Entwicklerdokumentation
Anbindung einer Linked Data Quelle

Inhaltsverzeichnis

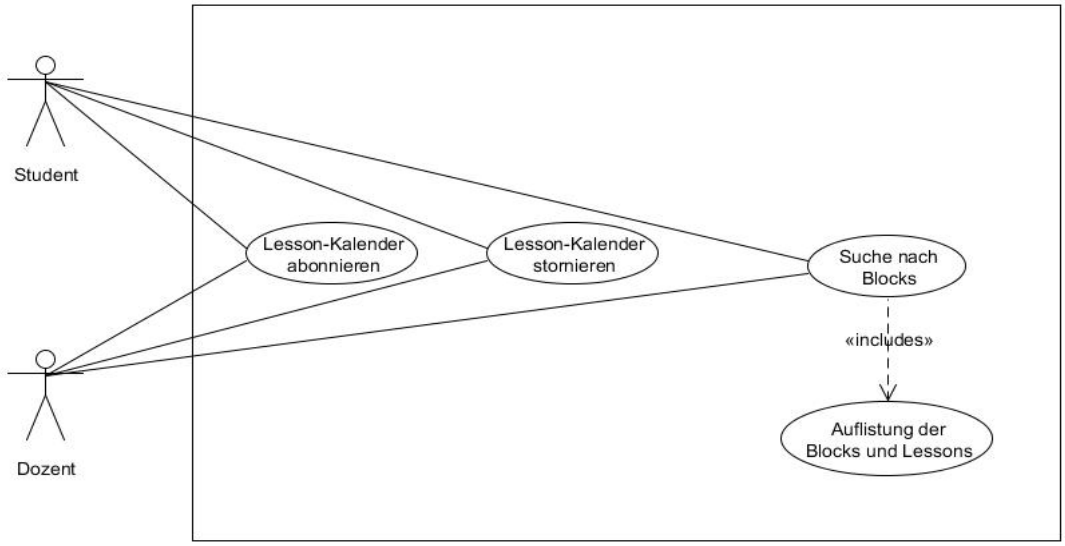
1. Allgemeines

OLAT bietet ein flexibles Kurssystem und unterstützt das kollaborative Arbeiten in Gruppen, mittels einer einfach zu bedienenden und mehrsprachigen Benutzeroberfläche, sowie verschiedenen Groupwaretools.

Das Projekt hat zum Ziel das bestehende OLAT System, um eine Stundenplanfunktion zu erweitern.

2. Produktübersicht

Das Produkt soll dem Nutzer eine Funktion zur Verfügung stellen, mit der er eine Übersicht über alle Lehrveranstaltungen hat. Zusätzlich erhält man einen Stundenplan, welcher alle abonnierten Lehrveranstaltungen anzeigt.



