

Bachelorarbeit

Ein Kursverwaltungssystem für programmierlastige Lehrveranstaltungen

Axel Lehmann

30.09.2013



Albert-Ludwigs-Universität Freiburg im Breisgau
Technische Fakultät
Institut für Informatik

Bearbeitungszeitraum

28.06.2013 – 30.09.2013

Gutachter

Prof. Dr. Hannah Bast

Erklärung

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

Freiburg, 30. September 2013

Axel Lehmann

Zusammenfassung

Daphne ist ein Kursverwaltungssystem für programmierlastige Kurse. In dieser Arbeit wird auf die Benutzern zur Verfügung gestellten Funktionen sowie die technischen Hintergründe eingegangen.

Daphne übernimmt die Verwaltung von Teilnehmern, Punkten und Einhaltung der Abgabefristen. Die Verwendung von Subversion als Versionkontrolle und des Continuous Integration System Jenkins zur Validierung von Abgaben ermöglicht direktes Feedback an die Teilnehmer.

Summary

Daphne is a Coursemanagement System which is focused on programming courses. This work describes provided features and the required technical background.

Daphen is used to manage participants, credits and deadlines. Using Subversion as version control system and Jenkins to validate submission and provide feedback to participants.

Inhaltsverzeichnis

1. Einleitung	1
1.1. Problemstellung	1
1.2. Zielsetzung	1
2. Komponenten des Systems	3
2.1. Supervisor	3
2.2. Webserver	4
2.2.1. Apache 2	4
2.2.2. Nginx	5
2.3. Django	7
2.4. Subversion	8
2.5. Continuous Integration System	9
2.6. Forum (phpBB3)	9
3. Funktionalität des Front-End	11
3.1. Registration	11
3.2. Vorlesungsperioden	13
3.2.1. Verwendung	13
3.2.2. Verwaltung von Vorlesungsperioden	14
3.3. Kurs	14
3.3.1. Kurs anlegen	15
3.3.2. Termine	15
3.3.3. Übungsblätter	15
3.3.4. Selbstregistration von Teilnehmern	16
3.3.5. Registration von Teilnehmern durch Kursverwalter	17
3.4. Informationen für	17
3.4.1. . . . Teilnehmer	17
3.4.2. . . . Tutoren	18
3.4.3. . . . Kursverwalter	18
3.5. Logging	18
3.6. Weitere Funktionalität	19
3.6.1. Portlet-Funktionalität	20
3.6.2. Rechteverwaltung	21
3.6.3. Terminabsprache	21

4. Funktionalität des Back-End	23
4.1. Cron	23
4.2. CAS	24
4.3. ØMQ Schnittstelle	24
4.4. Subversion-Daemon	25
4.5. Jenkins-Daemon	27
5. Evaluation	29
5.1. Verwandte arbeiten	29
5.1.1. CampusOnline & ILIAS	29
5.1.2. Moodle	29
5.2. Vergleich mit anderen Portalen	30
5.3. Benutzerfeedback	30
6. Ausblick	33
6.1. Allgemeine Arbeiten am System	33
6.2. Neue Funktionen	33
Danksagung	35
A. Anhang	37
Literaturverzeichnis	39

1. Einleitung

1.1. Problemstellung

An der Technischen Fakultät werden unterschiedliche Lehrveranstaltungen angeboten, welche theoretisch sowie praktische Abgaben erfordern. Dafür werden unterschiedliche System zum Upload von Dateien verwendet. Diese Systeme verfügen jedoch nicht über die Funktion Studenten direktes Feedback über den Zustand ihrer Abgabe zu geben. Auch fehlt die Möglichkeit die Entstehung eine Abgabe nachzuvollziehen oder eine frühere Version zu betrachten. Das Kursverwaltungssystem *Daphne*, welches in dieser Arbeit vorgestellt wird, versucht diese und weitere Funktionen, sowohl für Studenten, Tutoren, Assistenten und Dozenten von Lehrveranstaltungen, bei denen das Schreiben von Programmen einen nicht unwichtigen Teil der Arbeit einnimmt, bereitzustellen.

1.2. Zielsetzung

Im Allgemeinen war es bei der Umsetzung des Kursverwaltungsystems wichtig, dass keine neuen Zugangsdaten benötigt werden, wie es bei anderen Portalen der Fall ist. Einige der Systeme basieren auf dem Account der Technischen Fakultät¹, diese sind für Studierende anderer Fakultäten nur mit erhöhtem Aufwand nutzbar. Studenten sowie Mitarbeiter der Universität Freiburg erhalten jedoch automatisch eine Benutzerkennung des Rechenzentrums², wodurch die Wahl auf diese Benutzerkonten viel. Außerdem wurde die Möglichkeit geschaffen bei Bedarf Benutzerkonten anzulegen, welche unabhängig von der Authentifizierungsinfrastruktur des Rechenzentrums sind. Diese zusätzlichen Benutzerkonten können in allen an Daphne angeschlossenen Systemen verwendet werden.

Zeitnahes Feedback wird Teilnehmern von Kursen durch das Continuous Integration System *Jenkins*³ gegeben, welches die Abgaben automatisch kompiliert und anhand der zur Verfügung gestellten Testfälle verifiziert.

Ein Forum bietet Raum für Diskussionen und Fragen zu Aufgaben und allgemeinen Problemen mit Inhalten der Kurse.

¹<https://support.informatik.uni-freiburg.de/accounts.shtml>

²<https://myaccount.uni-freiburg.de/>

³<http://jenkins-ci.org>

Das gesamte System (mit Ausnahme von Jenkins) ist an das Corporate Design der Universität Freiburg⁴ angepasst. Durch Verwendung des Corporate Design sind einige Schaltflächen für Funktionen und deren Erscheinungsbild vorgegeben. So finden vor allem neue Benutzer eine vertraute Umgebung und grundlegende Funktionen an bekannten Stellen, was den Einstieg wie auch die Orientierung der Benutzer vereinfacht.

Durch die Spezialisierung des Systems auf die Abgabe von Quelltext und das vorhandene HIS/LSF-System⁵ mussten keine organisatorischen Aufgaben zur Planung von Räumen und Überschneidungen, wie sie in klassischen Kursverwaltungssystemen vorhanden sind, umgesetzt werden. Auch wurden keine interaktiven Lehrfunktionen, wie zum Beispiel Moodle-Quiz⁶ implementiert, da dafür auf SMILE⁷ und Produkte anderer Anbieter zurückgegriffen werden kann.

⁴<http://www.uni-freiburg.de/go/cd>

⁵<http://www.his.de/abt1/ab10>

⁶http://docs.moodle.org/23/en/Quiz_module

⁷<https://www.smile.informatik.uni-freiburg.de/startseite/funktionen.html>

2. Komponenten des Systems

Die in Daphne umgesetzten Funktionalitäten werden durch eine Vielzahl von Komponenten in unterschiedlichen Programmier- und Skriptsprachen bereitgestellt, in diesem Abschnitt werden diese näher erläutert.

Die Kommunikation zwischen einzelnen Teilen des Systems erfolgt durch JSON¹ sowie ØMQ². Diese Kombination erlaubt nach [Pië10] eine schnelle, sprachunabhängige sowie verständliche Kommunikation, da für Menschen lesbares JSON ohne nennenswerte Performanceeibußen gesendet werden kann.

Ein Hilfsmittel zur automatischen Verwaltung von Prozessen musste gefunden werden, welches im Fehlerfall möglichst schnell die benötigten Funktionalitäten wiederherstellen um Ausfallzeiten gering zu halten.

Das System verwaltet personenbezogene Daten der Teilnehmer, daher erfolgt sämtliche Kommunikation, welche das System verlässt, verschlüsselt. Dazu wird sowohl für Browser wie auch SVN-Clients SSL/TLS mit einem offiziellen Zertifikat des Rechenzentrums verwendet.

2.1. Supervisor

Supervisor³ startet bei Bedarf abgestürzte Prozesse neu und erlaubt es auch nicht Root-Benutzern Prozesse zu starten oder anzuhalten, sofern die entsprechenden Rechte eingeräumt wurden. Die bereitgestellte XML-RPC-Schnittstelle erlaubt es Supervisor durch Skripte zu verwalten und so dynamische Änderungen vorzunehmen.

Alle Prozesse einer Daphne-Instanz, welche Instanz spezifisch sind, werden durch einen Instanz-Supervisor verwaltet, die Webserverprozesse (Apache und Nginx) werden von einem übergeordneten Supervisor verwaltet, welcher auch die Kontrolle über die Supervisor Instanzen hat. Dadurch ist gewährleistet, dass Abstürze von Prozessen ohne manuelle Eingriffe behoben und entsprechende Ausfallszeiten minimiert werden.

