WiRe Bibliotheksmanagmentsystem

WiReLib

Software-Entwicklungspraktikum (SEP)
Sommersemester 2012

G r o b e n t w u r f

Auftraggeber
Technische Universität Braunschweig
Wissenschaftliches Rechnen
Prof. Hermann G. Matthies
Hans-Sommer-Straße 65
D-38092 Braunschweig

Betreuer: Elmar Zander

Auftragnehmer:

Name	E-Mail-Adresse
Eric Anders	eric.anders89@web.de
Johann Hong	johann.hong@googlemail.com
Jörn Hameyer	j.hameyer@tu-bs.de
Marco Melzer	marco.melzer@tu-braunschweig.de
Markus Dietrich	markus.dietrich@tu-bs.de
Philipp Offensand	PhilippOffensand@gmx.de
Stephan Sobol	stephan.sobol@web.de
Theodor van Nahl	t.nahl@tu-bs.de

Braunschweig, 24.04.2012

Versionsübersicht

Version	Datum	Autor	Status	Kommentar
...	...	...	...	...

Status: „in Bearbeitung“ oder „abgenommen“ Kommentar: hier eintragen, was geändert bzw. ergänzt wurde

Hinweis zum Template: Dieses Template enthült Hinweise, die alle kursiv geschrieben sind. Alles Kursivgeschriebene ist selbstverständlich bei Abgabe zu entfernen sind. Angaben in $<\ldots>$ sind mit dem entsprechendem Text zu füllen. Überzählige Kapitel, d.h. Kapitel, die nicht bearbeitet werden müssen, da sie nicht der Aufgabenstellung entsprechen, bitte entfernen.

Aufgabe des Grobentwurfs: Aufgabe dieses Dokumentes ist es, die Architektur des Systems zu beschreiben und die daraus resultierenden Pakete durch die Definition von Schnittstellen zu Komponenten auszubauen.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Hier ist die Arbeitsweise des Systems anhand von Aktivitätsdiagrammen darzustellen und kurz zu erläutern.

1.1 Projektdetails

Die Details der Aufgabenstellung bitte anhand von Aktivitätsdiagramme darstellen und kurz verbal beschreiben.

2 Analyse der Produktfunktionen

Im folgenden werden die Funktionen und Qualitätsmerkmale des System auf ihre Modulzugehörigkeit hin geprüft und eingeordnet. Dabei wird auf das Basis-Konstrukt von Django aufgebaut und die Funktionen bzw. Qualitätsmerkmale werden in unterschiedliche Kategorien eingeteil.

Hier zuerst die Kategorienen in die, Die einzelnen Funktionen und Qualitätsmerkmale eingeordnet werden.

- Build in Django
- Model einer Django-App
- Klassen bzw. Objectmethode
- Template
- View im Backend
- View im Frontend

2.1 Analyse von Funktion F100 (Anmeldung)

Diese Funktion ist eine Build-In-Funktion von Django. Innerhalb von Django werden die Benutzer durch eine eigenständige Klasse dargestellt. Diese Klasse muss erweitert werden um die zusätzlichen Daten der Benutzer zu speichern. Da Django eine Datenkbank zugrunde liegt müssen diese weiteren Daten in einer Extra Klasse mit Referenz auf die Benutzer gesetzt werden.

2.2 Analyse von Funktion F101 (Anmeldung über LDAP)

Diese Funktion muss in einer extra App dargestellt werden. Die LDAP-Anmeldung muss dabei die normale Anmeldung erweitern und im View evtl. durch eine Checkbox zur Verfügung gestellt werden.

2.3 Analyse von Funktion F102 (Anbindung an die Universitätsbibliothek)

2.4 Analyse von Funktion F103 (Mailtexte ändern)

Die Mailtexte sind Bestandteil einer veränderbaren aber nicht großen Tabelle an Basisinformationen wie auch andere Einstellungen für die Software. Um die Informationen bereit zu stellen muss eine neue Klasse erzeugt werden, das alle Informationen für die App bereit stellt.

2.5 Analyse von Funktion F201 (BibTeX import)

Ein oder mehrere Dokumente werden zum System hinzugefügt. Dafür wählt der Benutzer den entsprechenden View aus und läd über einen Upload-Dialog eine BibTeX-Datei hoch. Das System analysiert die Datei auf Validität, erzeugt Objekte aus den Daten die in eine Datenbank gespeichert werden.

Für die Funktion wird also ein View benötigt, der den Upload bereit stellt. Desweiteren werden die Dokument-Objekte benötigt, die auch von vielen weiteren Funktionen verwendet werden und die auch über Django die Datenbankeinträge verwalten.