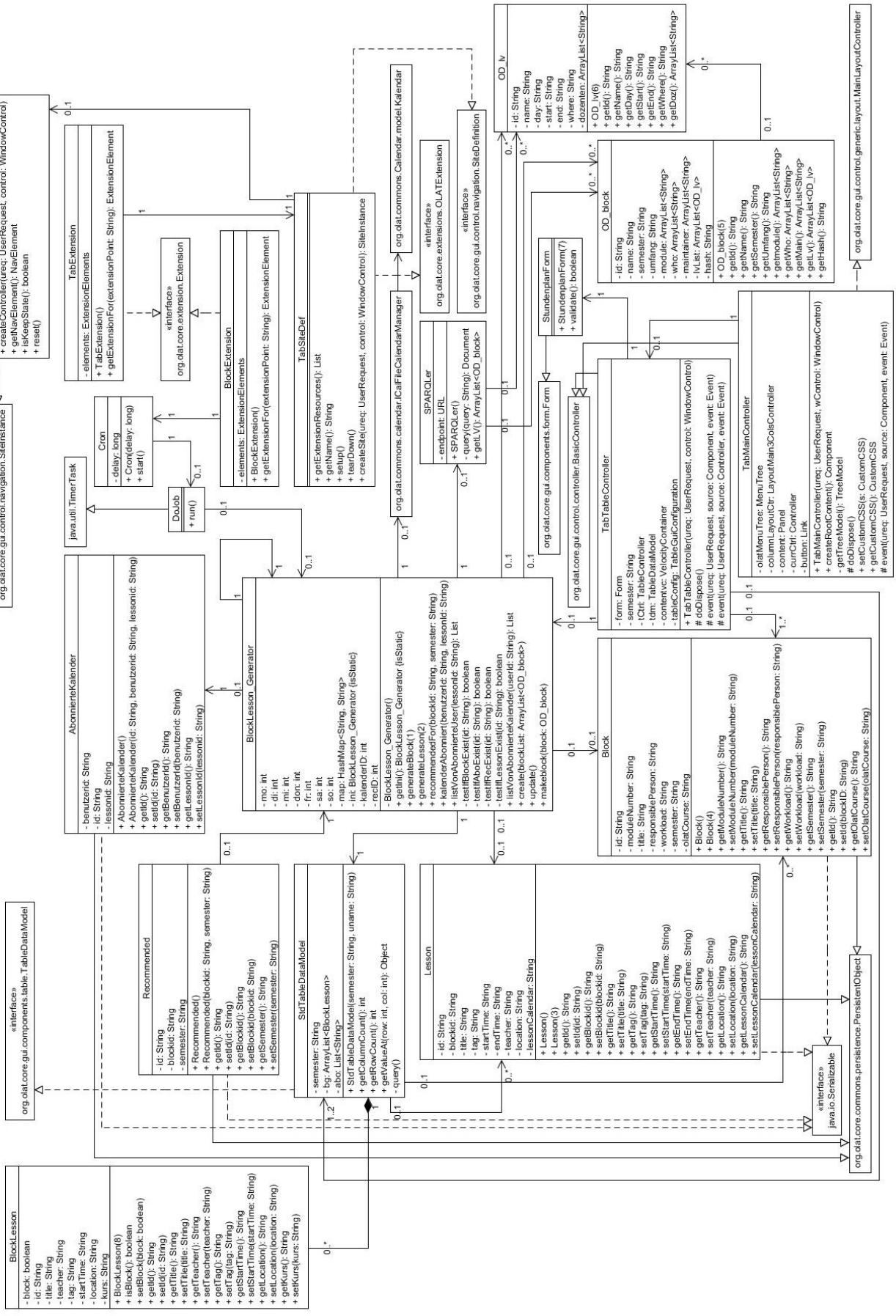
Use-Case-Diagramm

Sowohl Dozenten als auch Studenten können sich mit der Gesamtübersicht Veranstaltungsblöcke (Blocks) und deren einzelne Veranstaltungen (Lessons) anzeigen lassen. Um eine bestimmte Veranstaltung zu finden, kann der Nutzer sein Semester auswählen.
Zu jeder Veranstaltung werden Titel, Wochentag, Zeit, Dozent und Semester angezeigt.

Durch Selektierung einer aufgelisteten Veranstaltung kann der zugehörige Kalender abonniert werden. Die Veranstaltung ist dann im persönlichen Kalender eingetragen und kann vom Stundenplan-Tab aus wieder ausgetragen werden.

3. Grundsätzliche Struktur- und Entwurfsprinzipien für das Gesamtsystem

Das Produkt baut auf die MVC-Architektur auf und teilt sich in verschiedene Pakete. Im nachfolgenden Klassendiagramm ist die Struktur des Systems dargestellt.



Klassendiagramm

ergänzende Parameterlisten (Nummern sind im Klassendiagramm vermerkt):

1:
id: String, moduleNumber: String, title: String, responsiblePerson: String, workload: String, semester: String, olatCourse: String

2:
id: String, blockId: String, title: String, tag: String, startTime: String, endTime: String, teacher: String, location: String, lessonCalendar: String

3:
id: String, blockId: String, title: String, tag: String, startTime: String, endTime: String, teacher: String, location: String, lessonCalendar: String

4:
moduleNumber: String, title: String, responsiblePerson: String, workload: String, semester: String

5:
id: String, name: String, semester: String, umfang: String, module: ArrayList<String>, who: ArrayList<String>, maintainer: ArrayList<String>, lvList: ArrayList<OD_lv>

6:
id: String, name: String, day: String, start: String, end: String, where: String, dozenten: ArrayList<String>

7:
name: String, translator: Translator, wControl: WindowControl

8:
block: boolean, id: String, title: String, teacher: String, tag: String, startTime: string, location: String, kurs: String

Erläuterungen zu den Klassen:

BlockLesson_Generator
Zur Verwaltung der Datenstrukturen dient der BlockLesson_Generator.

DoJob
Die Klasse DoJob startet den Aktualisierungsvorgang der Klasse BlockLesson_Generator.