1 [program : main-daphne]

¹JavaScript Object Notation

²<http://zeromq.org/> - erweiterte Socket-Bibliothek

³<http://supervisord.org/>

```
2 command = /local/data/daphne/instances/productive/main/venv/  
  bin/python /local/data/daphne/instances/productive/main/  
  daphne/manage.py runfcgi socket=/local/data/daphne/  
  instances/productive/main/sockets/daphne.sock daemonize=  
  false  
3 environment=PATH="/local/data/daphne/instances/productive/  
  main/venv/bin "  
4 user = www-data  
5 autorestart = true  
6  
7 [program:main-daphne-api]  
8 command = /local/data/daphne/instances/productive/main/venv/  
  bin/python /local/data/daphne/instances/productive/main/  
  daphne/daemon.py -d  
9 environment=PATH="/local/data/daphne/instances/productive/  
  main/venv/bin "  
10 user = daphne  
11 autorestart = true
```

Listing 2.1: Supervisor-Eintrag der Daphne Komponenten

Listing 2.1 zeigt die Supervisor Konfiguration der Daphne-Prozesse einer Daphne-Instanz. Diese Prozesse werden als unterschiedliche Benutzer ausgeführt, da das Web-Front-End, welches als *www-data* ausgeführt wird, keine Änderungen an den Dateien des Benutzers *daphne* vornehmen soll. Durch die Verwendung verschiedener Benutzerknoten wird so im Vorfeld ein möglicher Angriffsvektor vermieden.

2.2. Webserver

Auf alle Komponenten, mit Ausnahme der Daemon, kann über das Internet zugegriffen werden. Für die Bereitstellung werden zwei Webserver-Dienste verwendet, welche unterschiedliche Aufgaben wahrnehmen. Grundlegende SVN-Funktionalitäten werden durch einen Apache2-Webserver mit entsprechenden Modulen bereitgestellt. Der Nginx-Webserver liefert normale Webseiten aus, was sich vor allem im Bereich der statischen Dateien wie Bilder oder CSS-Dateien positiv auf die Serverlast wie auch auf Übertragungszeiten auswirkt.

2.2.1. Apache 2

Der Apache-Webserver wird auf Port 80 (http) betrieben und antwortet auf Anfragen mit einem Redirect auf Port 443. Die Bereitstellung der SVN-Repositorien erfolgt über 127.0.0.1:8004. Somit müssen alle Anfragen direkt über den Host erfolgen.

Eingehende HTTPS-Verbindungen werden durch den Nginx behandelt und somit terminiert. Der Request-Header enthält weiterhin die Protokollkennung *https* welche zu einem ungültigen Request führt. Daher wird die Protokollkennung zu *http* umgeschrieben.

```
23 <VirtualHost 127.0.0.1:8004 >
24     ServerName daphne.informatik.uni-freiburg.de
25     ServerAlias daphne
26     ServerAdmin webmaster@localhost
27     ServerSignature off
28     UseCanonicalName on
29     RequestHeader edit destination ^https: http: early
```

Listing 2.2: Definition SVN Virtual-Host (Apache)

Die meisten Konfigurationen binden PHP und Python per Apache-Modul ein, somit werden PHP/Python wird dann direkt im Prozess des Webservers mit dessen Rechten ausgeführt. Auch kann so ein fehlerhaftes Skript, welches sich nicht per Timeout beenden lässt, den Webserver blockieren. Daher wurden alle Prozesse, mit Ausnahme von SVN, ausgelagert. Die Trennung ermöglicht es blockierte beziehungsweise abgestürzte Subprozesse neu zu starten, ohne dass andere Prozesse und durch diese bereitgestellte Dienste beeinträchtigt werden. Das Einbinden der Prozesse durch die FastCGI-Schnittstelle bereitete große Fehler wodurch alternative Webserver betrachtet wurden.

```
35     <Location "/svn">
36         AuthType Basic
37         AuthName "Geben Sie Ihren Daphne-Account ein:"
38         AuthBasicProvider external
39         AuthExternal "main-daphne-svn"
40         Require valid-user
41
42         DAV svn
43         SVNParentPath /local/data/daphne/instances/productive/
44             main/svn
45         AuthzSVNAccessFile /local/data/daphne/instances/
46             productive/main/svn/svn_access.txt
47         Satisfy Any
48         Order allow,deny
49     </Location>
```

Listing 2.3: Definition SVN-Location (Apache)

2.2.2. Nginx

Nginx wird auf Port 443 (https) betrieben und nimmt sämtliche Anfragen entgegen. Abhängig von der URL wird die Anfrage an entsprechende Prozesse weitergegeben.

Außerdem setzt der Nginx den Http-Header *Strict-Transport-Security* RFC 6707⁴ gesetzt, welcher Webbrowser anweist nur über verschlüsselte Verbindungen zu kommunizieren. Unsichere Cipher werden in der Nginx Konfiguration ausgeschlossen und die Verwendung von Perfect Forward Secrecy wird vom Server vorgeschlagen.

```

3     ssl on;
4     ssl_certificate /etc/nginx/cert-300622278-chain.pem;
5     ssl_certificate_key /etc/nginx/cert.key;
6     ssl_ciphers ECDHE-RSA-RC4-SHA:ECDHE-RSA-AES128-SHA256:
        HIGH:!aNULL:!MD5;
7     ssl_prefer_server_ciphers on;
8     ssl_session_cache shared:SSL:10m;
9     ssl_session_timeout 10m;
10    server_name daphne.informatik.uni-freiburg.de;
11    add_header Strict-Transport-Security max-age=600;

```

Listing 2.4: SSL Einstellungen (Nginx)

Eingehende Anfragen werden vom Nginx sowohl durch Proxy- als auch FastCGI-Verbindungen an zuständige Prozesse delegiert. Für die korrekte Zuweisung des Ziel-Prozesses genügt, außer für SVN welches zusätzlich den User-Agent benötigt, die URL.

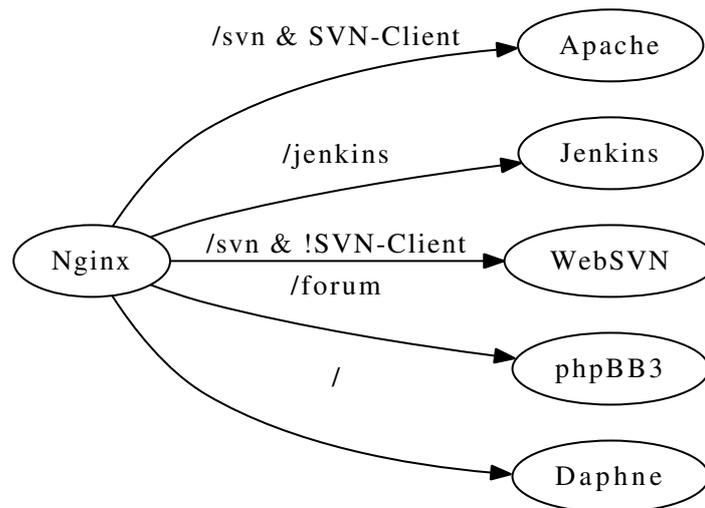


Abbildung 2.1.: Verteilung der Anfragen in Abhängigkeit des Pfades

```

97     if ($http_user_agent !~* ^SVN) {
98         fastcgi_pass unix:/local/data/daphne/instances/
        productive/main/sockets/websvn.sock;
99     }

```

⁴<https://tools.ietf.org/html/rfc6707>

```
100     if ($http_user_agent ~* ^SVN) {
101         proxy_pass http://127.0.0.1:8004;
102     }
```

Listing 2.5: Definition SVN-Location (Nginx)

2.3. Django

Django⁵ ist ein Open-Source Web-Framework zur Bereitstellung und Entwicklung von Internetdiensten. Dabei übernimmt das Framework das Routing von Anfragen, die Anbindung von Datenbanken sowie grundlegende Sicherheitsmaßnahmen wie die Verhinderung von Cross-Site-Request-Forgery. Die Community um Django zeichnet sich durch schnelle Reaktionen bei gemeldeten Sicherheitslücken⁶ aus.

Wie schon [Zho10] beschreibt, sind nur wenige Änderungen nötig um die bereitgestellten Grundklassen an eigene Bedürfnisse anzupassen.

```
1 class ExerciseListView(
2     ExerciseListBreadcrumbMixin,
3     ListView):
4     model = Exercise
5
6     def get_queryset(self):
7         return super(ExerciseListView, self) \
8             .get_queryset().select_related().filter(course=
                self.course)
```

Listing 2.6: Modell der Übungsliste

Diese Klasse erzeugt eine Auflistung aller Übungen eines bestimmten Kurses, für eine Liste aller Übungen, ohne Einschränkung auf einen bestimmten Kurs, genügt die Angabe des zu verwendenden Modells. Außerdem bietet Django Hilfsmittel zur Internationalisierung und Lokalisierung⁷ an.

Außerdem bietet Django eingebaute Möglichkeiten zur Ausführung von Unit-Test, diese können sowohl als eigene Funktionen als auch wie in Python vorgesehen im Doc-String definiert werden. Durch verwenden der Doc-String-Tests lässt sich ein auseinanderdriften der Beispiele der Dokumentation und des Quelltextes verhindern, da bei einem kompletten Test-Run auch die Dokumentationsbeispiele getestet werden.

