Dokumentation zur Anwendung "Newsticker" im Rahmen des Moduls Web-basierte Anwendungen 2: Verteilte Systeme SS2013

Daniela Kucharczuk
Paul Will

Einführung

Im Laufe der Veranstaltung Web basierte Anwendungen sind wir in die zweite Phase angekommen. Es gab eine Einführungsveranstaltung zu der Phase, wo einzelne Gruppen gebildet wurden mit jeweils zwei Teilnehmern. In der zweiten Phase musste jede Gruppe sich ein Projekt konzipieren, in dem ein Client-Server Architektur zum Einsatz kommt. Außerdem musste zwischen beiden Systemen eine Synchrone und Asynchrone Kommunikation stattfinden. Ein Teil der Funktionen des konzipierten Projektes musste anschließend implementiert werden. Parallel mit dem Projekt wurden einzelne Meilensteine festgelegt, die durch einzelne Schritte von Projektmentoren diskutiert wurden. Eine Projektdokumentation, als auch Logbuch gehörten zum Teil der Aufgabe. Dadurch wurden einzelne Ergebnisse bzw. Gedankengänge zur Ideenfindung und ihren Lösungen festgehalten.

Konzept Newsticker

Unser System soll eine Möglichkeit bieten Interessengebiete, die das aktuelle Semester betreffen, zu organisieren. Durch das abonnieren eines Interessengebiets sind alle Informationen, die diesbezüglich eingestellt sind, abrufbar. Dozenten werden sowohl Inhalte über das System erstellen, speichern und editieren, als auch aktuelle Neuigkeiten bekannt geben können. Diese Meldungen, aber auch weitere wissenswerte Informationen, wie zum Beispiel anstehende Pflichttermine, werden per Newsticker an die Abonnenten gesendet.

Meilenstein 1 : Projektbezogenes XML Schema / Schemata

Die Planung und Konzeption für das Projekt nötiger valider XML Schemata soll durchgeführt werden. Hinsichtlich der späteren Verwendung von JAXB sollte hier in jedem Fall darauf geachtet werden, dass die XML Schemata valide sind.

XML-Dokumente:

Dozent: Sequenz mit Daten eines Dozenten
Student: Sequenz mit Daten eines Studenten
Modul: Sequenz mit Daten eines Moduls (Abonnements)

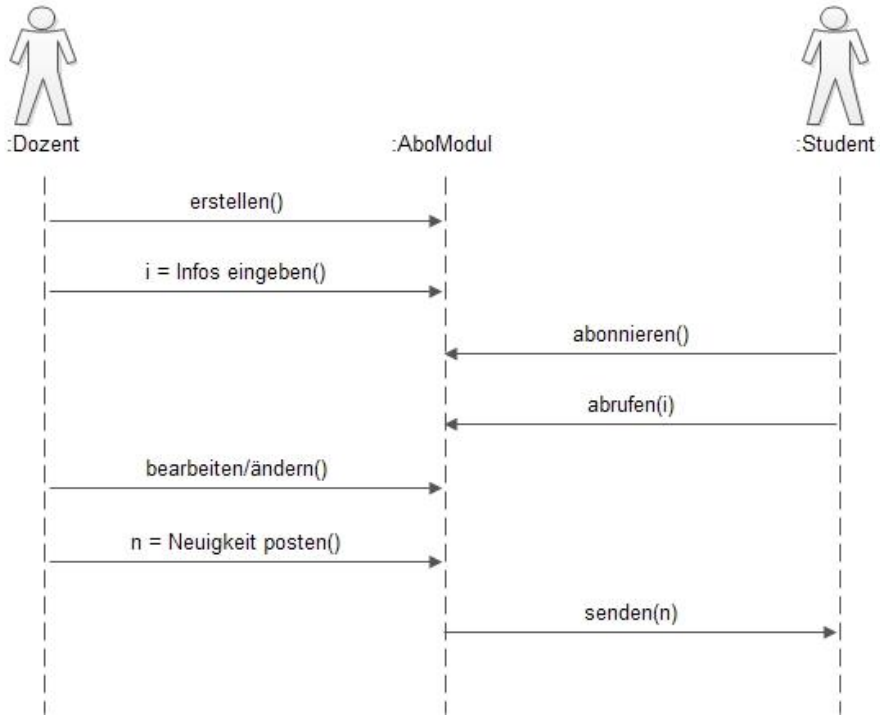
asynchrone Kommunikation:

- zu folgenden Angaben können Abos abgeschlossen werden:
- Modul, Dozent, Semester, Studiengang, Zeitraum (WS, SS)
- die Art des abgeschlossenen Abos legt fest, welche Informationen per Newsticker versendet werden
- Dozenten können Neuigkeiten bekannt geben über die per Newsticker informiert wird

synchrone Kommunikation:

- über das System können nach Abschluss eines Abos diesbezüglich alle vorhandenen Informationen abgerufen werden:
  - Modul: Veranstaltungszeitpunkt, Veranstaltungsraum, Dozent, Credits, Beschreibung, Übungen, ggf. Praktikum
  - Dozent: Fachrichtungen, laufende/vergangene Veranstaltungen, Kontakt
  - Semester: Modulübersicht, Praktika, Übungen, Prüfungsfristen/-zeitraum, Stundenplan
  - Studiengang: Studienverlaufsplan, Prüfungsordnung, Beschreibung, Regelstudienzeit, Studienumfang, Events
  - Zeitraum: Übersicht Modulangebot, Prüfungszeitraum, Rückmeldefrist, Blockveranstaltungen
- Dozenten können über das System Beiträge erstellen, speichern und editieren, z.B.:
- Modulverlaufsplan, Beschreibung, Vorkenntnisse, Erwartungen, ähnliche Module, Teilnehmerzahl,
- Credits/Leistungsumfang, Pflichttermine

Szenario und Interaktion



Anhand dieses Sequenzdiagramm ist eine synchrone sowohl als auch asynchrone Kommunikation zu sehen. Der Dozent erstellt ein Thema mit jeweiligen Informationen und schickt die Daten an das Modul. Falls ein Student sich für einen Studiengang, einen Fach oder eine Person interessiert sein sollte, kann er/sie das Interessensgebiet abonnieren. Alle Änderungen, die zu dem Thema eintreten, werden ebenfalls an dem Abonnent gesendet.

Meilenstein 2: Ressourcen und die Semantik der HTTP-Operationen für das Projekt

Eingangs soll eine theoretische Auseinandersetzung mit HTTP-Operationen und Ressourcen im Kontext RESTful Webservices erfolgen. Das erworbene Wissen ist Grundlage um die nächsten Schritte erfolgreich absolvieren zu können.
Es sollen projektbezogene Beschreibungen der für das Projekt benötigten Ressourcen und Operationen inklusive Begründung der Entscheidungen erstellt werden.

Ressource: Dozent

Beschreibung	HTTP Methode	URI
Alle Dozenten laden	GET	/dozenten/
Dozent laden	GET	/dozenten/{id}/
Dozent erstellen	POST	/dozenten/
Dozent aktualisieren	PUT	/dozenten/{id}/
Dozent löschen	DELETE	/dozenten/{id}/
Neuigkeit posten	POST	/dozenten/{id}/news/

Ressource: Student

Beschreibung	HTTP Methoden	URI
Student erstellen	POST	/student/
Student laden	GET	/student/{id}/
Student aktualisieren	PUT	/student/{id}/
Student löschen	DELETE	/student/{id}/
Abo abschließen	PUT	/student/{id}/
Abo löschen	DELETE	/student/{id}/abos/{id}/
Neuigkeit laden	GET	/news/{id}/
Alle Neuigkeiten laden	GET	/news/