Cron
Der Cron stößt einen täglichen Aktualisierungsvorgang an.

SPARQLer
Der SPARQLer stellt Anfragen an die externe Datenquelle und übergibt dessen Antworten an den BlockLesson_Generator.

Block
Ein Block realisiert einen Lehrveranstaltungsblock und wird in der Datenbank abgelegt.

Lesson
Eine Lesson entspricht einer Lehrveranstaltung. Sie wird in der Datenbank gespeichert.

Recommended
Die Klasse Recommended ordnet jedem Block mehrere Studiengänge zu.

AbonnierteKalender
AbonnierteKalender ist eine Datenbankschnittstelle die jedem Nutzer seine abonnierten Kalender zuordnet.

OD_block
Lehrveranstaltungsblöcke werden vom SPARQLer in Form von Objekten der Klasse OD_block zurückgegeben.

OD_lv
Lehrveranstaltungen werden vom SPARQLer in Form von Objekten der Klasse OD_lv zurückgegeben.

TabSiteDef
Die Klasse TabSiteDef definiert die Ausprägung eines Tabs.

TabSite
Zur Erzeugung eines OLAT-Tabs ist die Klasse TabSite zuständig.

TabExtension
Die TabExtension wird als Erweiterung eingebunden, um den Tab zu platzieren.

StundenplanForm
Die Klasse StundenplanForm ist für die Darstellung der Tabelle im Tab verantwortlich.

StdTableDataModel
Die Klasse StdTableDataModel dient dem Aufbereiten der Daten für eine OLAT-Tabelle.

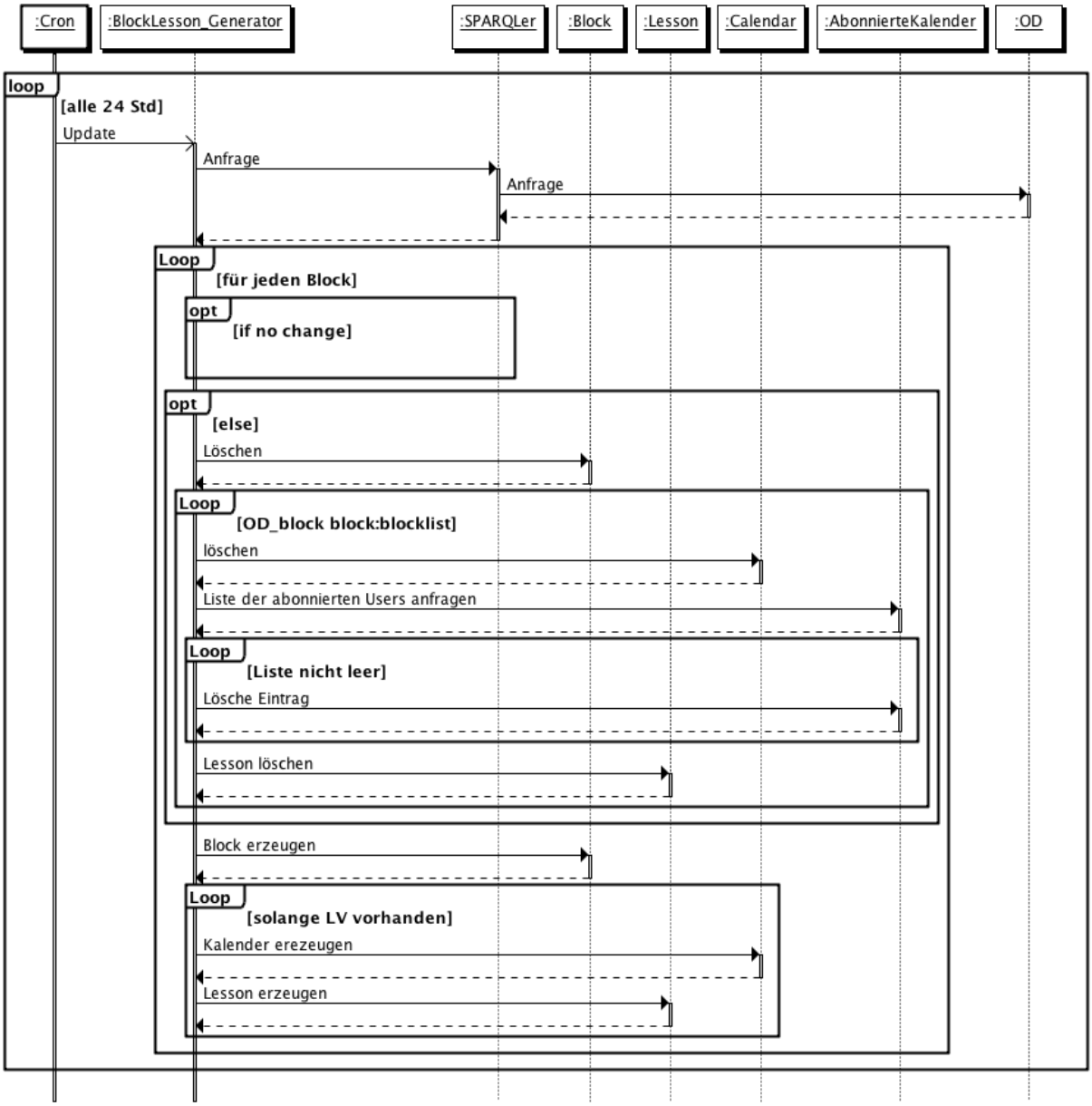
BlockExtension
Die Klasse BlockExtension platziert Daten aus der Datenbank in der Tabelle.