⁵<https://www.djangoproject.com/>

⁶<https://www.djangoproject.com/weblog/2013/sep/15/security/>

⁷<https://docs.djangoproject.com/en/1.5/topics/i18n/>

```

1  def __eq__(self, other):
2      """
3      Checks equality. Equality is given iff url and name
4          is identical.
5
6          :param other: 'Breadcrumb' to compare this to.
7
8          >>> crumb1 = Breadcrumb('name1', 'url1')
9          >>> crumb2 = Breadcrumb('name2', 'url1')
10         >>> crumb3 = Breadcrumb('name1', 'url2')
11         >>> crumb1.__eq__(crumb1)
12             True
13         >>> crumb1.__eq__(crumb2)
14             False
15         >>> crumb1.__eq__(crumb3)
16             False
17         """
18         return self.url == other.url and self.name == other.name

```

Listing 2.7: Doc-String-Test des Breadcrumb Identitätsvergleich

Das Rechtesystem von Django kennt Standardmäßig keine Rechte auf Objekt-Ebene, daher wurde das Rechtesystem nach Sichtung möglicher Erweiterungen um eine auf *django-guardian*⁸ basierende Eigenentwicklung erweitert. Diese bietet ähnliche Funktionen wie *django-guardian* sowie die Möglichkeit der Vergabe von Rechten auf übergeordneten Objekten (Kurse) welche für alle untergeordneten Objekte gelten, ohne dass diese Rechte erneut eingetragen werden.

2.4. Subversion

Subversion⁹ ist ein zentralistisches Versionskontrollsystem. Im Gegensatz zu *git* gibt es einen zentralen Server welcher von allen Benutzern verwendet wird. Da alle Änderungen an diesen Server übermittelt werden müssen, ist dieser in der Lage die Zugangsrechte zu überprüfen und zu erzwingen. Auch ermöglicht dies die Einschränkung der Leserechte, was bei *git* nicht möglich ist, da dort immer das komplette Repository geklont werden muss.

Als Front-End kommt WebSVN¹⁰ da diese sich mit wenigen Änderungen mit dem

⁸<https://github.com/lukaszbdjango-guardian>

⁹<https://subversion.apache.org/>

¹⁰<http://www.websvn.info/>

CAS verbinden lässt und die Zugriffsbeschränkungen durch das Apache Subversion-Modul honoriert.

2.5. Continuous Integration System

Als Continuous Integration System kommt *Jenkins*¹¹, ein *Hudson*-Fork, zum Einsatz. Jenkins übernimmt die Validierung der Abgaben, indem vorgegebene Schritte ausgeführt werden. Durch die Verwendung des VirtualBox Plugins¹² wird für jeden Kurs eine eigene Umgebung bereitgestellt, welche eigene Softwarekonfigurationen wie C++-Compiler oder Java Versionen beinhalten. Auch ist so eine Manipulation von Abgaben anderer Kurse und Teilnehmern ausgeschlossen.

Die Verwaltung der Buildjobs wird von Daphne übernommen indem über den zugehörigen Daemon Änderungen als Jenkins-Administrator über die vorhandene REST-API eingepflegt werden.

2.6. Forum (phpBB3)

Als Forensoftware wird phpBB¹³ in der derzeit aktuellen Version 3.0.11 verwendet, diese ist in PHP¹⁴ geschrieben und steht als Open-Source-Projekt auf GitHub¹⁵ zur Verfügung. Die Änderungen an der Forensoftware beschränken sich auf ein eigenes Authentifizierungsmodul auf Basis einer Arbeit von Anelis¹⁶ sowie Anpassungen des Layouts an das Corporate Design. Somit lassen sich Updates der Software ohne große Probleme anwenden, was besonders in Bezug auf sicherheitsrelevante Updates sinnvoll ist. Außerdem entfällt durch das Plugin die Notwendigkeit sich erneut für das Forum anzumelden, da die Authentifizierung über den zentralen Authentifizierungsservice erfolgt, welcher die zuletzt für den Benutzer hinterlegte E-Mail-Adresse an das Forum übermittelt bekommt und in Abschnitt 4.2 beschrieben wurde. Durch die Verwendung dieses Dienstes und die damit verbundene Abschaltung der Foren internen Registrierungsmöglichkeit ist eine wirksame Abwehr von Spambots gegeben, da Benutzerkennungen nicht frei angelegt werden können.

¹¹<http://jenkins-ci.org>

¹²<https://wiki.jenkins-ci.org/display/JENKINS/VirtualBox+Plugin>

¹³<https://www.phpbb.com>

¹⁴<http://php.net>

¹⁵<https://github.com/phpbb/phpbb3>

¹⁶<https://github.com/anelis/phpbb-cas>

3. Funktionalität des Front-End

Die von Daphne bereitgestellten Funktionen sind vielfältig und umfangreich. Die nachfolgende Reihenfolge der Funktionen erlaubt eine plastischere Darstellung der Möglichkeiten, da relevanten Teilfunktionalitäten bereits zuvor erläutert wurden.

3.1. Registration

Zur Identifikation der Benutzer finden in Daphne die Benutzerkennungen des Universitäts Rechenzentrums Verwendung. Die Anmeldemaske ist, wie in Abbildung 3.1 gezeigt, in zwei Bereiche aufgeteilt. Im linken Bereich befindet sich das Eingabeformular, im rechten Bereich wird die Reichweite der Anmeldung beschrieben.

The screenshot shows the login interface of the Daphne system. At the top, there are language options: Deutsch, British English, español, 日本語. The main header displays the 'Daphne' logo and 'Albert-Ludwigs-Universität Freiburg'. A navigation menu on the left includes links for 'Anmelden', 'Daphne', 'Forum', 'Jenkins', 'SVN', 'Status', 'Bugtracker', and 'Hilfe'. The central area contains a login form with 'Anmelden' and 'Anmelden' buttons, and a text box on the right explaining the authentication process.

Abbildung 3.1.: Anmelde Formular des Kursverwaltungssystems.

Fehlerhafte Anmeldeversuche werden dem Benutzer durch einen lokalisierten Hinweistext zur Beachtung der Groß- und Kleinschreibung und erneute Vorlage des Anmeldeformulars beantwortet. Wie in Abbildung 3.2 zu sehen wird der zuvor eingetragene Benutzername bereits in das Formular eingetragen, so dass in der Wartezeit nur das zu dem Benutzernamen korrespondierende Kennwort eingegeben werden muss.

Als Maßnahme gegen mögliche Brute-Force-Angriffe gegen Benutzer und deren Kennwörter verweigert das System nach einer Falschanmeldung für eine konfigurierbare



Abbildung 3.2.: Anmelde Formular des Kursverwaltungssystems nach fehlerhaftem Anmeldeversuch. Hinweis auf die Beachtung von Groß- und Kleinschreibung von Benutzername und Kennwort.

Anzahl an Sekunden eine erneute Anmeldung von der IP-Adresse, welche den Fehlerhaften versuch ausgelöst hat. Bei der in Abbildung 3.3 abgebildeten Fehlermeldung wird zusätzlich HTTP-Statuscode *403 Forbidden* gesendet, welcher in RFC 2616¹ definiert ist.



Abbildung 3.3.: Allgemeine Fehlerseite, ausgelöst durch zu schnelles übermitteln der Anmeldedaten nach einem gescheiterten Anmeldeversuch.

Nach der Identifikation durch Eingabe von Benutzername und Kennwort werden grundlegende Daten mit dem LDAP²-Server des Rechenzentrums abgeglichen und

¹<https://tools.ietf.org/html/rfc2616>

²Lightweight Directory Access Protocol

dem angemeldeten Benutzer, wie in Abbildung 3.4 gezeigt, präsentiert. Die Korrektur fehlerhafter Daten muss daher durch die vom Rechenzentrum vorgesehenen Korrekturverfahren erfolgen.



Abbildung 3.4.: Anzeige der persönlichen Daten nach erfolgreichem Login.

Im Bedarfsfall kann der Benutzerstamm um lokale, von Daphne direkt verwaltete, Kennungen erweitert werden. Beim lokalen Anlegen einer Kennung werden die erweiterten Daten, die nicht beim LDAP-Server des Rechenzentrums nachgefragt werden, in einem Formular abgefragt.

Dieses Formular erlaubt außerdem die Manipulation vorhandener Benutzerkennungen. Die Einstellungen zur Kontenaktivität und des Administratorenstatus sind dominant, die Namensdaten hingegen werden bei der nächsten Anmeldung einer vom Rechenzentrum verwalteten Benutzerkennung erneut vom LDAP-Server angefragt.

3.2. Vorlesungsperioden

Durch die Zuordnung von Kursen zu Semestern ist eine chronologische Sortierung möglich, welche das Auffinden von Kursen vereinfacht. Viele Lehrstühle nutzen diese Organisationsstruktur auch innerhalb des eigenen Webauftrittes. Daphne organisiert Vorlesungsperioden analog zu der in Abbildung 3.5 gezeigten Anordnung.