Ressource: Modul

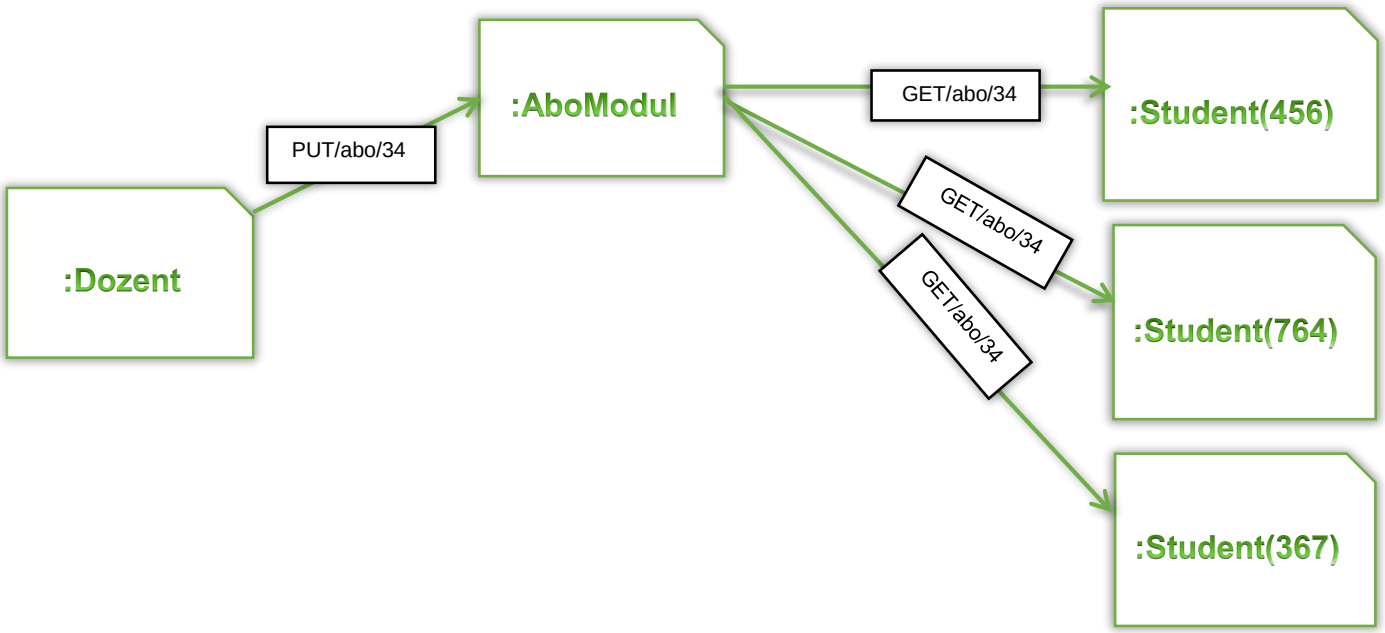
Beschreibung	HTTP Methoden	URI
Modul erstellen	POST	/dozenten/{id}/ module /
Modul aktualisieren	PUT	/dozenten/{id}/ module /{id}/
Modul laden	GET	/module/{id}/
Alle Module laden	GET	/ module /
Modul löschen	DELETE	/dozenten/{id}/ module /{id}/
Modul suchen	GET	/ module /
Neuigkeit posten	POST	/dozenten/{id}/module/{id}/news/

Ressource: Semester

Beschreibung	HTTP Methoden	URI

Semester erstellen	POST	/dozenten/{id}/semester/
Semester aktualisieren	PUT	/dozenten/{id}/semester/{id}
Semester laden	GET	/semester/{id}
Alle Semester laden	GET	/semester/
Semester löschen	DELETE	/dozenten/{id}/semester/{id}
Semester suchen	GET	/semester/
Neuigkeiten posten	POST	/dozenten/{id}/semester/{id}/news/

Unser System enthält mehrere Ressourcen (u.a. Dozent, Student, Module, Semester, Studiengang etc.), dabei lässt sich eine Interaktion mit dem System deutlich zwischen dem Dozenten und Student zum Einsatz kommen. Diese Ressourcen sind wesentliche Bestandteile unsers Systems. Wobei der Dozent wird als wichtigste Ressource festgelegt, da er als einziger darf Informationen publizieren. Hinterher folgt Student, der die publizierte Informationen abonniert.