TabTableController
Der TabTableController realisiert Controller-Funktionalitäten für den Tab.

TabMainController
Die Klasse TabMainController ist für die Interaktion des Nutzers mit der Hauptseite des Tabs verantwortlich.

4. Grundsätzliche Struktur- und Entwurfsprinzipien für einzelne Pakete

4.1 Ablauf

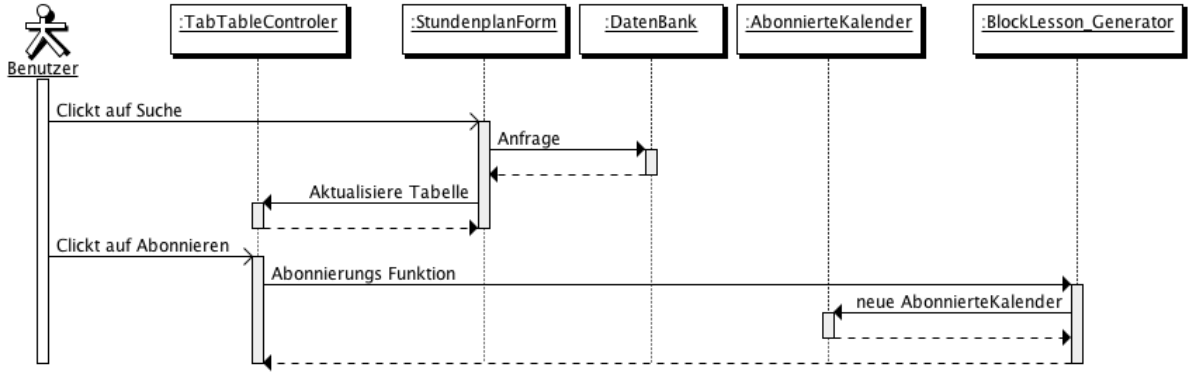


Sequenzdiagramm zum Aktualisierungsvorgang

Dieses Sequenzdiagramm stellt den Aktualisierungsvorgang des Datenbestandes dar. Aus Gründen besserer Lesbarkeit sind hier nur die wichtigsten Classifier dargestellt und es wurde auf Nennung einzelner Methodenaufrufe verzichtet. Der Classifier OD entspricht keiner Klasse, sondern repräsentiert nur die Schnittstelle der externen Datenquelle OD.

Die Datenbankeinträge werden alle 24 Stunden aktualisiert. Dazu erzeugt eine Instanz der Klasse Cron einen DoJob (nicht dargestellt), welcher die Methode update() des BlockLesson_Generators aufruft und damit den Aktualisierungsvorgang anstößt. Der BlockLesson_Generator erzeugt eine Instanz der Klasse SPARQLer. Diese stellt eine SPARQL-Anfrage an die externe Datenquelle OD, um alle Lehrveranstaltungsblöcke und Lehrveranstaltungen zu erhalten. Die Antwort im RDF-Format von OD nimmt der SPAQRLer

entgegen, parst sie und speichert sie in einem Übergangsformat. Anschließend wird von jedem Eintrag die Hashsumme ermittelt, um diese mit den Hashsummen der alten Daten zu vergleichen. Dieser Vergleich wird für jeden Block separat durchgeführt. Unterscheiden sich die Hashsummen des alten und neuen Blocks nicht, so werden für diesen keine weiteren Aktionen ausgeführt. Treten jedoch Inkonsistenzen auf, so wird der entsprechende Block gelöscht. Daraufhin werden die Kalender der, zum Block gehörenden, Lessons, alle entsprechenden Einträge in der Datenstruktur AbonnierteKalender und anschließend die Lessons selbst gelöscht. Nach dem Entfernen der Einträge wird ein neuer Block mit den, aus OD erhaltenen, Informationen erzeugt. Anschließend wird für jede Lehrveranstaltung des Blocks ein neuer Kalender generiert und die entsprechenden Lessons werden erzeugt.

