuterungen hinterlegen), die noch vor den vom System generierten automatischen Vorschlägen dem Dozenten angezeigt werden.

Eingabevalidierung: Es ist nur das zuweisen eines Status, wie schon bei /DW80/ spezifiziert, möglich. Dieser ist vom System vorgegeben, so dass Falscheingaben nicht möglich sind. Im Falle einer Ablehnung kann zusätzlich eine Nachricht übermittelt werden, die bezüglich der Größe validiert wird. Im Fehlerfall wird der Nutzer darauf hingewiesen und der Vorgang abgebrochen.

/F50/ Prozess mit Dateneingabe: Verwaltung von Nutzerdaten**Akteur:** Verwaltung**Beschreibung:** Mitglieder der Verwaltung können Benutzer (Dozenten und Verwaltungsmitglieder) in das System einpflegen, sie bearbeiten oder löschen.**Eingabevalidierung:** Das Eintragen und Bearbeiten umfasst die in /D30/ spezifizierten Attribute (außer dem letzten Login und dem Salt, welche automatisch vom System generiert werden, sowie der systeminternen ID). E-Mail und Benutzerkennung dürfen nicht schon im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Das Passwort wird beim Erstellen und Bearbeiten direkt zur Laufzeit zusammen mit einem benutzerspezifischen zufälligen Wert (dem „Salt“) in einen SHA512-Hash umgewandelt und nur als solcher persistent gespeichert. Es muss mindestens 8 Zeichen umfassen. Alle Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.**/F51/ Prozess mit Dateneingabe:** Verwaltung von Lehrstühlen**Akteur:** Verwaltung**Beschreibung:** Mitglieder der Verwaltung können Lehrstühle in das System einpflegen, sie bearbeiten oder löschen.**Eingabevalidierung:** Das Eintragen und Bearbeiten umfasst die in /D20/ spezifizierten Attribute. Der Lehrstuhlname darf nicht bereits im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Alle Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.**/F60/ Prozess mit Dateneingabe:** Sammlung von Lehrveranstaltungen**Akteur:** Studenten**Beschreibung:** Studenten können Lehrveranstaltungen sammeln (diese werden nicht persistent im System gespeichert). Überschneidungen (gesammelte Lehrveranstaltungen die zur gleichen Zeit stattfinden) sind grundsätzlich möglich, das System generiert jedoch einen Warnhinweis.**Eingabevalidierung:** Das System bietet nur die Möglichkeit eine öffentliche Veranstaltung als hinzuzufügen oder diese wieder zu entfernen. Eine vom Benutzer erstellte Eingabe und somit eine Validierung ist nicht von Nöten. Wünschenswert wäre die persistente Speicherung dieser Zuordnungen wie sie durch die Funktion /FW61/ möglich wäre und im Datenobjekt /DW70/ spezifiziert ist.

/FW61/ **Prozess mit Dateneingabe:** Profil-Verwaltung (inkl. Speichermöglichkeiten von Sammlungen)

Akteur: Studenten

Beschreibung: Studenten sollten die Möglichkeit bekommen, ein eigenes Profil anzulegen, welches persistent gespeichert wird und mit welchem ihre Sammlungen dauerhaft verknüpft werden.

Eingabevalidierung: Die Profil-Bearbeitung umfasst die Attribute E-Mail, Passwort, Vor- und Nachname des Datenobjekts /D30/ für Studenten. Das Passwort wird dabei wie in /F50/ beschrieben gehasht und muss mindestens 8 Zeichen umfassen. Die E-Mail Adresse darf nicht bereits im System vorhanden sein, anderenfalls erscheint eine entsprechende Fehlermeldung und der Vorgang wird abgebrochen. Alle betreffenden Attribute werden sofern möglich nach Logik und passender Größe überprüft und bei fehlerhaften Eingaben wird ein Bearbeitungs- oder Erstellvorgang abgebrochen, sowie der Nutzer darauf hingewiesen.

5.2 Listen

/F70/ **Liste:** Lehrveranstaltungen eines Lehrstuhls

Akteur: Dozenten

Beschreibung: Dozenten können sich im System eine Liste von allen Lehrveranstaltungen, die ihrem Lehrstuhl zugeordnet sind anzeigen lassen. Hierbei steht ein Filter zur Verfügung, der nur bestimmte Semester, bestimmte Dozenten und den Veröffentlichungsstatus anzeigt.

Enthaltene Daten: Name der Veranstaltung, Dozent, Semester, Tag(e), Zeit(en), gebuchter Raum, Freigabestatus, *wünschenswert wäre die Anzahl eingeschriebener Studenten*

/F80/ **Liste:** Räume

Akteur: Verwaltung

Beschreibung: Mitglieder der Verwaltung können sich im System eine Liste mit allen Räumen anzeigen lassen. Hierbei steht ein Filter zur Verfügung, der es erlaubt nach Stockwerk, Raumnummer, Kapazität und den sonstigen Attributen der Räume zu filtern.

Enthaltene Daten: Raumnummer, Stockwerk, Kapazität, PC-Plätze, Overheads, Vizualizern, Beamer, etc.

- /F81/ **Liste:** Raumanfragen
 Akteur: Verwaltung
 Beschreibung: Mitglieder der Verwaltung können sich im System eine Liste mit allen Raumanfragen, also noch nicht freigegebenen oder abgelehnten Anfragen anzeigen lassen. Hierbei steht ein Filter zur Verfügung, der es erlaubt nach Semester, Freigabestatus, Tag, Raum oder Lehrstuhl zu filtern. Noch nicht bearbeitete Raumanfragen (weder freigegeben noch abgelehnt) werden an erster Stelle angezeigt.
 Enthaltene Daten: Name der Veranstaltung, Dozent, Lehrstuhl, Semester, Tag(e), Zeit(en), gebuchter Raum, Freigabestatus, ggf. Verweis auf überschneidende Buchungen, *wünschenswert wäre die Anzahl eingeschriebener Studenten*
- /F90/ **Liste:** Nutzer (Dozenten, Lehrstühle, Hausverwaltung)
 Akteur: Verwaltung
 Beschreibung: Mitglieder der Verwaltung können sich im System eine Liste mit allen Benutzern des Systems anzeigen lassen. Es steht ein Filter zur Verfügung, der es erlaubt nach Vor- oder Nachnamen sowie der Nutzerzugehörigkeit zu filtern.
 Enthaltene Daten: Nutzerkennung, Vorname, Nachname, Nutzerzugehörigkeit (Lehrstuhl, Verwaltung oder ggf. auch Student), Zahl der Lehrveranstaltungen im aktuellen Semester, Datum des letzten Login
- /F100/ **Liste:** Lehrveranstaltungen
 Akteur: Studenten (und andere)
 Beschreibung: Studenten (aber auch anderen Nutzern) können sich im System eine Liste aller durch die Verwaltung freigegebenen Lehrveranstaltungen anzeigen lassen. Es steht ein Filter zur Verfügung, der es erlaubt nach Semester, Lehrstuhl, Dozent, Name der Veranstaltung, Tag, Zeit und Raum zu filtern.
 Enthaltene Daten: Name der Veranstaltung, Dozent, Lehrstuhl, Semester, Tag(e), Zeit(en), gebuchter Raum, *wünschenswert wäre die Anzahl eingeschriebener Studenten*
- /F110/ **Liste:** Live-Ticker Meldungen
 Akteur: Studenten (sowie wünschenswerterweise Verwaltung und Dozenten mit Bearbeitungsfunktionen)
 Beschreibung: Studenten (aber auch anderen Nutzern) bekommen im System automatisch Live-Ticker Meldungen angezeigt, welche die nächsten (falls möglich den Stundenplänen entsprechenden) Lehrveranstaltungen anzeigen, sowie wünschenswerterweise auch von Dozenten oder Verwaltung zu diesen erstellte Meldungen.
 Enthaltene Daten: Lehrveranstaltungsname, Raum, Zeit oder Meldungstexte

5.3 Berichte

/F120/ **Bericht:** Raumbelegung/Raumplan

Akteur: Verwaltung (und wünschenswerterweise andere Nutzer)

Beschreibung: Mitglieder der Verwaltung (und wünschenswerterweise auch andere Nutzer) können sich eine Übersicht zur Belegung eines bestimmten Raumes in einem Semester anzeigen lassen, welche als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar ist.

/F130/ **Bericht:** Stundenplan

Akteur: Studenten

Beschreibung: Studenten können sich eine Übersicht zu ihrer Sammlung an Lehrveranstaltungen in einem Semester anzeigen lassen. Überschneidungen werden hierbei gesondert hervorgehoben. Die Übersicht (die einem Stundenplan gleich kommt) ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

/F140/ **Bericht:** Lehrstuhlveranstaltungen/plan

Akteur: Dozenten

Beschreibung: Dozenten können sich eine Übersicht generieren, in der sie sehen, wann alle Lehrveranstaltungen ihres Lehrstuhls in einem bestimmten wählbaren Semester sind. Überschneidungen werden hervorgehoben. Wünschenswert wäre, wenn auch Überschneidungen mit Veranstaltungen anderer Lehrstühle markiert werden. Die Übersicht ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

/F150/ **Bericht:** Dozentenveranstaltungen/Dozentenwochenplan

Akteur: Dozenten

Beschreibung: Dozenten können sich einen Wochenplan ihrer zu haltenden Lehrveranstaltungen in einem semester generieren. Überschneidungen werden hervorgehoben. Wünschenswert wäre auch eine Visualisierung von Überschneidungen anderer Dozenten. Die Übersicht ist als Ansicht im System aufrufbar und als PDF exportierbar bzw. druckbar.