3.2.1. Verwendung

Vorlesungsperioden, Wintersemester und Sommersemester, organisieren Kurse in Gruppen und regulieren die Möglichkeit zum Anmelden an einen Kurs. Nach dem

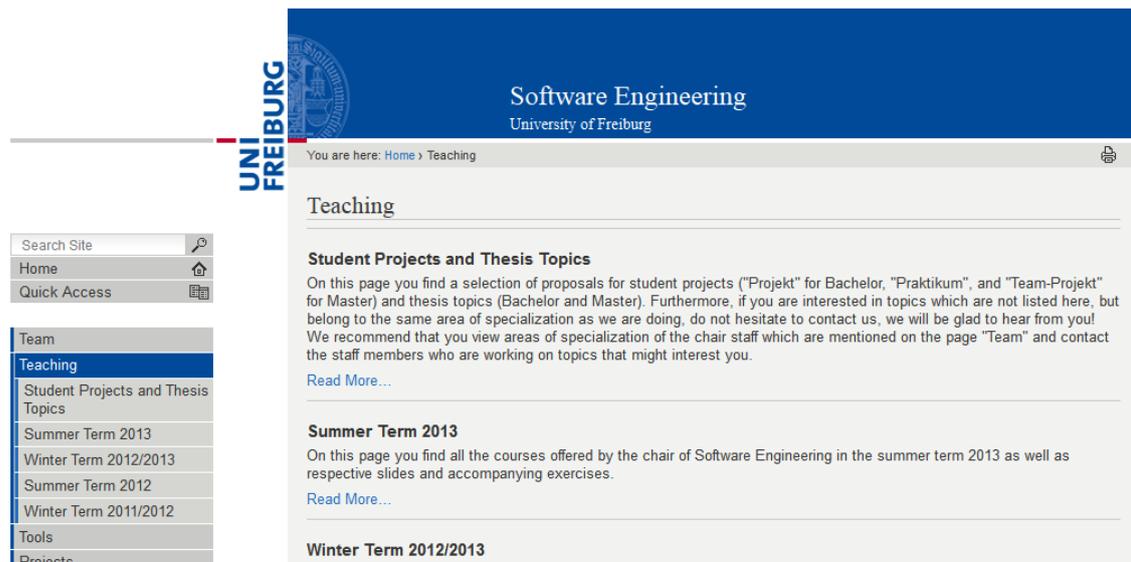


Abbildung 3.5.: Organisation der Informationen zu Lehrveranstaltungen des Lehrstuhles für Softwaretechnik.

Ablauf einer Periode ist es nicht mehr möglich sich für einen Kurs anzumelden. Des Weiteren werden die Zugriffsrechte der assoziierten Versionierungssystem sowohl für Teilnehmer als auch für Tutoren in einen schreibgeschützten Zustand versetzt um Manipulationen zu verhindern.

3.2.2. Verwaltung von Vorlesungsperioden

Vorlesungsperioden sind Zeitspannen zwischen zwei fixierten Zeitpunkten, welche als kursunabhängige Termine im System hinterlegt. Somit kann das Ende einer Periode als Anfang der Nächsten verwendet werden, was bei Anpassungen und Verschiebungen den Arbeitsaufwand verringert.

Schneller Zugriff auf in Daphne hinterlegte Kurse erfolgt über die Auswahl des relevanten Semesters, welche wie in Abbildung 3.6 gezeigt, automatisch in das Menü des Systems aufgenommen werden.

3.3. Kurs

Ein Kurs umfasst, unter der Annahme, dass es nur Abgabetermine für alle Teilnehmer gibt, eine Lehrveranstaltung. Sollten unterschiedliche Abgabefristen für verschiedene Teilnehmergruppen benötigt werden, so werden mehrere Kurse für diese Lehrveranstaltung benötigt. Der Kurs verknüpft die zusätzlich angeschlossenen Systeme und bietet Möglichkeiten zur Verwaltung von Teilnehmern, Tutoren sowie Links und Übungsblättern.



Abbildung 3.6.: Im Menü und System eingetragene Vorlesungsperiode.

3.3.1. Kurs anlegen

Wie in Abbildung 3.7 gezeigt sind nur wenige Informationen für einen Kurs verpflichtend. Diese umfassen einen möglichst kurzen *URL-Namen* welcher Teil der URL wird und den Haupteinstiegspunkt für alle mit dem Kurs assoziierten Personen darstellt. Außerdem wird ein lesbarer Name für den Kurs benötigt, welcher dem Benutzer an verschiedenen Stellen angezeigt wird.

Für den Kurs kann eine Python-Funktion hinterlegt werden. Diese kann aus den für Übungen eingetragenen Punkten eine Endnote errechnen.

Der Kurs *Lehrveranstaltung mit Programmieren* kann mit dem URL-Namen *kurs1* angelegt werden, da die Zuordnung zu einem Semester durch die Verknüpfung zur Vorlesungsperiode, unter welcher der Kurs erzeugt wird, hergestellt wird. Durch diese Verknüpfung kann der URL-Name in einem weiteren Semester wieder verwendet werden.

3.3.2. Termine

Als Termine werden die mit Übungsblättern verknüpften Zeiten der Ausgabe, Abgabe sowie Korrekturfristen abgebildet.

Durch optionalen Felder kann eine Beschreibung hinzugefügt werden sowie weitere mit dem Kurs verbundene Termine an Teilnehmer weitergegeben werden. Diese Liste ist per RSS-Abonnier bar.

3.3.3. Übungsblätter

Übungsblätter sind der Kern vieler Lehrveranstaltungen, da diese Inhalte und Vorgehensweisen vertiefen und Teilnehmer an Problemstellungen heranzuführen.

The screenshot shows the 'Add' form for creating a course in the Daphne system. The form is part of a web interface for the University of Freiburg. The header includes the university logo and the text 'Daphne Albert-Ludwigs-Universität Freiburg'. The breadcrumb trail indicates the user is in 'Daphne > Wintersemester 2013/2014 > Add'. The form fields are as follows:

- Url name:** A text input field containing 'kurs1'.
- Long name:** A text input field containing 'ung mit Programmieren'.
- Description:** A large text area for entering the course description.
- Qis url:** A text input field for the QIS URL.
- Mark function:** A large text area for entering the mark function.
- Show marks:** A checkbox that is currently unchecked.
- Use default tutorage:** A checkbox that is currently unchecked.
- Daten absenden:** A button to submit the form.

On the left side of the interface, there is a navigation menu with the following items: Abmelden [admin], Daphne, Forum, Jenkins, SVN, Wintersemester 2013/2014, Logs, Benutzer, Services, Status, Bugtracker, and Hilfe.

Abbildung 3.7.: Formular zum Anlegen eines Kurses. Es können optionale Daten hinterlegt werden.

Ein Übungsblatt benötigt zwingend Termine für den Beginn und das Ende des Bearbeitungszeitraum, eine URL zu den Aufgaben sowie einen in URLs zulässigen Kurznamen, welcher vom Versionsverwaltungssystem zur Festlegung der Zugriffsrechte verwendet wird.

Der persönliche Abgabepfad bildet sich als:

Domain/Kurzname des Kurs/Benutzerkennung/Kurzname des Übungsblatt

3.3.4. Selbstregistration von Teilnehmern

Benutzer registrieren sich innerhalb einer Vorlesungsperiode selbst für einen Kurs. Bei der Registration werden verschiedene Daten abgefragt, welche nicht aus den Stammdaten bestimmbar sind.

Durch die Eingabe einer aktuellen E-Mail-Adresse soll sichergestellt werden, dass Tutoren diese zur Kommunikationsaufnahme verwenden können. Die Informationen zu Studiengang, geplantem Abschluss und Semester werden nicht vom Rechenzentrum

The screenshot shows the Daphne interface for the University of Freiburg. At the top, there are language options: Deutsch, British English, español, 日本語. The main header displays 'Daphne' and 'Albert-Ludwigs-Universität Freiburg'. A breadcrumb trail indicates the current location: 'Sie sind hier: Daphne > Wintersemester 2013/2014 > Lehrveranstaltung mit Programmieren'. A green notification box at the top of the main content area states: 'Lehrveranstaltung mit Programmieren wurde erfolgreich angelegt.' Below this, the course title 'Lehrveranstaltung mit Programmieren' is displayed. The interface lists several categories with zero items: 'Links (0)', 'Übungen (0)', 'Termine (0)', and 'tutors (0)'. A 'Data' section includes a 'Mail an alle' link. A table header is visible with columns: 'Benutzername', 'tutor', 'Points', and 'Mark'. The footer contains 'Unicorn | Cookies löschen' and 'Über Daphne | © 2010 - 2013 Universität Freiburg, IIF, AD'. On the left side, there is a navigation menu with options like 'Abmelden [admin]', 'Daphne', 'Forum', 'Jenkins', 'SVN', and a course selection menu for 'Wintersemester 2013/2014' where 'Lehrveranstaltung mit Programmieren' is selected. Other menu items include 'Logs', 'Benutzer', 'Services', and 'Status'.

Abbildung 3.8.: Angelegter Kurs, welcher automatisch im Menü eingetragen wurde.

zur Verfügung gestellt. Diese sind aber für die Einschätzung der Abgabequalitäten und dem bereits vorhandenen Wissen relevant.

3.3.5. Registration von Teilnehmern durch Kursverwalter

Ein Kurs kann auch durch Teilnehmer ergänzt werden, welche sich nicht selbst registrieren. Hierzu muss der Teilnehmer, welcher hinzugefügt werden soll, bereits eine erfolgreiche Anmeldung am Kursverwaltungssystem vorweisen können, da diese das benötigte Benutzerobjekt anlegt.

3.4. Informationen für . . .

Auf der Kursseite werden relevante Informationen abhängig der vorhanden Rechte angezeigt. Je nach Zugehörigkeit verändert sich die Form der Darstellung wie auch die Reihenfolge.

3.4.1. . . . Teilnehmer

Teilnehmern erhalten Informationen über die angeschlossenen Systeme sowie Übungsblätter und erhaltene Punkte. Das aktuellste Übungsblatt befindet sich immer an der obersten Position so können relevante Daten wie Abgabezeitpunkt und erhaltene Punkte schnell eingesehen werden. Wie in Abbildung 3.9 zu sehen ist, steht die Zeichenfolge - - - für ein noch nicht bepunktetes Übungsblatt.