2.6 Analyse von Funktion F201 (Webinterface-Import)

Der Benutzer wählt einen View der ein Interface bietet um die Daten für einzelne Dokumente

2.7 Analyse von Funktion F202 (Editieren von Dokumenten)

2.8 Analyse von Funktion F203 (Löschen von Dokumenten)

2.9 Analyse von Funktion F210 (Generelle Suche)

2.10 Analyse von Funktion F220: Ausleihe

Ein oder mehrere Dokumente werden von einem Benutzer ausgeliehen. Dabei wählt der Benutzer die gewünschten Dokumente aus und das System muss in der Datenbank nachprüfen, ob diese

derzeit ausleihbar sind. Der Status des Dokumentes müsste also Öösein. Wenn dies der Fall sein sollte, werden dem Benutzer diese Bücher zugewiesen.

2.11 Analyse von Funktion F221: Ausleihe an Externe

Die Ausleihe an Externe ist ähnlich gestaltet wie F220. Es existiert lediglich der Zusatz, dass der Benutzer einen Externen als eigentlichen Ausleiher angibt. Dafür wird beim Entleihen die Möglichkeit gegeben Daten über den Externen anzugeben, dessen Daten dann gespeichert und beim Entleihen vermerkt werden.

2.12 Analyse von Funktion F222: Ausleihe übertragen

Ein Dokument wechselt den Ausleihenden. Dabei muss ein Benutzer ein von ihm ausgeliehenes Dokument auswählen und einem anderen Benutzer übertragen. Das System vermerkt dieses Austausch dann in der Datenbank.

2.13 Analyse von Funktion F223: Ausleihe zurückgeben

Ein Benutzer braucht ein Dokument nicht mehr und gibt es in den Bestand der Bibliothek zurück. Auch hierfür muss der Benutzer als Entleiher des Dokumentes eingetragen sein. Wenn dies der Fall sein sollte, kann er *zurückgeben* wählen und das Dokument wird auch im Datensatz mit dem Status Övversehen.

2.14 Analyse von Funktion F224: (Optional) Ausleihe vermisst melden

Das Dokument befindet sich nicht mehr an dem laut Datenbank befindlichen Ort. Dann kann ein Benutzer ein Dokument auf der Dokumentenansicht als *vermisst* melden und dieses wird in der Datenbank mittels Statusänderung vermerkt. Dazu wird eine E-Mail an alle Benutzer verschickt oder alternativ eine Meldung auf der Homepage angezeigt.

2.15 Analyse von Funktion F225: Ausleihe verloren melden

Das Dokument ist auch nach einer Vermisstenmeldung nicht wieder aufgetaucht. Dann ist ein Bibliothekar berechtigt, es als *verloren gegangen* einzustufen. Das Dokument bekommt den entsprechenden Status und wird demnächst bei weiteren Suchanfragen ausgeschlossen.

2.16 Analyse von Funktion F226 (Ausleihfrist abgelaufen)

Die Mailtexte

2.17 Analyse von Funktion F227 (Ausleihhistory)

Diese Funktion ist eine Built-In-Funktion von Django. Es wird ein Filter über die Dokument-Objekte gelegt, damit nur bestimmte bzw. die gewünschten Dokumente und deren bisherige Ausleiher ausgegeben werden. Diese so erhaltenen Daten müssen nun noch durch ein Frontend-View an geigneter Stelle der Web-Seite angezeigt werden.

2.18 Analyse von Funktion F228 (Derzeitiger Leihender)

Da in Django die User durch eine eigene Klasse dargestellt werden, muss diese Klasse auf die Rechte des momentan angemeldeten Benutzers überprüft werden. Falls diese Rechte ausreichend sind, wird wiederum durch ein Queryset+Filter die jeweilige Literatur und der dazugehörige Leihende durch einen View ausgegeben. Damit ist auch diese Funktion Built-In-Django.

2.19 Analyse von Funktion F229 (Entleihliste)

2.20 Analyse von Funktion F230(-Export)

Die für diesen Export nötigen Informationen sind vollständig in der Datenbank enthalten und müssen mit Django ermittelt werden. Die so erhaltenen Daten werden dem Benutzer in einer View im -Format zum Kopieren angezeigt.

2.21 Analyse von Funktion F231(Universitätsbibliothek-Export)

Diese Funktion muss in einer extra App dargestellt werden. Die für diesen Export nötigen Informationen sind vollständig in der Datenbanken enthalten und müssen mittels einer direkten Datenbankabfrage ermittelt werden. Diese Daten werden durch eine Funktion in das richtige Format gebracht und in einem vom Benutzer vorher gewählten Dateipfad in einer .alg-Datei gespeichert. Dieses Sequenzdiagramm ist strukturell genauso wie das für den BibTex-Export, nur das auf eine andere Weise ausgegeben wird. Deswegen wird dies hier weggelassen.

2.22 Analyse von Funktion F300(Benutzerverwaltung)

Alle Informationen zu den Benutzern werden in einer Datenbank gespeichert. Änderungen an bestehenden Benutzern und das Hinzufügen neuer Benutzer wird durch ein Webformular von einem Benutzer mit ausreichenden Rechte eingeleitet. Mithilfe von Django werden die Informationen aus dem Formular gelesen und alle Änderungen in der Datenbank eingetragen. Das Sequenzdiagramm ist trivial, da nur einfach Frage/Antwort-Vorgänge stattfinden.