6 Produktdaten

(Hannes Stadler)

- /D10/ **Datentyp:** Räume
Attribute: Raum ID (*systemintern*), Raumnummer, Gebäude, Stockwerk, Sitzplätze, PC-Plätze, Beamer, Visualizer, Overheads, Tafeln, Whiteboards
- /D20/ **Datentyp:** Lehrstühle
Attribute: Lehrstuhl ID (*systemintern*), Lehrstuhlname, Lehrstuhlinhaber, Fakultät, Gebäude (für die zukünftige Erweiterung), (Haupt-)Stockwerk
- /D30/ **Datentyp:** Benutzer
Attribute: Benutzer ID (*systemintern*), Benutzerkennung, Passwort (Hash), Salt, E-Mail, Benutzerzugehörigkeit (Verwaltung, betreffender Lehrstuhl, wünschenswerterweise auch Student), Vorname, Nachname, letzter Login
- /D31/ **Datentyp:** Dozenten
Attribute: Dozenten ID (*systemintern*), Benutzer ID, Lehrstuhl ID
- /D50/ **Datentyp:** Lehrveranstaltungen
Attribute: Veranstaltungs ID (*systemintern*), Benutzer ID (Dozent), Veranstaltungskurzbezeichnung, Veranstaltungsname, Semester, Benötigte SWS, Art (Vorlesung|Übung|Tutorium), Freigabe durch Dozent, Beschreibung, Erwartete Teilnehmer
(Tag, Zeiteinheiten, etc. wird über „Raumbelegung [siehe /D60/]“ ermittelt, wo für jede Zeiteinheit ein Eintrag erstellt wird und Veranstaltungen mehrere Einträge pro Semester buchen können.)
- /D60/ **Datentyp:** Raumbelegungen (aller Freigabestatus-Arten)
Attribute: Belegungs ID (*systemintern*), Veranstaltungs ID (Lehrveranstaltung), Raum ID, Semester, Tag, Zeiteinheit, Freigabestatus (unbearbeitet|freigegeben|abgelehnt|gegenvorschlag), Freigabenachricht (Falls ein Vorschlag abgelehnt wurde und dies nun ein reservierter Vorschlag des Status „gegenvorschlag“ ist), Kommentar
- /DW70/ **Datentyp:** Studentensammlung
Attribute: Studenten-Sammlungs ID (*systemintern*), Benutzer ID (Student), Belegungs ID (freigegebene Lehrveranstaltung)

/DW80/ **Datentyp:** Ticker-Nachricht

Attribute: Meldungs ID (*systemintern*), Meldungstext, Start-Datum, End-Datum

/DW90/ **Datentyp:** Ticker-Zuordnung

Attribute: Ticker-Zuordnungs ID (*systemintern*), Meldungs ID, Belegungs ID, Veranstaltungs ID, Lehrstuhl ID, Raum ID

Im Structured-Entity-Relationship-Modell (SERM, siehe Abbild 1) wird der Zusammenhang der oben spezifizierten persistent zu speichernden Daten verdeutlicht.

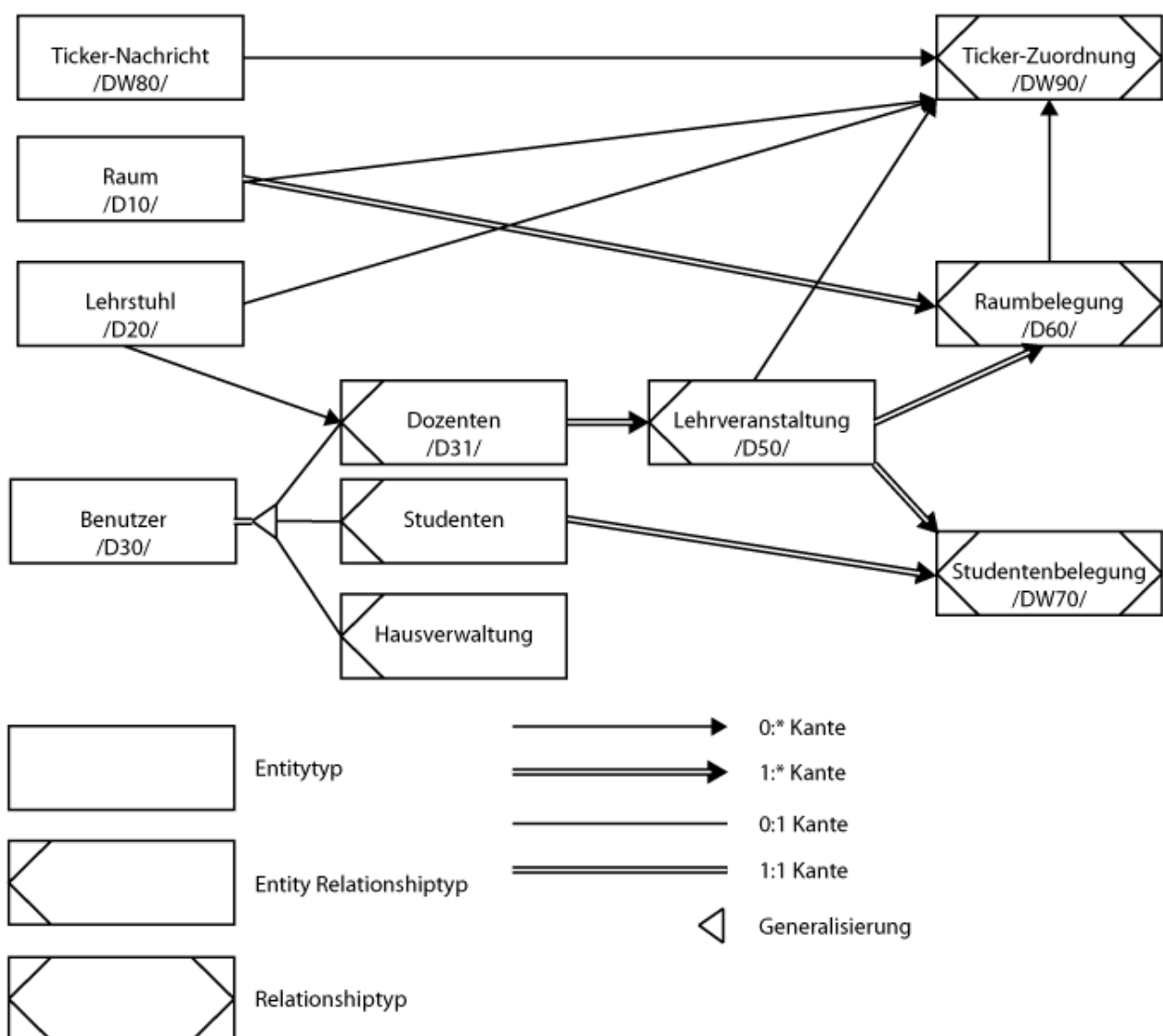


Abbildung 1: SERM der Daten-Architektur

Wie in der Abbildung zu sehen ist, existieren Räume, Lehrstühle, Benutzer und Ticker-Nachrichten als eigenständige Einheiten (Entity). Benutzer können einem Lehrstuhl zugeordnet werden, womit sie der Klasse „Dozent“ entsprechen. Sie können aber auch der Klasse „Verwaltung“ oder „Student“ angehören und somit keiner anderen Einheit zugeordnet werden. Lehrveranstaltungen müssen einem Dozent zuzuordnen sein. Eine Raumbellegung ist keine Einheit sondern eine Zuordnung (Relationship) und muss einer Lehrveranstaltung und einem Raum zugeordnet werden können. Die Zuordnung Studentenbellegung muss einem Student und einer Lehrveranstaltung zugeordnet werden können. Ticker-Nachrichten können einem Raum, einer Lehrveranstaltung, einem Lehrstuhl, einer Raumbellegung oder allen Einheiten davon zugeordnet werden, nichts davon ist jedoch zwingend (existiert kein Zuordnungseintrag zu einer Ticker-Nachricht, so ist sie global gültig).

7 Produktleistung

(Hannes Stadler)

- /L10/ Alle Daten, die im vorherigen Abschnitt aufgelistet sind, müssen, sofern sie realisiert werden, persistent mittels einer SQL-Datenbank (erreichbar und verwaltet durch einen PostgreSQL-Server) gespeichert werden.
- /L20/ Bei allen Prozessen mit Dateneingabe-Funktionen erhalten Nutzer aussagekräftige Fehlermeldungen, sollte ihr Prozess nicht ordnungsgemäß durchgeführt werden können.
- /L30/ Die Realisierung erfolgt als Java-Anwendung, so dass ein möglicher Betrieb auf allen gängigen PCs der Universität sichergestellt ist.

/L40/ Die Programmarchitektur richtet sich nach einem Drei-Schichten-Modell, wobei in einer Schicht die Anwendungslogik („A-Schicht“), in einer weiteren die nötigen Funktionen und Schnittstellen zur Datenbank-Kommunikation („D-Schicht“) und in einer letzten die für den Nutzer sichtbaren Teile, mit denen er interagiert, („K-Schicht“) untergebracht sind.