Meilenstein 3: RESTful Webservice

Ein RESTful Webservice soll in Java erstellt werden. Dazu gelten folgenden Bedingungen:

- Mindestens zwei Ressourcen müssen implementiert sein
- JAXB soll für das marshalling / unmarshalling verwendet werden
- Die Operationen GET, DELETE und POST müssen implementiert sein
- Es sollen mindestens einmal PathParams und einmal QueryParams verwendet werden

Die Ergebnisse vom Meilenstein 3 wurden umgesetzt und sind im Paket restfulWebservice zu sehen.

Meilenstein 4: Konzeption + XMPP Server einrichten

Es handelt sich hierbei um konzeptionellen Meilenstein für die genauere Planung der asynchronen Kommunikation. Dazu sollen folgende Aufgaben erledigt werden:

- Leafs (Topics) sollen für das Projekte definiert werden
- Wer ist dabei Publisher und wer Subscriber?
- Welche Daten sollen übertragen werden?
- Zusätzlich soll der XMPP Server installiert und konfiguriert werden (sofern auf eigenen Geräten gearbeitet wird)

Unsere Entscheidung ist auf folgende Topics gefallen:

- Dozent
- Module

es wird eine Möglichkeit dem Subscriber (Student) gegeben sich aus eine Liste einen oder mehrere Dozenten zur abonnieren und Informationen, die diese Personen betreffen, zu laden. Ebenfalls werden auch Module von Subscriber abonniert und per asynchrone Kommunikation, mit Hilfe Extensible Messaging und Presence Protocol (XMPP), zur Verfügung gestellt. Die Module werden nur von Publisher erstellt und aktualisiert. Dort sind einzelne Fächer aufgelistet und deren Status angezeigt. Eine Aktualisierung wird nur bei Modulen mit dem Status „Online" möglich. Für diese Aufgabe haben wir diese beiden Topics verwendet. Natürlich sind weitere Topics erweiterbar.

Die Rollen für Publisher und Subscriber waren uns direkt klar und wurden dementsprechend verteilt. Als Publisher war der „Dozent" eingeteilt, da er für die Erstellung und Aktualisierung der Topics zuständig ist. Folglich sind die Subscriber mit Rolle des „Student" zu besetzen, weil dieser die Topics abonniert.

Übertragene Daten sind alle Änderungen bzw. Updates, die in Abonnierten Topics, vorgenommen wurden.

Meilenstein 5: XMPP - Client

Es sollte ein XMPP Client entwickelt werden, dafür musste folgendes berücksichtigt werden:

- Mittels der Anwendung soll es möglich sein Leafs zu abonnieren, Nachrichten zu empfangen und zu veröffentlichen.

- Ermöglichen Sie die Übertragung von Nutzdaten (Beispielsweise Simplepayload)

- Leafs und ggf. mögliche Eigenschaften der Leafs sollen angezeigt werden können (Service Discovery)

Meilenstein 6: Client - Entwicklung

Abschließend sollte ein Client erstellt werden, welcher das Projekt repräsentiert. Dafür ist eine Nutzung des REST Webservices und des XMPP Servers erforderlich:

- Das System (RESTful Webservices sowie XMPP Server) muss über ein grafisches User Interface nutzbar sein

- Die Verwendung von Swing wird empfohlen

Getrenntes Ausführen auf unterschiedlichen Systemen von Client und Server /Webservice sollte optimaler Weise möglich sein.

Die Ergebnisse der Umsetzung des letzten Meilensteins sind unter dem Link zu erreichen.