The screenshot shows the Daphne interface for a participant named Max Mustermann. The page is titled 'teilnehmer1 (Max Mustermann)'. It displays personal information such as Vorname (Max), Nachname (Mustermann), and Studiengang (Beispiel). It also shows course details like Semester (5) and Abschluss (Bachelor). A table at the bottom shows the current exercise sheet 'Übungsblatt 01' with a score of 21 out of 28, and a total score of 20.0 out of 20.0. The page includes a navigation menu on the left, a language selector at the top, and a footer with 'Unicorn | Cookies löschen' and 'Über Daphne | © 2010 - 2013 Universität Freiburg, IIF, AD'.

Abbildung 3.9.: Kursinformationen für einen Teilnehmer

3.4.2. ... Tutoren

Tutoren erhalten die Informationen tabellarisch aufgearbeitet. Dabei werden alle zugeordneten Studenten sortiert nach Benutzernamen aufgelistet. Das aktuellste Übungsblatt wird stets Links angezeigt. Eine Übersicht über die Übungsblätter findet sich wie in Abbildung 3.10 zu erkennen ist, unterhalb der Tabelle. Die Zeichenfolge - - - steht auch hier für noch nicht eingetragene Punkte, welche mit dem der Zeichenfolge nachgeschalteten Formular gesetzt werden können.

3.4.3. ... Kursverwalter

Für Kursverwalter ist die Ansicht ähnlich zu Abbildung 3.8. Die Tabelle im unteren Bereich des Bildes wird hierbei um Informationen zu sämtlichen Teilnehmern ergänzt und ist im Aufbau identisch zu der Tabelle aus Abbildung 3.10.

3.5. Logging

Jede Änderung an Objekten wird mit Uhrzeit, vorherigem Zustand sowie dem Auslöser der Änderung gespeichert. So kann im Falle von Abweichungen und unterschiedlicher Aussagen eine verlässliche Bewertung der Fakten erfolgen. Auch kann ein älterer Zustand wiederhergestellt werden.

Deutsch British English español 日本語

Daphne
Albert-Ludwigs-Universität Freiburg

Sie sind hier: Daphne > Wintersemester 2013/2014 > Lehrveranstaltung mit Programmieren

tutor1 (Paul Streng)

Vorname: Paul Studiengang: Beispiel
 Nachname: Streng Abschluss: Master
 Kurs-E-Mail: tutor1@provider.de Semester: 3

Termine Übungen

tutor1

Mail an alle

Benutzername	Points	Übungsblatt 01
teilnehmer1	0 / 20	--- / 20

Übungsblatt 01
<http://ad.informatik.uni-freiburg.de/>

Unicorn | Cookies löschen Über Daphne | © 2010 - 2013 Universität Freiburg, IIF, AD

Abbildung 3.10.: Kursinformationen für einen Tutor

```
1 [{"pk": 1, "model": "course.course", "fields": {"show_marks": false, "description": "", "mark_function": "", "period": 1, "long_name": "Lehrveranstaltung mit Programmieren", "qis_url": "", "use_default_tutorage": false, "root": 1, "slug": "kurs1"}}]
```

Listing 3.1: Ausschnitt aus dem Logeintrag beim Anlegen der Kurse

3.6. Weitere Funktionalität

Das System bietet außerdem einige Funktionen, welche bisher nur durch Systemadministratoren genutzt werden können. Sobald die entsprechenden Benutzeroberflächen vervollständigt sind, werden diese den entsprechenden Benutzergruppen zur Verfügung gestellt. Ausnahme hierzu bietet die Portlet-Funktionalität welche nur Systemadministratoren zur Verfügung stehen wird.

3.6.1. Portlet-Funktionalität

Als Content-Management-System der Universität Freiburg findet Plone³ Verwendung. Dieses umfangreiche CMS ermöglicht es Text- sowie HTML-Widgets, welche als Portlets bezeichnet sind, abhängig von der Position der aktuellen Seite im Portal anzuzeigen. Die in Abbildung 3.11 umrandeten Bereiche zeigen eine Auswahl der Möglichkeiten sowie die in Plone möglichen Positionen.

The screenshot shows the homepage of the University of Freiburg (UNI FREIBURG). The page is structured with several portlets and navigation elements:

- Header:** Language selection (English, Deutsch) and the university name "Albert-Ludwigs-Universität Freiburg".
- Navigation:** "Sie sind hier: Startseite" and a search bar.
- Left Sidebar:** A menu with options: "Website durchsuchen", "Startseite", "Schnellzugriff", "Informationen für...", "Universität", "Studium", "Forschung", and "Weiterbildung". Below this are logos for "EXZELLENZINITIATIVE UNIVERSITÄT FREIBURG", "Ausgezeichnet! Wettbewerb exzellente Lehre", "TOTAL E-QUALITY", "LE RU", "U5", and "eucor".
- Main Content Area:**
 - Im Gespräch:** A section titled "Flexibel, freundlich, frisch" with a photo of a lecture hall and text about renovation work.
 - Nachrichten:** A section with three news items: "Nikolaus, der Vielseitige", "Vorsicht, Rutschgefahr!", and "Strasburger-Preis für Adrian Alder".
 - Veranstaltungen:** A section for upcoming events.
- Right Sidebar:**
 - Informationen für...:** A list of links for "Studierende", "Wissenschaftler/-innen", "Mitarbeiter/-innen", "Presse / Newsroom", and "Alumni".
 - Listen und Verzeichnisse:** A list of links for "Vorlesungsverzeichnis", "Podcasts", "Forschungsdatenbank", "ZUV - Service A-Z", "Stellenbörse", "Wahlplattform", "Alle Einrichtungen", and "Index A-Z".
 - Kontakt & Anreise:** Contact information including address, postal code, phone number, and links for "Kontakt", "Personensuche", "Öffnungszeiten", "Anreise", and "Lagepläne".
- Footer:** "Übersicht | Index A-Z | Feedback | Kontakt | Impressum" and "Copyright © 2013 Universität Freiburg".

Abbildung 3.11.: Startseite der Universität Freiburg. Zusätzliche Rahmen markieren Portlets im Seitenrandbereich.

In der aktuellen Umsetzung erlaubt die Portlet-Funktionalität ebenfalls das Einfügen von HTML-Code, da dies bei falscher Verwendung zu Sicherheitsrisiken führen kann, ist die Verwendung nur durch Systemadministratoren vorgesehen. Außerdem sind Portlets auf den linken Seitenrand, unterhalb des Menüs beschränkt.

³<https://plone.org>

3.6.2. Rechteverwaltung

Eine umfangreiche Rechteverwaltung ermöglicht es Rechte auf einzelne Objekte, einen Kurs oder Systemweit zu definieren, einige Rechte ergeben sich implizit aus der Zuordnung von Tutoren zu Teilnehmern. So ist es möglich einem Assistenten umfangreiche Rechte über den Kurs zu erteilen, diesem aber das Recht zum Verändern der Tutorenzuordnung zu verweigern, indem das Recht `assign_tutor` verweigert wird.

3.6.3. Terminabsprache

Zur einfachen Absprache beinhaltet Daphne ein an *Doodle*⁴ angelehntes Abstimmungsverfahren, welches in Abbildung 3.12 abgebildet ist. Zum aktuellen Zeitpunkt fehlt jedoch die Benutzeroberfläche zum Anlegen der Umfragen, die Teilnahme hingegen ist uneingeschränkt möglich.

The screenshot shows the Daphne interface for a pre-course poll. The header includes the UNI FREIBURG logo and the text 'Daphne Albert-Ludwigs-Universität Freiburg'. The main content area is titled 'Vorkursplanung' and shows a poll for '24.10.2013' regarding the 'Uhrzeit des Vorkurses'. The poll results are displayed in a table with columns for time slots and rows for users. The table shows that 'kursleiter1' has voted 'Yes' for all slots, while other participants have voted 'Yes' for various slots, with some 'No' votes and 'WSM' (Workshop) entries. The footer includes 'Unicorn | Cookies löschen' and 'Über Daphne | © 2010 - 2013 Universität Freiburg, IIF, AD'.

Benutzer	24.10.2013						
	15:30:00	15:45:00	16:00:00	16:15:00	16:30:00	16:45:00	17:00:00
kursleiter1	Yes	Yes	Yes	Yes	Yes	Yes	Yes
teilnehmer1	Yes	Yes	Yes	Yes	WSM	WSM	WSM
teilnehmer2	WSM	Yes	WSM	Yes	WSM	No	No
teilnehmer3	Yes	Yes	Yes	Yes	Yes	Yes	No
teilnehmer4	No	No	No	No	No	No	No
tutor1	No	No	No	No	Yes	Yes	Yes
tutor2	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	5	5	5	5	6	5	4

Abbildung 3.12.: Abstimmung über die Uhrzeit des Vorkurses.

⁴<http://doodle.com>

4. Funktionalität des Back-End

Im vorherigen Kapitel wurden die Funktionen des Front-Ends beschrieben, welche durch direkte Interaktionen der Benutzer mit dem System ausgelöst werden. Die in diesem Kapitel beschriebenen Funktionen werden entweder Zeitgesteuert ausgelöst oder von verknüpften Diensten genutzt.