Das Programm ist zudem in logische Bereiche (Module) untergliedert, die innerhalb der ADK-Struktur existieren und die sich bei der Implementierung eines GUI (Graphical User Interface) nach dem MVC-Konzept (Model-View-Controller Konzept) richten. So kann ein Modul verschiedene Views (etwa Fenster, Tabs oder Eingabemasken) besitzen, die innerhalb der „K-Schicht“ liegen, innerhalb der „A-Schicht“ sind die den Views zugrundeliegenden Models, sowie deren Controller und ggf. zusätzlich nötige Anwendungslogik angesiedelt. Sofern Teile der Datenbank exklusiv bestimmten Modulen zuzuordnen sind, kann sich ein Modul auch bis in die „D-Schicht“ erstrecken und hier eigene Funktionen und Schnittstellen bereitstellen.

Diese Architektur erlaubt dabei einerseits maximale Flexibilität in Bezug auf zukünftige Erweiterungen und gewährleistet zudem durch viele klar voneinander abgegrenzte Bereiche ein effektives Arbeiten im Team bei der Implementierung.

Die erste Version der Anwendung soll aus folgenden Modulen bestehen:

CoreGUI: Hier wird die Start-Routine sowie der Startbildschirm der Anwendung realisiert. Die Funktion /F01/ und /F60/ werden von diesem Modul umgesetzt.

Verwaltung: Hier wird die Anzeige für Verwaltungsmitglieder umgesetzt, die die Funktion /F10/, wünschenswerterweise /FW21/, /F40/, /F50/, /F51/, /F80/ und /F90/ realisiert.

Dozenten: Hier wird die Anzeige für Dozenten umgesetzt, die die Funktion /F20/, wünschenswerterweise /FW21/, /F30/, /F140/, /F150/ und /F70/ realisiert.

Stundenplan: Hier wird die Anzeige für den Stundenplan von Studenten umgesetzt, die die Funktion /F130/ realisiert.

Raumplan: Hier wird die Anzeige für Mitglieder der Verwaltung (und ggf. andere) von Räumen und deren Belegung umgesetzt, die die Funktion /F120/ realisiert.

Studentenprofil (wünschenswert): Hier wird die Anzeige für Studenten umgesetzt, die die Funktion /FW61/ realisiert.

Alle Module sind von „CoreGUI“ abhängig.

- /L50/ Es wird eine Nutzerauthentifizierung realisiert, die es gewährleistet, dass nur berechnigte Nutzer Änderungen durchführen. Um Missbrauch ausschließen zu können, sollte allerdings eine Middleware (die großteils die Model- und Controller-Schicht abdeckt) zwischen Datenbank und Client (der hauptsächlich die View-Schicht umsetzt) realisiert werden. In diesem Fall sollte die Kommunikation zwischen Client und Middleware durch SSL (mithilfe von javax.net.ssl.* und javax.rmi.ssl.*) abgesichert werden.
- /L60/ Jede Interaktion sollte im Mittel zur Durchführung nicht länger als 3 Sekunden brauchen.
- /L70/ Über die Datenbank als externe Komponente, die auf einen Server ausgelagert wird, wird sichergestellt, dass ein gemeinsamer Betrieb mit anderen Client-Programmen, die ebenfalls die Datenbank nutzen, ermöglicht wird. So wird es beispielsweise möglich sein, Anwendungen für mobile Endgeräte oder Webanwendungen zu entwerfen, die sich in Echtzeit die Datenbasis teilen.

8 Benutzeroberfläche

(Anna Kupfer)

Dieser Abschnitt beschäftigt sich mit der grundlegenden Gestaltung der Benutzeroberfläche für die unterschiedlichen Nutzer [vgl. [fS12a], [fS12b], [Bal96] [Bal09], [Bra12], [Kor07]]. Zunächst werden allgemeine Anforderungen definiert worauf spezifische zum Bildschirm- und Drucklayout sowie der Tastaturbelegung folgen. Das Kapitel schließt mit einer ausführlichen Betrachtung der Dialogstruktur und untermalt diesen Punkt mit Screenshots zum ersten GUI-Prototypen.

- /B10/ Fensterlayout, Dialogstruktur und Mausbedienung orientieren sich wenn möglich am Windows-Gestaltungs-Regelwerk [vgl. [Pre95]].
- /B11/ Sämtliche Daten sind passwortgeschützt und dürfen nur von autorisierten Mitarbeitern des Lehrstuhls bearbeitet werden.

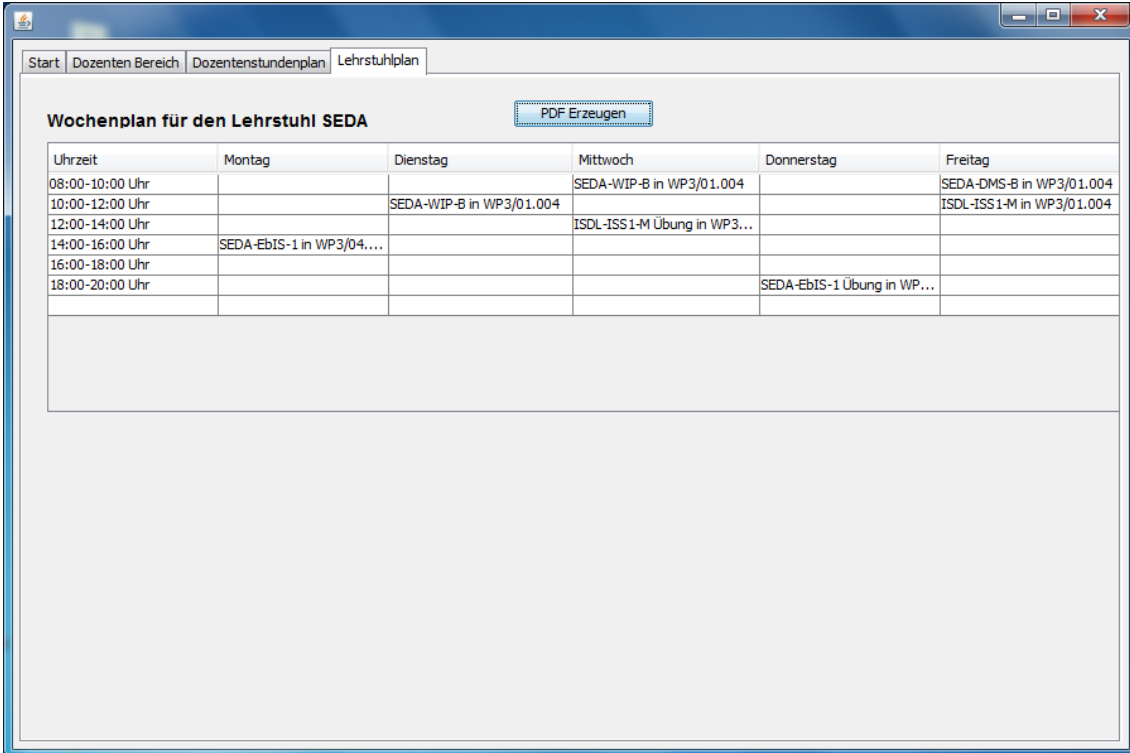
8.1 Bildschirmlayout

- /B20/ Eine übersichtliche Gestaltung der Funktionen und intuitive Nutzung durch ein angepasstes Bildschirmlayout soll vorherrschen.
- /B30/ Standardmäßig startet das Programm mit einer Suchmaske. Weitere Funktionen sind via Tabs und einem Login erreichbar.
- /BF40/ Individuelle Anpassungsfähigkeit des Bildschirmlayouts an die Fenstergröße sollte möglich sein.

8.2 Drucklayout

- /B50/ Nicht angemeldete Nutzer können über die Auswahl verschiedener Lehrveranstaltungen einen Stundenplan im PDF-Format in Din-A4 Größe erzeugen lassen [siehe Abbildung 4].
- /B60/ Die Hausverwaltungsnutzer können Raumpläne im PDF-Format in Din-A4 Größe erzeugen lassen (ähnlich der Studenten-, Dozenten-, und Lehrstuhlstundenpläne).
- /B70/ Mitarbeiter der Lehrstühle können personen- oder lehrstuhlbezogene Wochenpläne in einem PDF-Format in Din-A4 Größe exportieren[siehe Abbildungen 3, 2].