2.23 Analyse von Funktion F301(Rechtezuweisung für Rollen)

Die Rechte einer Rolle werden in einer eigenen Tabelle der Datenbank gespeichert. Ein Benutzer mit ausreichenden Rechten kann über ein Webformular Änderungen eingeben. Diese Änderungen werden mithilfe von Django aus dem Webformular ausgelesen und in der Tabelle der Datenbank gespeichert. Das Sequenzdiagramm ist trivial, da nur einfach Frage/Antwort-Vorgänge stattfinden.

2.24 Analyse von Funktion F302(Benutzer Rolle(n) zuweisen)

Die Rollen eines Benutzer werden in einer speziellen Tabelle in der Datenbank gespeichert. Ein Benutzer mit ausreichenden Rechten kann über ein Webformular Änderungen eingeben. Diese Änderungen werden mithilfe von Django aus dem Formular ausgelesen und in der entsprechenden Datenbank gespeichert. Das Sequenzdiagramm ist trivial, da nur einfach Frage/Antwort-Vorgänge stattfinden.

2.25 Analyse von Qualitätsmerkmal /Q10/ (Sicherheit der Anmeldung)

Die Verschlüsselung der Anmeldung wird von Django bereit gestellt und ist daher bereits eine Build-In-Funktion.

2.26 Analyse von Qualitätsmerkmal /Q11/ (SQL-Injections vermeiden)

SQL-Statements werden ausschließlich durch Übergabe von Parametern an Funktionen von Django ausgeführt. SQL-Injections sind in Django nicht möglich.

2.27 Analyse von Qualitätsmerkmal /Q20/ (Layoutstruktur)

Die Templates für Django müssen so angepasst werden, dass sie dem Corporate Design der TU Braunschweig entsprechen.

2.28 Analyse von Qualitätsmerkmal /Q21/ (Klare Struktur)

Die Stylesheets für die Templates von Django müssen auf Übersichtlichkeit ausgelegt werden.

2.29 Analyse von Qualitätsmerkmal /Q22/ (Suche)

Die in Django bereits implementierte Suche ist intuitiv und einfach.

2.30 Analyse von Qualitätsmerkmal /Q30/ (Zeichenketten)

Die Bearbeitung von Zeichenketten als Unicode wird bereits von der benutzten Datenbank SQ-Lite erfordert, welche nur Unicode-codierte Zeichenketten benutzt.

2.31 Analyse von Qualitätsmerkmal /Q31/ (Löschen von Dokumenten)

Löschen von Dokumenten ist mit Django möglich. Intern muss verhindert werden, dass andere Nutzer als Administratoren löschen dürfen.

3 Resultierende Softwarearchitektur

Dieser Abschnitt hat die Aufgabe, einen Überblick über die zu entwickelnden Komponenten und Subsysteme zu liefern.

3.1 Komponentenspezifikation

In diesem Abschnitt wird die aus der Analyse der Produktfunktionen (Kapitel 2) resultierende Komponentenstruktur zunächst überblickartig durch ein Komponentendiagramm beschrieben. Die Bezeichnungen und Anzahl der Komponenten muss natürlich konsistent sein mit der in Kapitel 2!

3.2 Schnittstellenspezifikation

Im Folgenden werden die einzelnen Schnittstellen der Komponenten aus der Komponentenspezifikation näher erläutert, d.h. die von Ihnen zur Verfügung gestellten Operationen werden dokumentiert. Die Tabelle ist dabei um so viele Zeilen zu erweitern, wie es Schnittstellen im Komponentendiagramm gibt. In der innen liegenden Aufteilung ist für jede Operation einer Schnittstelle eine Zeile einzufügen. Reine Set- und Get-Aufrufe brauchen nicht aufgeführt zu werden (sollten auch möglichst nicht komponentenübergreifend auftauchen).

Schnittstelle	Aufgabenbeschreibung
<Schnittstellen – ID>:	Operation | Beschreibung
<Bezeichnung>	<Signatur der Operation>|<Aufgabenbeschreibung der Operation>

3.3 Protokolle für die Benutzung der Komponenten

In diesem Abschnitt wird mit Hilfe von Protokoll-Statecharts die korrekte Verwendung der zu entwickelnden Komponenten dokumentiert. Dies ist insbesondere für diejenigen Komponenten notwendig, für die eine Wiederverwendung möglich erscheint oder sogar bereits geplant ist.

Begründen Sie für welche Komponenten eine Wiederverwendung sinnvoll erscheint und für welche nicht!

Fügen Sie so viele Statechartdiagramme ein, wie sie Komponenten gefunden haben.

4 Verteilungsentwurf

Sollte es sich bei dem Produkt um eine verteilte Anwendung handeln, so wird diese in diesem Abschnitt dokumentiert. Die Verteilung der Komponenten auf die beliebigen Knoten wird durch das folgende Verteilungsdiagramm beschrieben.

Eigenes Verteilungsdiagramm einsetzen!