4.1. Cron

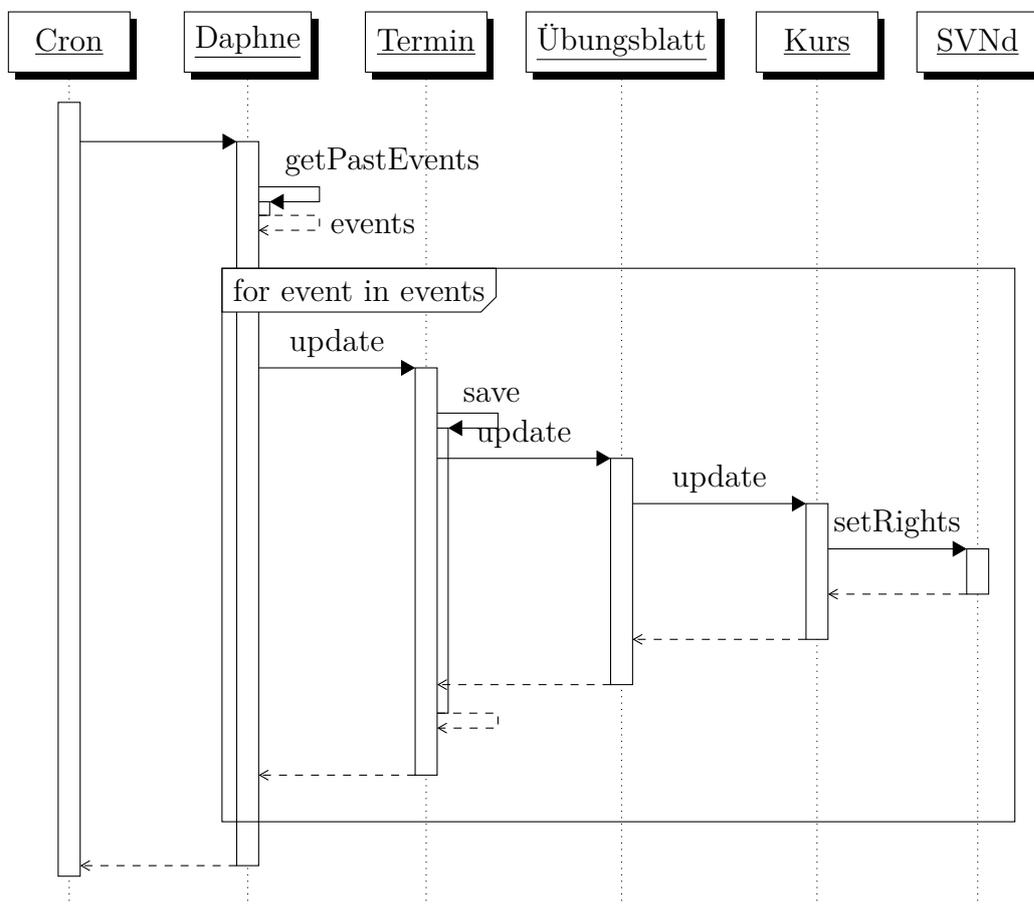


Abbildung 4.1.: Ablauf des zeitgesteuerten Rechtemanagements.

Termine können jederzeit ablaufen, daher wird jede Minute nach abgelaufenen Ter-

minen gesucht. Wird ein abgelaufener Termin erkannt, wird dessen Status, wie in Abbildung 4.1 dargestellt, auf abgelaufen gesetzt und verknüpfte Übungen werden informiert. Diese informieren den zugehörigen Kurs und dieser führt eine Aktualisierung der Zugriffsrechte der Repositorien aus.

4.2. CAS

Der zentrale Authentifizierungs Dienst (englisch: Central Authentication Service) basiert auf einer Arbeit der Universität Yale welche an das Jasig Konsortium¹ übergeben wurde. Das Kursverwaltungssystem übernimmt dabei die Rolle eines CAS-Providers und bestätigt anderen Diensten gegenüber die Identität des Benutzers.

[AMJ] beschreibt das im Folgenden vereinfacht dargestellte Verfahren. Die Identifizierung des Benutzer erfolgt über ein vom CAS-Provider gesetztes Cookie, welches an die Domain des CAS-Providers gebunden ist. Dienste, die die Identität eines Benutzer bestätigt wünschen, leiten den Benutzer mit zusammen mit einem Request-Token an die Adresse des CAS-Providers, dieser liest und validiert das Token sowie den vom Benutzer übermittelten Cookie. Sollte der CAS-Provider kein Cookie erhalten, präsentiert dieser automatisch das Anmeldeformular. Nach erfolgreicher Authentifizierung gegen den CAS-Provider leitet dieser den Benutzer zurück zu dem Dienst, welcher die Identität benötigt.

Da das Forum eine aktuelle E-Mail-Adresse für die Zustellung von Informationen über neue Beiträge benötigt, wird diese ebenfalls an den anfragenden Dienst übermittelt.

4.3. ØMQ Schnittstelle

Die Anbindung externer Dienste (SVN, Jenkins) wird über eigene Prozesse realisiert, welche auf der Maschine des Dienstes ausgeführt werden. Dazu wird jeweils das API-File für den Dienst kopiert, da dieses alle benötigten Klassen enthält, welche es ermöglichen sowohl die externen Dienste direkt zu steuern, als auch die Anbindung innerhalb des Kursverwaltungssystems zu realisieren. Realisiert wird die Anbindung der Prozesse durch ØMQ². Sowohl die Anfragen als auch die Antworten werden als JSON-Dictionary formatiert, welches alle benötigten Informationen enthält.

```
1 {
2   "parameter": "Lehrveranstaltung",
3   "command": "create",
4   "repository": "Lehrveranstaltung",
```

¹<http://www.jasig.org/cas>

²<http://zeromq.org>

```
5  "module": "svnadmin",
6  "revision": null
7 }
```

Listing 4.1: JSON-Nachricht zum Anlegen eines Repositoriums

```
1 {
2   "url": "job/test1/buildWithParameters",
3   "data": "SHEETNO=04",
4   "method": "POST"
5 }
```

Listing 4.2: JSON-Benachrichtigung zum Bauen des vierten Übungsblattes für den Job *test1*

Der Daemon ergänzt den übertragenen *URL*-Parameter um die Adresse des zugeordneten Build-System und führt einen *POST*-Request mit dem Wert aus dem *data*-Parameter an diese URL aus.

```
1 {
2   "action": "post_commit",
3   "repository": "/path/to/svn/Lehrveranstaltung",
4   "module": "svn",
5   "revision": 42
6 }
```

Listing 4.3: JSON-Benachrichtigung über die Erstellung der Revision 42 im Repository der Lehrveranstaltung.

Daphne wird angewiesen den für das Repository *Lehrveranstaltung* hinterlegten Post-Commit-Hook auszuführen. Durch die Übermittlung der erzeugten Revision ist der Hook in der Lage weitere Informationen zu diesem Commit beim SVN-Daemon abzufragen.

Das Back-End führt in Abbildung 4.2 nach dem Erhalt der *post-commit* Benachrichtigung das für das korrespondierende Repository hinterlegte *Post-Commit-Skript* aus. Dieses fordert in der Regel Informationen der Änderung beim zuständigen *SVN-Daemon* an. Diese Informationen beinhalten den Benutzername des Autors der Revision sowie die Liste der veränderten Dateien. Wenn der Benutzername mit dem Besitzer der veränderten Dateien übereinstimmt, so wird Jenkins beauftragt, die Abgabe zu bauen. Auf diese Weise wird verhindert, dass Korrekturen der Tutoren neue Builds starten.

4.4. Subversion-Daemon

Die Anbindung und Kontrolle der Subversion-Repositories erfolgt über einen eigenen Prozess. Diese Trennung erlaubt das Einbinden von Repositories welche sich auf

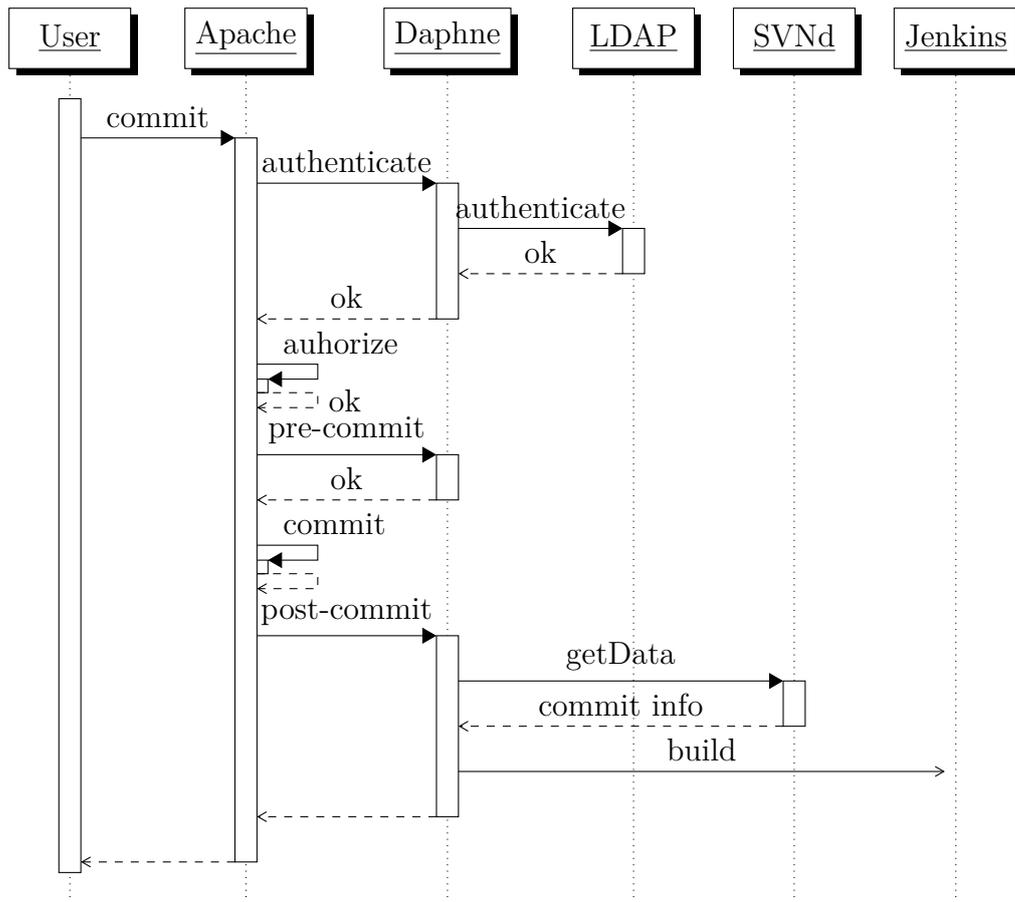


Abbildung 4.2.: Erfolgreicher Durchlauf nach dem erzeugen einer neuen Revision im Subversion Repositorium.