Hier einige Beispiele zur Ausgabe des Stundenplans für Studenten, Dozenten und Lehrstühle:

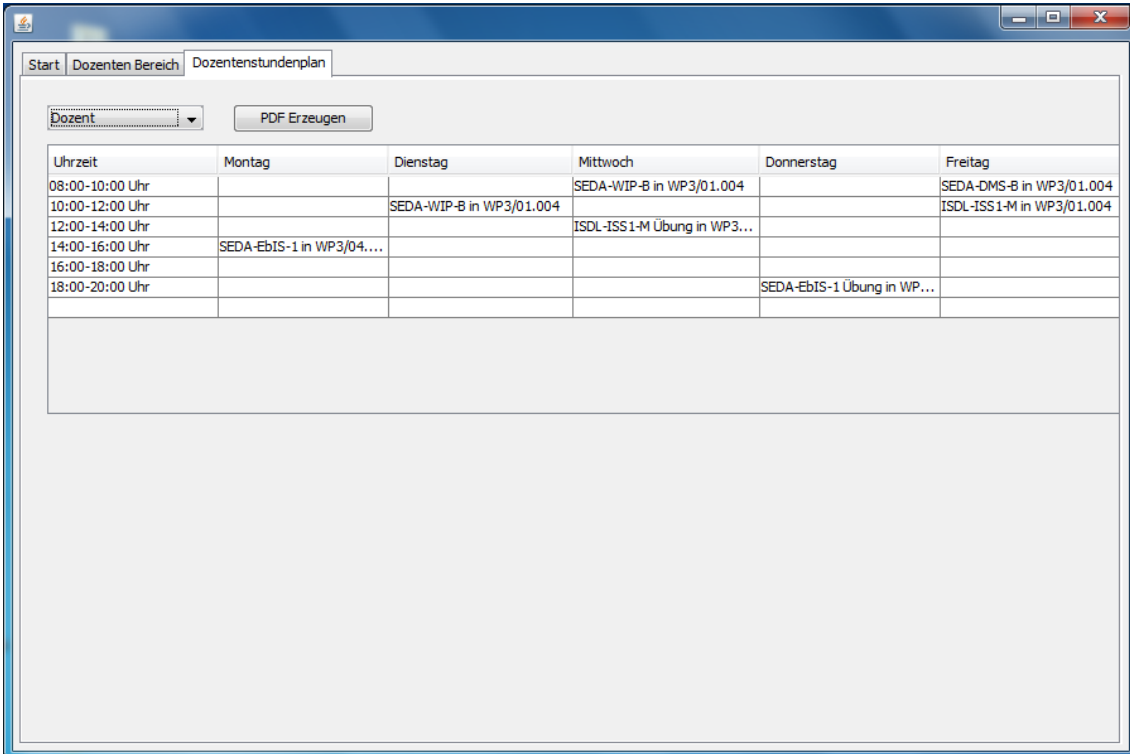


Wochenplan für den Lehrstuhl SEDA

Uhrzeit	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
08:00-10:00 Uhr			SEDA-WIP-B in WP3/01.004		SEDA-DMS-B in WP3/01.004
10:00-12:00 Uhr		SEDA-WIP-B in WP3/01.004			ISDL-ISS1-M in WP3/01.004
12:00-14:00 Uhr			ISDL-ISS1-M Übung in WP3...		
14:00-16:00 Uhr	SEDA-EbIS-1 in WP3/04....				
16:00-18:00 Uhr					
18:00-20:00 Uhr				SEDA-EbIS-1 Übung in WP...	

PDF Erzeugen

Abbildung 2: Lehrstuhlplan



Dozent: PDF Erzeugen

Uhrzeit	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
08:00-10:00 Uhr			SEDA-WIP-B in WP3/01.004		SEDA-DMS-B in WP3/01.004
10:00-12:00 Uhr		SEDA-WIP-B in WP3/01.004			ISDL-ISS1-M in WP3/01.004
12:00-14:00 Uhr			ISDL-ISS1-M Übung in WP3...		
14:00-16:00 Uhr	SEDA-EbIS-1 in WP3/04....				
16:00-18:00 Uhr					
18:00-20:00 Uhr				SEDA-EbIS-1 Übung in WP...	

Abbildung 3: Dozentenstundenplan

Uhrzeit	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
08:00-10:00 Uhr			SEDA-WIP-B in WP3/01...		SEDA-DMS-B in WP3/01.004
10:00-12:00 Uhr		SEDA-WIP-B in WP3/01.004			ISDL-ISS1-M in WP3/01.004
12:00-14:00 Uhr			ISDL-ISS1-M Übung in ...		
14:00-16:00 Uhr	SEDA-EbIS-1 in WP3/04...				
16:00-18:00 Uhr					
18:00-20:00 Uhr				SEDA-EbIS-1 Übung in ...	

Abbildung 4: Stundenplananzeige, rollenunabhängig

8.3 Tastaturbelegung

- /B80/ Die Benutzeroberfläche ist auf eine Bedienung mittels Tastatur und Maus auszu-
legen. Benutzer in der Studentenrolle können keine Eingaben mittels der Tastatur
vornehmen. Dozenten und die Hausverwaltung benötigen die Tastatur um ver-
schiedene Funktionen auszuführen.
- /BF81/ Mögliche individuelle, nicht dem Windows Standard entsprechende, Tastaturbe-
legungen sind ein mögliches Wunschkriterium.

8.4 Dialogstruktur

- /B90/ Zu Beachten ist: ISO 9241-10: 1996 bzgl. der ergonomischen Anforderungen für
Bürotätigkeiten mit Bildschirmgeräten, Teil 10: Grundsätze der Dialoggestaltung.
- /B91/ Folgende Rollen sind zu unterscheiden:

Rolle	Rechte
Student	(F01), F60 , FW61, F130
Dozent	F01, F20, FW21, F30, F70, F140, F150
Verwaltungsangestellter	F01, F10, FW21, F40, F50, F51, F80, F90,
Alle	F01, F100, F110, F120

Nachfolgend wird die Dialogstruktur durch den GUI-Prototypen verdeutlicht, um die Benutzeroberfläche vorzustellen und zu erläutern. Im Anschluss an den Programmstart erscheint die Startseite des UnivIS 2.0 [siehe Abbildung 5]. Hier können unangemeldete Nutzer (vor allem Studenten) über Dropdown Menüs verschiedene Lehrveranstaltungen suchen und diese anschließend mittels dem „+“-Button die Lehrveranstaltungen zur Stundenplan-Sammlung hinzufügen [siehe /F60/]. Schon bei der ersten Verwendung des „+“-Buttons öffnet sich ein neuer Tab, in welchem der Stundenplan gemäß der selektierten und gesammelten Lehrveranstaltungen ausgegeben wird. Anhand der Radiobuttons (Lehrveranstaltungen und Räume) können gleichzeitig Belegungen einzelner Räume selektiert und weiterhin im Stundenplan-Tab visualisiert werden [siehe /F100/, /F130/]. Im linken Bereich der Benutzeroberfläche befindet sich der Live-Ticker [siehe /F110/], über welchem anstehende Lehrveranstaltungen oder sonstige Informationen ausgegeben werden. Im unteren linken Bereich befindet sich der Login [siehe /F01/] für Dozenten und die Hausverwaltung (evtl. auch einmal für Studenten) [siehe /FW61/] mittels Benutzerkennung/name und Passwort. Im Falle einer erfolgreichen Anmeldung als Hausverwaltungsmitglied folgt die Startseite für Hausverwaltungsmitarbeiter [siehe Abbildung 6]. Sie haben einen unmittelbaren Überblick über alle Raumanfragen und können diese sogleich bearbeiten (freigeben, ablehnen oder etwaige Konflikte lösen) [siehe /F80/, /F40/]. Im linken Bereich bleibt Platz für einen personalisierten Live-Ticker. Weiter unten können die Benutzer auf weitere Aktionen zugreifen. Werden weitere Aktionen ausgeführt, gelangt der Benutzer zu weiteren Listen und kann jeweils Lehrstühle, Nutzer und Räume verwalten (hinzufügen, bearbeiten oder löschen) [siehe /F10/, /F50/, /F51/, /F90/, /F100/ und siehe Abbildungen 8 und 9]. Bei der Raumverwaltung kann zusätzlich ein Raumplan erstellt werden [siehe /F120/ und siehe Abbildung 10]. Eine letzte abdingbare Funktion wäre die Bearbeitung des LiveTickers um bspw. Dozentenverwaltungstechnische Nachrichten anzeigen lassen zu können [siehe /FW21/].