externen Servern befinden. Es besteht daher die Möglichkeit, dass ein Lehrstuhl das System verwendet, aber vollen Root-Zugriff auf das Repositorium hat, da es sich auf den Servern des Lehrstuhles befindet. In diesem Fall muss nur der Kontrollprozess des Kursverwaltungssystems auf diesem Server ausgeführt werden. Zu den Aufgaben des Prozesse gehört die Verwaltung der Zugriffsrechte, die Bereitstellung von im Repositorium hinterlegten Dateien und Informationen zu einzelnen Revisionen.

Die Zugriffsrechte werden im Dateisystem für jeden Kurs einzeln hinterlegt und danach vom SVN-Daemon zu einer Datei zusammengeführt. Für jeden Kurs werden zugeordnete Gruppen einzeln verwaltet, damit allen Teilnehmer, Tutoren und angemeldeten Personen unmittelbar Rechte zugewiesen werden können. Da jeder Teilnehmer für sich selbst eine Gruppe bildet und Gruppen nur einmalig definiert werden dürfen, werden die einzelnen Gruppdateien zusammengefasst und von Duplikaten bereinigt. Die effektiven Zugriffsrechte werden an die aufbereiteten Gruppdateien angehängt.

1 svn/

```
2 +- Repository_01/
3 | +- ...
4 | +- access.txt
5 | +- access_additions.txt
6 | \- access_groups.txt
7 +- Repository_02/
8 | +- ...
9 | +- access.txt
10 | +- access_additions.txt
11 | \- access_groups.txt
12 +- ...
13 \- access_additions.local.txt
```

Listing 4.4: Ausschnitt aus dem Verzeichnisbaumes zur SVN Rechteverwaltung.

Die Zugriffsrechte werden nach Kurs sortiert angehängt, wobei die von Verwaltern eingetragenen Rechte nach den vom System erzeugten Rechten hinzugefügt werden, damit diese bereits gesetzte Rechte überschreiben. Auch kann eine feste Anpassung durch verwenden der *access_additions.local.txt* auf Repository Ebene durchgeführt werden. Nachdem die Rechte für die einzelnen Repositorien aus deren Verzeichnissen generiert wurden wird abschließend die globale *access_additions.local.txt*-Datei angehängt, welche für Zugriffsrechte von Repositorien enthalten kann, welche nicht von Daphne verwaltet werden.

Das Anstoßen des in Abbildung 4.2 beschriebene Ablaufes liegt ebenfalls im Aufgabenbereich des SVN-Daemon. Die Anfrage wird durch den Post-Commit-Hook (in der Abbildung als Apache bezeichnet) des Subversion-Repository gestartete und läuft danach autonom ab.

4.5. Jenkins-Daemon

Der Jenkins-Daemon-Prozess beinhaltet eine Umsetzung der Remote Access API³ welche einem autorisierten Benutzer umfangreichen Zugriff gewährt. Zu diesem Zweck meldet sich der Daemon als Jenkins-Administrator an und kann so die Verwaltung der Buildjobs übernehmen.

Das Buildsystem verwendet für jeden Job eine gesonderte Konfigurationsdatei, damit diese nicht für jeden Job einzeln gepflegt werden muss ist die Verwendung eines Templates in das Front-End eingebaut. Dieses Template ermöglicht es Platzhalter zu verwenden, welche nach dem Speicher vom Back-End ausgefüllt und für jede Gruppe eingesetzt werden. Sobald eine Gruppe verändert wird aktualisiert dies automatisch die Build-Konfiguration.

³<https://wiki.jenkins-ci.org/display/JENKINS/Remote+access+API>

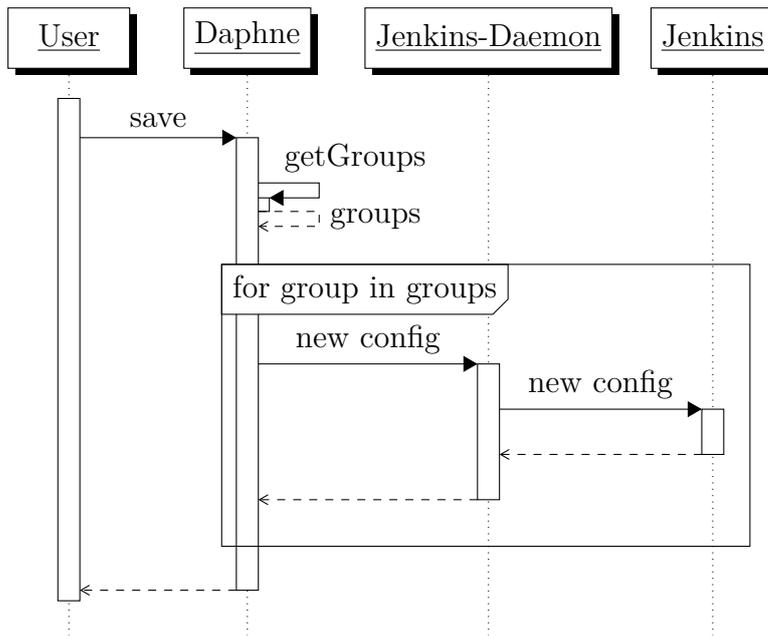


Abbildung 4.3.: Verteilung der Build-Konfigurationen

5. Evaluation

Das System wurde sowohl mit einem an der Technischen Fakultät etablierten Portal als auch externen Systemen verglichen. Dabei wurden Teilbereiche, welche an der Technischen Fakultät durch externe Systeme wie QIS/POS/LSF übernommen werden nicht näher betrachtet. In diese Bereiche fällt die Verwaltung von Räumen, Prüfungsterminen und weitere organisatorische Funktionen wie die Verwaltung von gesamt Noten und ähnlichem.

5.1. Verwandte arbeiten

Der allgemeineren Problemstellung, der Verwaltung von Lehrveranstaltungen, haben sich viele Projekte mit unterschiedlichen Schwerpunkten angenommen.

5.1.1. CampusOnline & ILIAS

Diese Systeme werden derzeit an der Universität Freiburg eingesetzt und stehen sowohl Lehrenden wie auch Lernenden zur Verfügung.

CampusOnline¹ ist in der Lage umfangreiche Materialsammlungen zu verwalten sowie Abgaben innerhalb des Bearbeitungszeitraumes per Datei-Upload anzunehmen. Das System soll in naher Zukunft durch die Open-Source Alternative ILIAS² abgelöst werden.

Diese Systeme sind für allgemeine Lehrveranstaltungen und Fortbildungen optimiert, jedoch nicht für die Abgabe von Programmen beziehungsweise Quelltext, da keine automatischen Verifikationen für diese Abgabeformen eingebunden sind, wie es in Daphne der Fall ist. Dafür beinhalten diese Systeme die Möglichkeit von Selbst-Tests, Vorlesungsverwaltung und Mediendatenbanken.

5.1.2. Moodle

Moodle³ ist vom grundlegenden Aufbau vergleichbar mit ILIAS. Beides sind Open-Source Projekte welche Lernende in vielen Bereichen unterstützen sollen. Moodle

¹<http://www.rz.uni-freiburg.de/services/elearning/ewerkzeuge/campusonline>

²<http://www.rz.uni-freiburg.de/services/elearning/ewerkzeuge/lmsilias/ilias>

³<https://moodle.org/>

wird von Schulen im Rahmen des Landesbildungsservers Baden-Württemberg eingesetzt, jedoch fehlt wie bei ILIAS eine Spezialisierung für Quelltexte und Programme.

5.2. Vergleich mit anderen Portalen

Der Lehrstuhl für Rechnerarchitektur betreibt ein eigenes Portal welches bei verschiedenen Vorlesung verwendet wird. Die theoretische Vorlesung *Technische Informatik*⁴ sowie die praktische Vorlesung *Mobiles Hardware-Praktikum*⁵ werden durch dieses Portal unterstützt.

Die Portale bieten Benutzern nach der Registration die Möglichkeit sich selbständig in Abgabegruppen zu organisieren. Dabei werden jedoch für jede Vorlesung neue Kennungen benötigt. Diese Registration entfällt bei Daphne, da nur Portal für alle verwalteten Kurse verwendet wird.