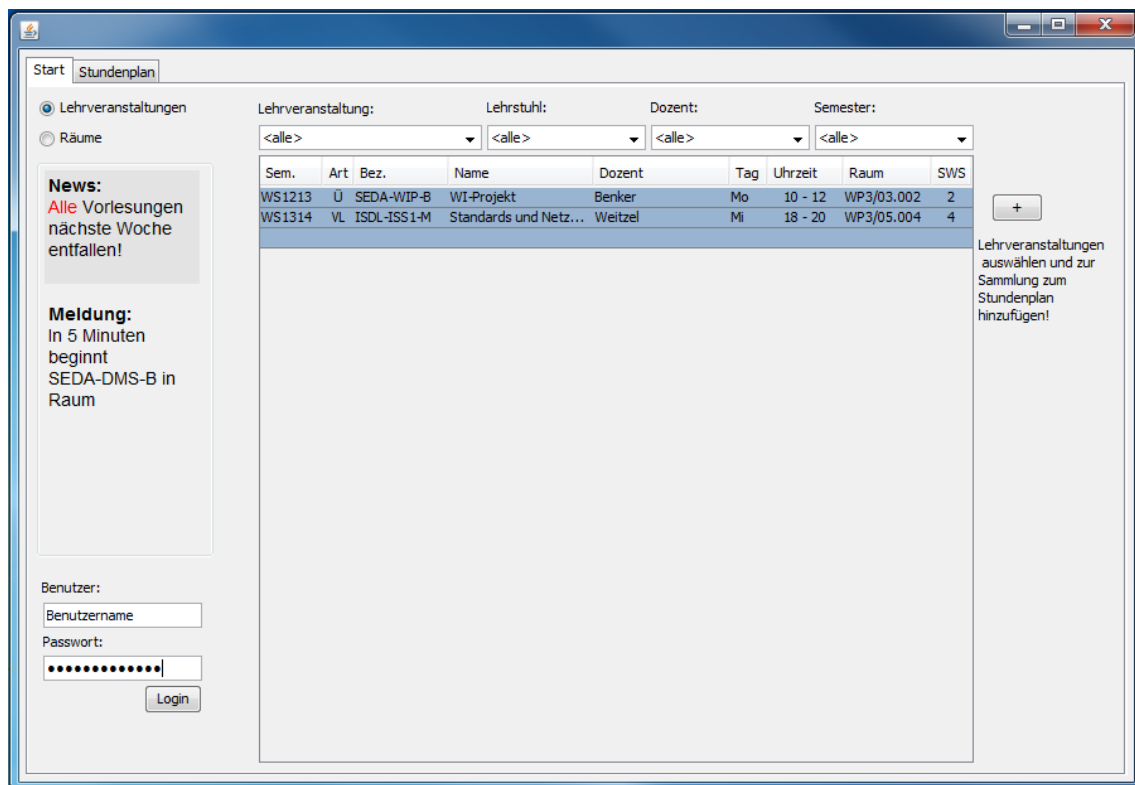


Abbildung 5: GUI Startseite, rollenunabhängig

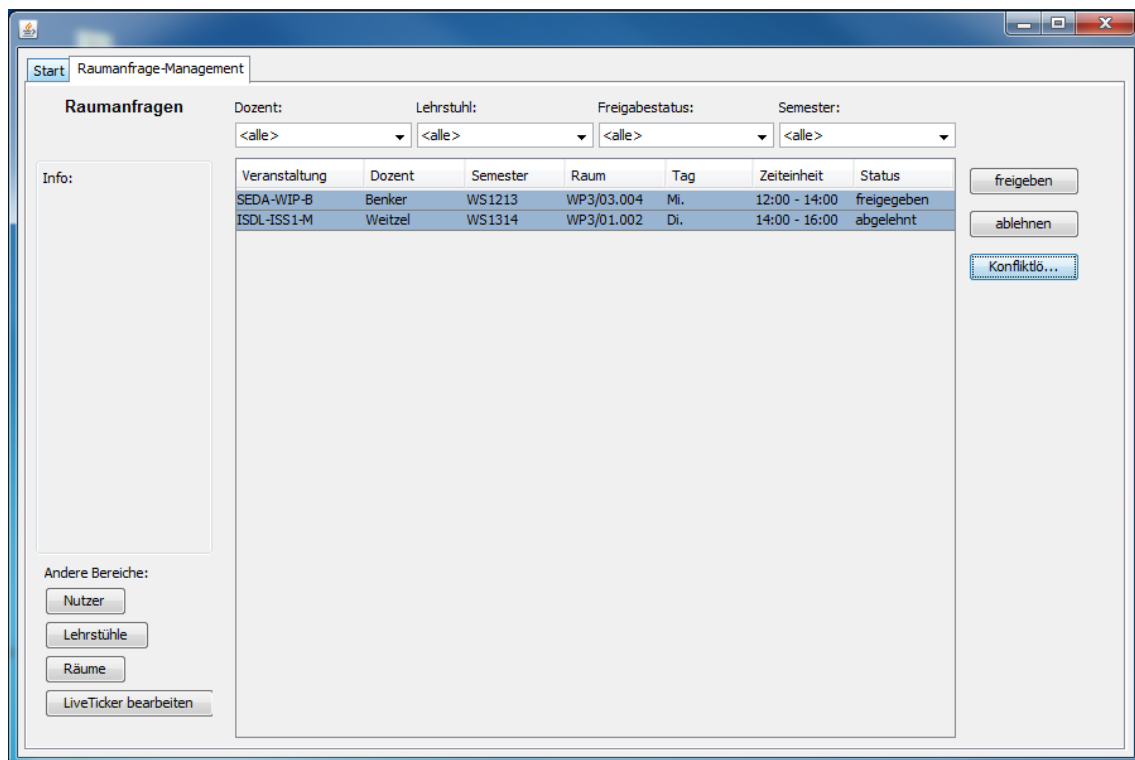


Abbildung 6: Startseite für authentifizierte Benutzer der Hausverwaltung

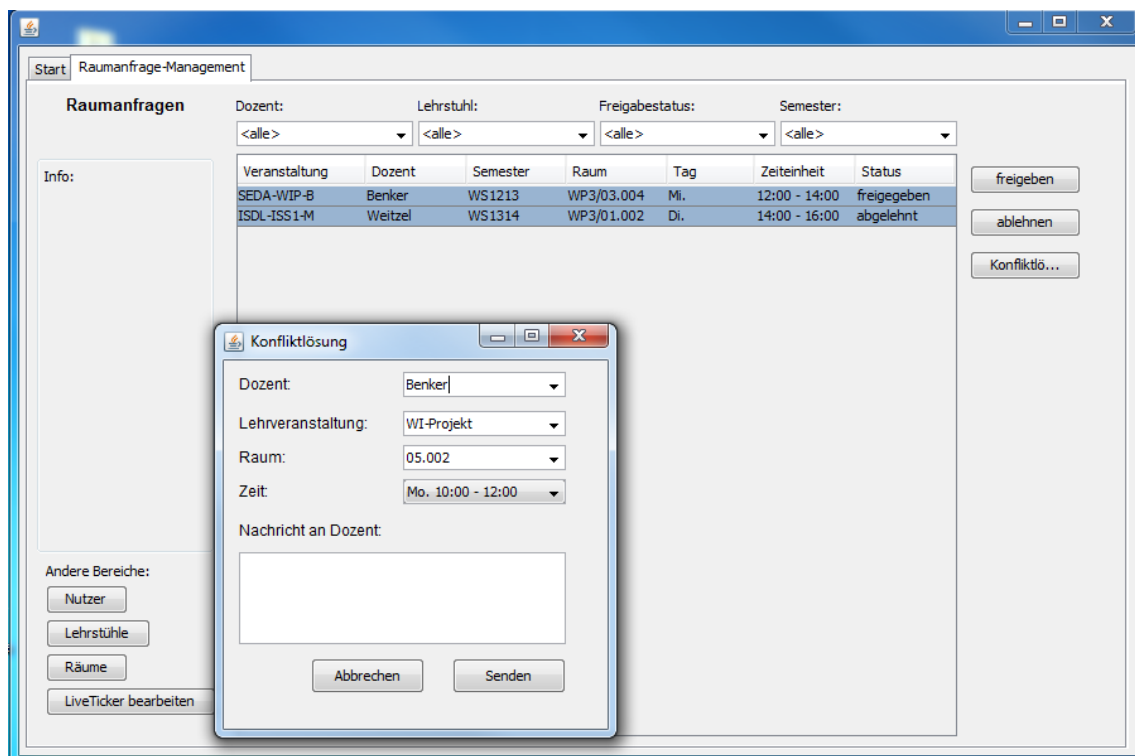


Abbildung 7: Konfliktlösung der Raumanfragen

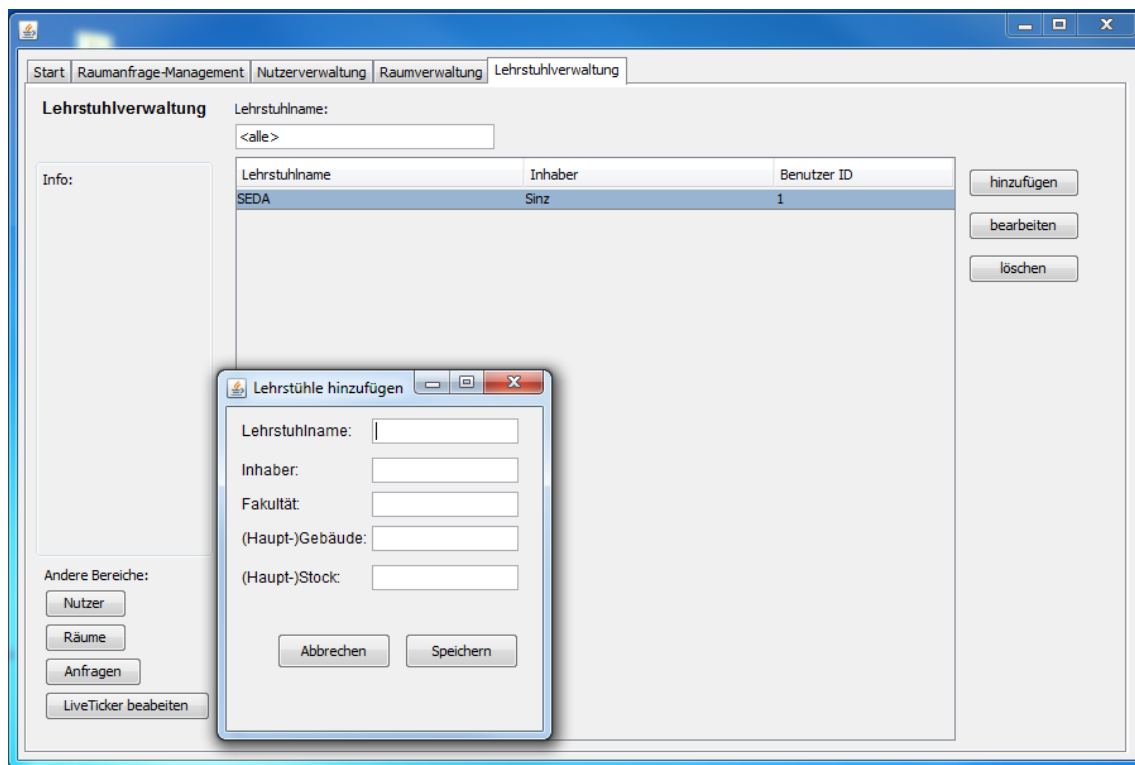


Abbildung 8: Verwaltung der Lehrstühle

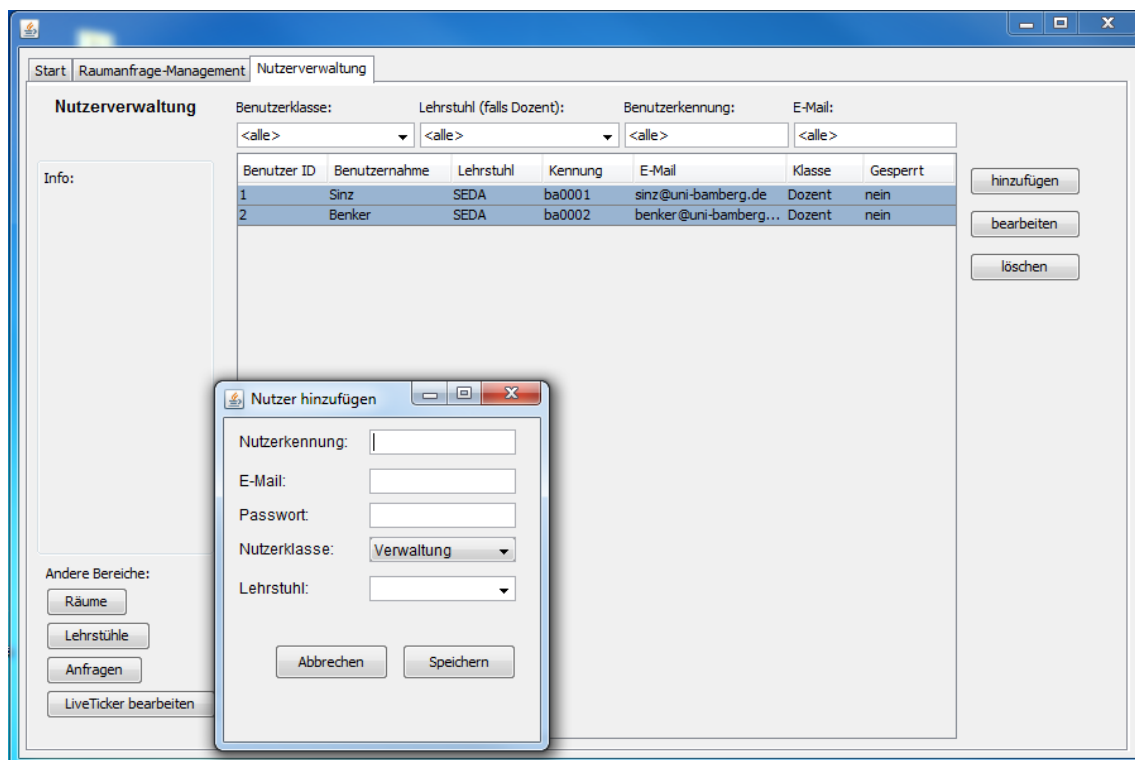


Abbildung 9: Verwaltung der Nutzer

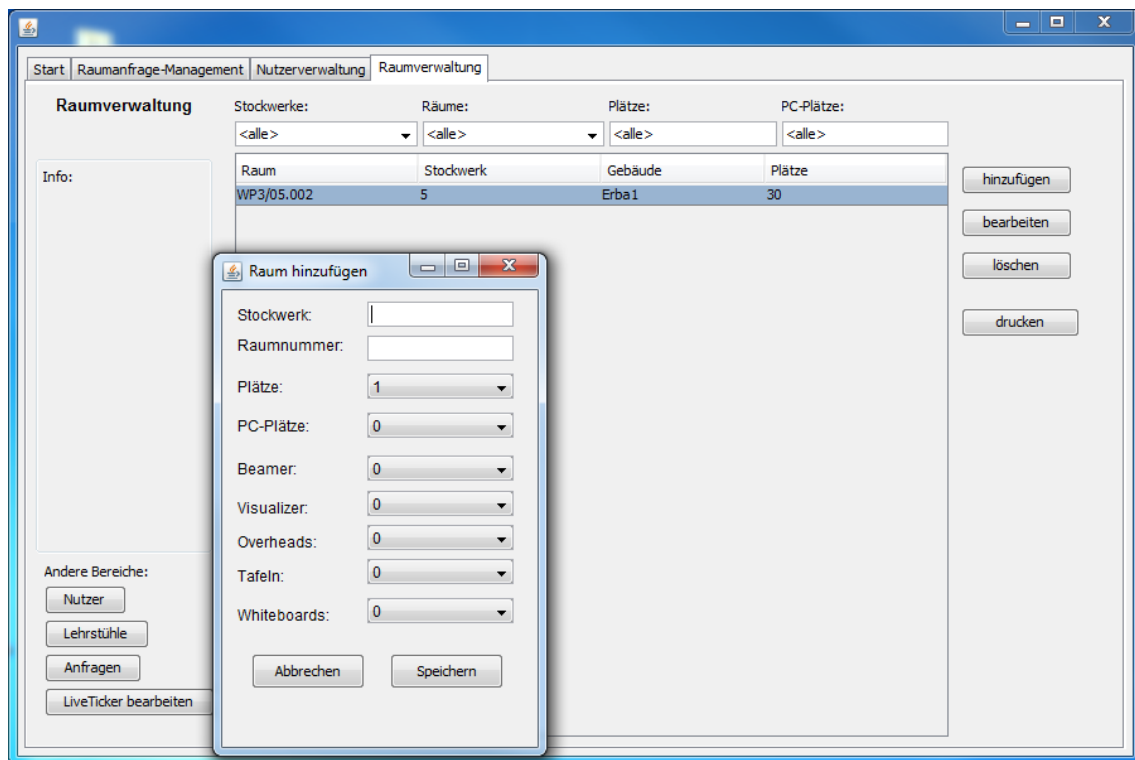


Abbildung 10: Verwaltung der Räume

Die personalisierte Startseite der Dozenten beinhaltet zwei Listen [siehe Abbildung 11]. In der oberen werden alle Veranstaltungen visualisiert und können nach Lehrstühlen, Dozenten und Semester gefiltert werden. Außerdem wird der Veröffentlichungsstatus sowie die erfolgreich Raumanfrage angezeigt[siehe /F70/]. Eine weitere Liste gibt zusätzliche Auskunft zum Status der Raumanfragen/Raumzuordnungen. Bei einer erfolgreich durchgeführten Raumanfrage kann der jeweilige Dozent seine Lehrveranstaltung anschließend veröffentlichen. Die mit der Raumanfrage bezogenen Nachrichten der Hausverwaltung werden im Live-Ticker (links) angezeigt. Um Lehrveranstaltungen zunächst überhaupt hinzuzufügen wird der Button „hinzufügen“ gewählt und in einem neuen Fenster können Lehrstuhl und Dozent ausgewählt werden und somit die Bezeichnung neuer Veranstaltungen gesetzt werden [siehe /F20/ und siehe Abbildung 12]. Um der Lehrveranstaltung einen Raum zuzuordnen, kann eine Raumanfrage über den Button „Raumanfrage“ [siehe Abbild 13] spezifiziert und versendet werden [siehe /F30/]. Hierfür kann entweder ein gewünschter Raum oder nur die besonderen Anforderungen an die Hausverwaltung zur weiteren Bearbeitung übermittelt werden. Außerdem können die Aktionen eigener Stundenplan [siehe /F150/ und siehe Abbildung 3] und Lehrstuhlplan [siehe /F140/ und siehe Abbildung 2] sowie eine LiveTicker Bearbeitung (um bspw. Studenten veranstaltungs- oder lehrstuhlsbezogene Nachrichten anzeigen lassen zu können) [siehe /FW21/] gewählt werden.

Lehrveranstaltungen:

Lehrstuhl: <alle> Dozent: <alle> Semester: <alle>

Bezeichnung	Dozent	Semester	SWS	Erw....	Termine	Raumanfrage freige...	Öffentlich
SEDA-WIP-B	Benker	WS1213	4	30	2	nein	nein

hinzufügen
bearbeiten
Raumanfrage
veröffentlic...

Raumzuordnungen:

Lehrstuhl: <alle> Dozent: <alle> Lehrveranstaltung: <alle> Semester: <alle> Veröffentlichungsstatus: <alle>

Bezeichnung	Dozent	Semester	Tag	Zeit	Raum	Status
SEDA-WIP-B	Benker	WS1213	Do.	12:00 - 14:00	WP3/02.001	abgelehnt
SEDA-WIP-B	Benker	WS1213	Mo.	14:00 - 16:00	WP3/04.002	wartend

zurückziehen

Stundenplan
Lehrstuhlplan
LiveTicker bearbeiten

Abbildung 11: Startseite für Dozenten

Lehrveranstaltungen hinzufügen

Lehrstuhl: SEDA
Dozent: Wolf
Bezeichnung:
Art: Übung
Semester: WS1213
SWS: 2
Erw. Teilnehmer:
Beschreibung:

Abbrechen OK

Abbildung 12: Verwaltung der Lehrveranstaltungen durch die Dozenten

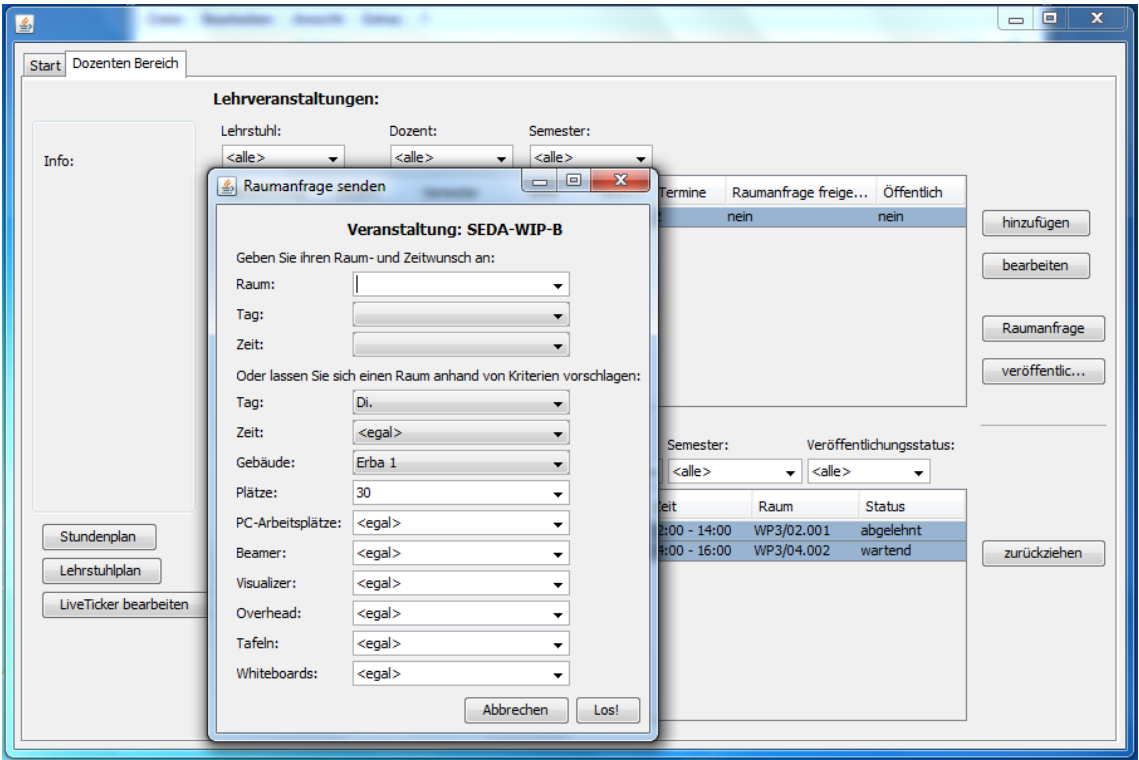


Abbildung 13: Raumanfragen durch die Dozenten

9 Qualitätszielbestimmung

(Christian Hindelang)

Das zu entwerfende Programm (UnivIS 2.0) hat folgende Qualitätsziele, die nachfolgend erläutert werden [vgl. [fS12c]].

- /Q10/ Das Programm soll eine korrekte Funktionalität aufweisen, d. h. es soll richtige Ergebnisse bei den Ein- und Ausgaben liefern.
- /Q20/ Die Software soll zuverlässig arbeiten. Es sollten möglichst keine Fehler passieren.
- /Q30/ Es soll gewährleistet sein, dass die Daten sicher verwahrt sind und vor unberechtigtem Zugriff geschützt sind.
- /Q40/ Im Falle eines Datenrückgewinns soll gewährleistet sein.
- /Q50/ Eine leichte und intuitive Bedienung von UnivIS 2.0 ist von Nöten, da eine Vielzahl von potenziellen Nutzern auf das System zugreifen können sollen. Die Heterogenität der Studenten und des Personals macht es erforderlich, dass sich alle Beteiligten schnell an die Bedienung des Programms gewöhnen und keine langen Einarbeitungsphasen oder sogar Kurse notwendig werden.
- /QW60/ Damit die Nutzer des Systems effizient arbeiten können, soll im Mittel die Ergebnisausgabe eines Stundenplans drei Sekunden in Anspruch nehmen.
- /Q70/ Die im Lastenheft gemachte Angabe, dass das System eventuell universitätsweit und auf mobilen Plattformen zum Einsatz kommen kann, macht es erforderlich, dass die Änderbarkeit und Anpassung des Programms schnell erfolgen kann. Maximal drei Monate sollten hierfür aufgewendet werden müssen.
- /Q80/ Schlussendlich muss das Programm an andere Windowsversionen anpassbar sein. Dies ist durch eine .java-Datei gewährleistet, welche plattformunabhängig läuft.

10 Testszenarien - Christian Hindelang

(Christian Hindelang)

Damit festgestellt werden kann, ob die Software auch einwandfrei funktioniert, müssen verschiedene Testszenarien durchlaufen werden. Die Tests werden auf verschiedenen Ebenen ausgeführt. Unterschieden wird hierbei zwischen dem Modultest, Integrationstest, Systemtest und letztendlich dem Abnahmetest. Im Folgenden werden nun diese Testszenarien genauer beschrieben und dargelegt, wie sie am Programm durchgeführt werden.

- /T10/ Im Komponententest, welcher wie auch der nachfolgende Integrationstest unter White-Box-Tests fällt, beinhaltet das Testen von einzeln abgrenzbaren Teilen (Modulen) des Gesamtprogramms. Das Testen erfolgt bei White-Box-Tests in der Entwicklungsumgebung Eclipse, wo der Quelltext sichtbar ist. Einzelne Klassen, Funktionen und Unterprogramme werden im Modultest dem Tester sequenziell unterworfen und auf Fehlerfreiheit geprüft. Dies umfasst unter anderem eine korrekte Ausgabe von Uhrzeiten. Auch SQL-Abfragen an die Datenbank ob die Passwortabfrage korrekt abgespeichert wurden, soll der Test beinhalten.
- /T20/ Nachdem im Modultest jedes Modul einzeln überprüft wurde, wird nun im Integrationstest dazu übergegangen, auch die Interaktion einzelner Module mit anderen Modulen zu überprüfen. Hier soll u. a. getestet werden, ob der Live-Ticker auch alle Lehrveranstaltungen anzeigt, die in Kürze starten. Zudem soll überprüft werden, ob die Stundenplanausgabe die richtigen Inhalte ausgibt und diese anschließend in der PDF korrekt formatiert ausgegeben werden.
- /T30/ Die dritte Teststufe beherbergt den Systemtest, bei dem das System gegen sämtliche Anforderungen aus dem Pflichtenheft getestet wird. Ab hier erfolgt das Testen nicht mehr in der Entwicklungsumgebung, sondern wird direkt am lauffähigem Programm ausgeführt. Dies wird auch als Black-Box Test bezeichnet. Hier werden dann ca. 200 Lehrveranstaltungen zu Testzwecken angelegt. Diese werden vom Systemverwalter manipuliert und auch die Suche nach Alternativterminen für Veranstaltungen wird hierbei überprüft. Ebenso soll getestet werden, ob die persistente Speicherung der Stammdaten (z. B. Name und Uhrzeit der Lehrveranstaltung) funktioniert. Hierfür wird ein Systemabsturz simuliert, bei dem aufeinanderfolgend erst dem Rechner, dann dem Server der Strom entzogen wird und beide somit neu booten müssen. Anschließend wird eine neue Datenbankabfrage ausgehend des Systems durchgeführt, um zu überprüfen, ob die Daten noch vorhanden sind.

/T40/ Schlussendlich erfolgt der Abnahmetest. Dieser wird allerdings nicht von unserem Softwareunternehmen ausgeführt, sondern erfolgt durch den Abnehmer selbst. Durch den tatsächlichen Gebrauch können hierbei noch Fehler entdeckt werden, die bislang nicht auftauchten, da unter Umständen eine andere Reihenfolge der Menüaufrufe, etc. stattfindet. Der Endabnehmer muss schließlich die Abnahme der Software bestätigen, damit die Tests erfolgreich beendet werden können. Besteht das Programm diesen letzten Test nicht, soll von Seiten der SF-GmbH nachgebessert werden.

11 Entwicklungsumgebung

(Denis Hamann)

Im Folgenden wird die Umgebung beschrieben, in der die spätere Software entwickelt wird. Die Entwicklungsumgebung umfasst Anforderungen an Softwarewerkzeuge, die zur Entwicklung verwendete Hardware, organisatorische Randbedingungen. Des weiteren wird auf Schnittstellen der Entwicklungsumgebung eingegangen.

11.1 Software

Die Software beschreibt die eingesetzte Software-Umgebung um eine effiziente und effektive Erstellung des Programms sicherzustellen.

/EU10/ Als Betriebssystem kommt Windows 7 (Professional | Ultimate, x64 - 64bit) zum Einsatz. Dieses hat die neusten Updates vorzuweisen und eine gängige aktuelle Antiviren Lösung installiert zu haben.

/EU20/ Als Laufzeitumgebung wird auf die Java Virtual Machine Version 7 von Oracle gesetzt. Hier kommt das Software Development Kit (SDK) in entsprechender Version zum Einsatz. Zum Testen des Programms ist das entsprechende Java Runtime Environment in Version 7 vorzuhalten. Es muss sichergestellt werden, dass die entsprechenden Klassenpfade, sowie Systempfade gesetzt sind, sodass der Prozess java als auch javaw im CLI bekannt ist. Darüber hinaus müssen .jar-Dateien mit der JRE verknüpft sein um einen Start der Anwendung per Doppelklick zu gewährleisten. Entsprechende Einstellungen sind notfalls in der Registry vorzunehmen [vgl. Kapitel 12.1].