Den Portalen ist darüber hinaus ein Forum⁶ (Phorum⁷) zugeordnet, bei dem sich Teilnehmer mit beliebigen Pseudonymen registrieren können. An Daphne ist ein phpBB3 angeschlossen, bei dem die vom Rechenzentrum vorgegebenen Benutzerkennungen verwendet werden. Diese sind für Außenstehende ebenfalls nicht zuzuordnen, können aber durch Aufbau und Benutzung Lehrenden oder Kommilitonen zugeordnet werden.

Der größte Unterschied sind die zur Verfügung gestellten Abgabeformen, während Daphne diese Funktionalität komplett an Subversion überträgt, erlauben die Portale des Lehrstuhl für Rechnerarchitektur den komfortablen Upload über ein Web-Formular. Ein erneutes Hochladen einer Datei ersetzt die bereits vorhandene vollständig, wodurch zwar Speicherplatz gespart wird, jedoch ein älterer Stand nicht mehr hergestellt werden kann. Diese Einschränkung umgeht Daphne durch die Verwendung von Subversion.

Es ist zu erkennen, dass die Systeme mit unterschiedlichen Anforderungen konzipiert und umgesetzt wurden, daher haben beide Systeme ihre Anwendungsbereiche.

5.3. Benutzerfeedback

Die folgenden Zitate stammen von Benutzern, die Daphne bereits verwenden und testen konnten. Der Personenkreis umfasst hierbei Bachelor und Master Studenten

⁴<https://webira.informatik.uni-freiburg.de/cgi-bin/teaching/ti-ws-1213/login.cgi>

⁵<https://webira.informatik.uni-freiburg.de/cgi-bin/teaching/mhwp-2013-ss/portal/login.cgi>

⁶<http://ira.informatik.uni-freiburg.de/phorum/index.php>

⁷<http://www.phorum.org/>

verschiedener Studiengänge. Die kompletten Meinungen können auf dem Wiki des Lehrstuhl für Algorithmen und Datenstrukturen⁸ nachgelesen werden.

Gute Organisation der Übungen (SVN und Jenkins)!

... sowie einem ausgefeilten Online-Kurssystem inklusive Wiki, SVN und Buildserver nachvollziehen kann. Keine andere Veranstaltung ist mit Live-Coding und aktuellen Entwicklungswerkzeugen (Stichwort Buildserver, Checkstyle/Lint, etc.) so nah an der Praxis.

Die Benutzung von Daphne/Svn/Jenkins fand ich auch gut, damit man das auch mal kennenlernt, außerdem war das schön übersichtlich und gut erklärt.

Besonders positiv überrascht war ich vom Daphne System, G-Test und cpplint.

Mit Abstand bestes Angebot an Materialien (Aufzeichnung, Forum, SVN-Server, Jenkins ...).

Gute Arbeitsumgebung bei den Übungen durch Daphne, Jenkins und Forum.

Great exercise-system: SVN, Daphne, ...

⁸<http://ad-wiki.informatik.uni-freiburg.de/teaching/>

6. Ausblick

6.1. Allgemeine Arbeiten am System

Zukünftige Arbeiten am Kursverwaltungssystem umfassen die Vervollständigung der Übersetzungen und Benutzeroberflächen. Dies umfasst die Anzeige relevanter Informationen aus dem Log bei Detailansichten.

An sinnvollen Stellen werden durch die Verwendung von JavaScript die Arbeitsabläufe vereinfacht. Frühzeitig wird hierbei die Eingabe der Punkte für Tutoren und Kursleiter in der Art verbessert, dass die Punkte ohne Verlassen der Seite eingetragen werden können. Die Anzeige von Systeminformationen beim Abschluss von Hintergrundaufgaben soll ebenfalls eine JavaScript Erweiterung erhalten, damit neue Statusmeldungen ohne erneutes laden der Seite angezeigt werden.

Die Anpassung der Anzeige der Statusmeldungen wird außerdem für die Umrüstung des Signal Systems welches von Django bereitgestellt wird benötigt. Das System arbeitet in der Standardausführung synchron und soll auf ein asynchrones System umgestellt werden, damit die Hintergrundprozesse das Laden der Seite nicht weniger stark beeinflussen.

6.2. Neue Funktionen

Gewünscht wird eine Funktion ähnlich zu Reviewboard¹, die es erlaubt Zeilen in Dateien zu annotieren und zu diskutieren. Diese Funktionalität soll Tutoren entlasten, da Anmerkungen direkter an Problemstellen, ohne Einfügen von Kommentarzeilen im Quelltext, hinterlassen werden können. Die Umsetzung dieser Funktionalität wird vermutlich durch einen Ersatz von WebSVN als SVN-Front-End durch eine angepasste Eigenentwicklung erfolgen, welche darüber hinaus weitere Möglichkeiten bieten wird. Angedacht sind Datei-Upload-Felder die Übungsblättern zuzuordnen sind und die so hochgeladenen Dateien direkt in das Subversion-Repository einzupflegen.

¹<http://www.reviewboard.org/>

Danksagung

Danken möchte ich Jens Hoffmann, der mir sowohl die Möglichkeit eröffnet hat mich mit diesem Thema auseinander zu setzen also auch bei konzeptuellen Fragen zur Seite stand.

Johanna Götz möchte ich für die Unterstützung beim Umsetzen des Corporate Design sowie für die aufgebrauchte Zeit zum Testen neuer Funktionen danken.

Außerdem danke ich meinen Eltern, welche mir Freiraum sowohl für die intensive Arbeit an der technischen Seite der Bachelorarbeit gewährten wie auch für die zeitraubende Arbeit an der Ausarbeitung.

A. Anhang

Literaturverzeichnis

- [AMJ] AUBRY, Pascal ; MATHIEU, Vincent ; JULIEN, Mar-
chal. *ESUP-Portail: open source Single Sign-On
with CAS (Central Authentication Service)*. Webseite,
http://immagic.com/eLibrary/ARCHIVES/GENERAL/ESUP_FR/E040427A.pdf
- [Pië10] PIËL, Nicholas. *ZeroMQ an introduction*. Webseite,
<http://nichol.as/zeromq-an-introduction>. 2010
- [Sim07] SIMONSON, Michael (Hrsg.): *The Quarterly Review of Distance Education*.
Bd. 8. Information Age Publishing, Inc., 2007
- [Zho10] ZHOU, Yirhan. *PROJECT REPORT OF BUILDING COURSE
MANAGEMENT SYSTEM BY DJANGO FRAMEWORK*. Web-
seite, [https://www.cs.sfu.ca/content/dam/sfu/computing/](https://www.cs.sfu.ca/content/dam/sfu/computing/Undergraduate_students/YiranZhou.pdf) Undergradu-
ate_students/YiranZhou.pdf. 2010

Abbildungsverzeichnis

2.1. Verteilung der Anfragen in Abhängigkeit des Pfades	6
3.1. Anmelde Formular des Kursverwaltungssystems.	11
3.2. Anmelde Formular des Kursverwaltungssystems nach fehlerhaftem Anmeldeversuch. Hinweis auf die Beachtung von Groß- und Kleinschreibung von Benutzername und Kennwort.	12
3.3. Allgemeine Fehlerseite, ausgelöst durch zu schnelles übermitteln der Anmeldeinformationen nach einem gescheiterten Anmeldeversuch.	12
3.4. Anzeige der persönlichen Daten nach erfolgreichem Login.	13
3.5. Organisation der Informationen zu Lehrveranstaltungen des Lehrstuhles für Softwaretechnik.	14
3.6. Im Menü und System eingetragene Vorlesungsperiode.	15
3.7. Formular zum Anlegen eines Kurses. Es können optionale Daten hinterlegt werden.	16
3.8. Angelegter Kurs, welcher automatisch im Menü eingetragen wurde.	17
3.9. Kursinformationen für einen Teilnehmer	18
3.10. Kursinformationen für einen Tutor	19
3.11. Startseite der Universität Freiburg. Zusätzliche Rahmen markieren Portlets im Seitenrandbereich.	20
3.12. Abstimmung über die Uhrzeit des Vorkurses.	21
4.1. Ablauf des zeitgesteuerten Rechtenmanagements.	23
4.2. Erfolgreicher Durchlauf nach dem Erzeugen einer neuen Revision im Subversion Repository.	26
4.3. Verteilung der Build-Konfigurationen	28

Listings

2.1. Supervisor-Eintrag der Daphne Komponenten	3
2.2. Definition SVN Virtual-Host (Apache)	5
2.3. Definition SVN-Location (Apache)	5
2.4. SSL Einstellungen (Nginx)	6
2.5. Definition SVN-Location (Nginx)	6
2.6. Modell der Übungsliste	7
2.7. Doc-String-Test des Breadcrumb Identitätsvergleich	8
3.1. Ausschnitt aus dem Logeintrag beim Anlegen der Kurse	19
4.1. JSON-Nachricht zum Anlegen eines Repositoriums	24
4.2. JSON-Benachrichtigung zum Bauen des vierten Übungsblattes für den Job <i>test1</i>	25
4.3. JSON-Benachrichtigung über die Erstellung der Revision 42 im Repos- itorium der Lehrveranstaltung.	25
4.4. Ausschnitt aus dem Verzeichnisbaumes zur SVN Rechteverwaltung. . .	26