/EU30/ Als IDE kommt die Eclipse Plattform mit den entsprechenden Java Entwicklungs-Komponenten zum Einsatz. Installierte Addons sind der 'WindowBuilder Pro' [vgl. <http://www.eclipse.org/windowbuilder/download.php>] sowie optional das Subclipse Plugin für das Versionsverwaltungstool Subversion.

/EU40/ Für die Versionsverwaltung wird Subversion in der neusten Version (1.7.7) eingesetzt. Auf den einzelnen Entwicklern PCs kommt das Tool Tortoise SVN [<http://tortoisesvn.net/downloads.html>] Während der Entwicklung wird ein SVN-Server von Google verwendet [<http://code.google.com/p/seda-wip-sfgmbh/>]. Die finale Abgabe des Quellcodes erfolgt über das vom Auftraggeber zur Verfügung gestellte-SVN Repository.¹

/EU41/ Zum Erstellen der Dokumentation/Benutzerhandbücher wird auf das textverarbeitungsprogramm LaTeX gesetzt.

- /EU42/ Für die konkrete Projektplanung kommt MS Project zum Einsatz.
- /EU50/ Für den Dateiaustausch innerhalb des Entwicklungsteams wird auf den Cloud-Dienst Dropbox gesetzt. Für die interne Kommunikation wird auf Skype gesetzt, da hier bereits entsprechende Konten vorhanden sind. Hangouts von Google+ kommen nicht in Frage, da entsprechende Accounts vorausgesetzt und die Videoübertragungsfunktion primär nicht benötigt werden.
- /EU60/ Analog zur Produktumgebung wird die gleiche Datenbank verwendet, welche in Verbindung mit der Software eingesetzt wird, PostgreSQL in Version 9.2.1. Es muss sichergestellt werden, dass die betreffenden PCs welche die neue UnivIS Software verwenden sollen sowie die Entwickler-PCs für die Datenbank freigeschaltet sind (Freigabe der jeweiligen IPs).
- /EU70/ Zusätzlich muss sichergestellt werden, dass entsprechende Netzwerk-Einstellungen (Firewalls, Router) eine ordnungsgemäße Verbindung zwischen Anwendungs-PC, Entwicklungs-PCs und Datenbank erlauben.
- /EU80/ Zur Entwicklung der in Kapitel 4.1 beschriebenen Oberfläche des Programm wird wie bereits angedeutet auf Swing gesetzt. Um die GUI schneller zu erstellen und testen wird das Plugin 'WindowBuilder Pro' der Eclipse Foundation verwendet. Das Programm wird auf die Swing-Komponenten von Java setzten, daher ist es notwendig sicher zustellen, dass eine entsprechende Shell vorhanden ist. Als Oberfläche wird auf die Standard Fensteroberfläche von Windows 7 (shell: explorer.exe) gesetzt. Kiosk-Systeme sind als GUI nicht vorgesehen.

11.1.1 Programmierstil

- /EU90/ Bei der Programmierung ist darauf zu achten, dass die Java Code Conventions eingehalten werden. [<http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>]. Zusätzlich sind alle Klassen entsprechend Javadoc konform zu kommentieren um die Dokumentation entsprechend automatisiert per Ant-Skript durchzuführen.

11.2 Hardware

/EU100/ Die Software soll auf IBM-Kompatiblen Computern der Intel x86 Architektur ausgeführt werden. Um sicher zu gehen, dass die Anwendung ausreichend schnell ausgeführt wird, werden nachfolgende Systemvoraussetzungen empfohlen:

CPU: 1Ghz

RAM: 1GB

HDD: 100MB

Peripherie: Maus & Tastatur

Zusätzlich muss sichergestellt werden, dass ein entsprechender Netzwerkanschluss für die Verbindung zum Datenbankserver vorhanden ist.

11.3 Orgware

/EU110/ Die Voraussetzungen für die Entwicklung wurden bereits in Kapitel 4.3 beschrieben. Darüber hinaus muss sichergestellt werden das ein ständiger Dialog zwischen Auftragsnehmer und Auftragsgeber stattfindet um eventuelle Detailfragen zu klären und entsprechend das Pflichtenheft zu korrigieren.

/EU120/ Des weiteren wird bei der Entwicklung auf das Wasserfallmodell zurückgegriffen. Hierfür verfügen alle Entwickler über entsprechende Kenntnisse. Ablauf und Ergebnisse der Phasen sind Ihnen bekannt. Alle Entwickler weisen entsprechende Kenntnisse über die Entwicklungssoftware und Hardware auf. Sofern Wissenslücken vorhanden sind, werden diese vor Beginn des Projektes entsprechend kompensiert.

11.4 Produkt-Schnittstellen

/EU130/ Für die Entwicklung der in Kapitel 4.4 beschriebenen Datenbankanbindung wird der entsprechende JDBC-Treiber für PostgreSQL verwendet [<http://jdbc.postgresql.org/download.html>]. Da die Interaktion zwischen den einzelnen Systemen durch die Datenbank stattfindet und keine spezielle Schnittstelle zwischen der geplanten Software und dem angekündigten Webinterface, sowie der mobilen Nutzung besteht, muss keine zusätzliche Schnittstelle implementiert werden.

12 Ergänzungen

(Denis Hamann)

12.1 Jar- Registry Extension

```
[HKEY_CLASSES_ROOT\.jar]
```

```
@="jarfile"
```

```
[HKEY_CLASSES_ROOT\jarfile]
```

```
@="Executable Jar File"
```

```
[HKEY_CLASSES_ROOT\jarfile\shell]
```

```
[HKEY_CLASSES_ROOT\jarfile\shell\open]
```

```
[HKEY_CLASSES_ROOT\jarfile\shell\open\command]
```

```
@="\"C:\\Program Files\\Java\\jre7\\bin\\javaw.exe\" -jar \"%1\" %*"
```


13 Glossar

API	Die API stellt eine dokumentierte Software-Schnittstelle dar, die von anderen Programmen aus genutzt werden kann.
CLI	Command Line Interface - Kommandozeile. Die Kommandozeile ist ein Eingabebereich für die Steuerung einer Software, die typischerweise im Textmodus abläuft.
DNS	Ermöglicht es Klarnamen in numerische IP Adressen (z.B. google-public-dns-a.google.com in 8.8.8.8 umzuwandeln).
GUI	Hierbei handelt es sich um die grafische Benutzeroberfläche.
IP	Ein Protokoll das für die Vermittlung von Daten dient.
ISO	Eine internationale Vereinigung von Normungsorganisationen.
JDBC	Hierbei handelt es sich um eine Datenbankschnittstelle für Java.
Klasse	Im Kontext der Programmierung handelt es sich hierbei um einen abgegrenzten Bereich (ein sogenanntes „Objekt“) mit bestimmten Attributen und Methoden.
LDAP	Ein Verzeichnisdienst um Abfragen und Modifikationen von Informationen zu erlauben.
ODBC	Hierbei handelt es sich um eine Datenbankschnittstelle von Microsoft.
RFC	RFCs sind eine Reihe von technischen und organisatorischen Dokumenten zum Internet, die sie zu einem Standard entwickelt haben.
Salt	Ein Salt (zu Deutsch „Salz“) wird bei der Erstellung einer Prüfsumme zu einem Passwort beigegeben, um zu gleichen Passwörtern unterschiedliche Prüfsummen zu erhalten. Ohne ein Salt wäre anhand der Prüfsumme das Passwort zu erkennen, wenn einmal bekannt ist, welche Prüfsumme zu welchem Passwort gehört.
SHA-256	SHA steht für „Secure Hash Algorithm“ wobei die angefügte Zahl die Länge der generierten Prüfsumme („Hash“) in Bit angibt.
Shell	Eingabe-Schnittstelle zwischen Computer und Benutzer - meist grafisch
SQL	Eine deskriptive Abfragesprache von Datenbanken.

TCP	Ein verbindungsorientiertes Protokoll, um Daten im Netzwerk zu transportieren.
UDP	Ein verbindungsloses Protokoll, um Daten im Netzwerk zu transportieren.

Literatur

- [Bal96] Helmut Balzert. *Lehrbuch der Software-Technik: Teil 1: Software-Entwicklung*. Spektrum Akademischer Verlag, Heidelberg, Germany, 1996.
- [Bal09] Helmut Balzert. *Lehrbuch der Software-Technik: Basiskonzepte und Requirements Engineering*. Spektrum, Heidelberg, 3. edition, 2009.
- [Bra12] Ina Schaefer TU Braunschweig. Pflichtenheft - Software Engineering I - WS2011/2012, 2012.
- [fS12a] Lehrstuhl für Softwaretechnik. Gliederungsschema eines Pflichtenheftes (1. Auflage), 2012.
- [fS12b] Lehrstuhl für Softwaretechnik. Gliederungsschema eines Pflichtenheftes (2. Auflage), 2012.
- [fS12c] Lehrstuhl für Softwaretechnik. Qualitätsmerkmale, 2012.
- [Kor07] Jurij Kormilez. 2.2 Pflichtenheft, 14.11.2007.
- [Pre95] Microsoft Press. *The Windows interface guidelines for software design*. Microsoft professional reference. Microsoft Press, 1995.