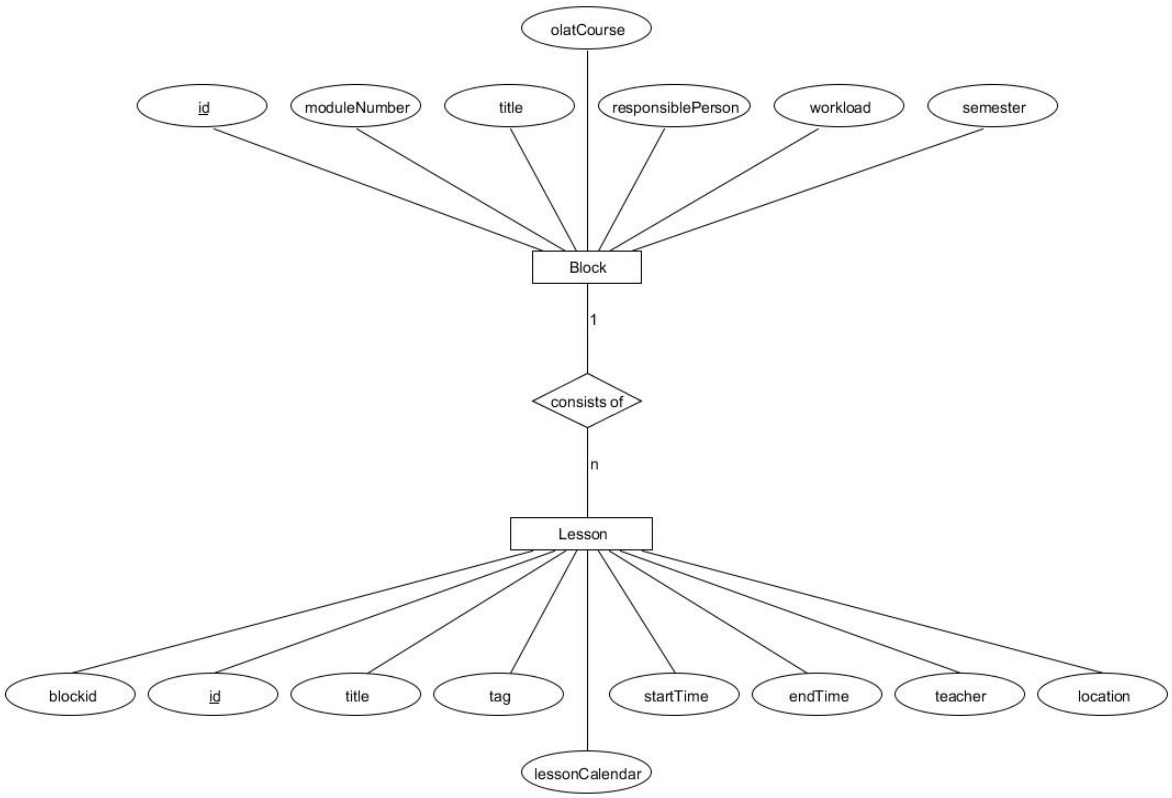


Sequenzdiagramm zum Vorgang des Abonnierens einer Veranstaltung

Dieses Sequenzdiagramm stellt den Vorgang der Suche und des Abonnierens einer Lehrveranstaltung dar. Auch hier wurden aus Gründen besserer Lesbarkeit nur die wichtigsten Classifier dargestellt und auf Nennung einzelner Methodenaufrufe verzichtet. Der Classifier DatenBank entspricht keiner Klasse, sondern repräsentiert nur die Datenbankschnittstelle.

Möchte ein Nutzer eine Lehrveranstaltung seinem Kalender hinzufügen, so lässt er sich zunächst in einem Tab eine Liste aller Lehrveranstaltungen anzeigen. Diese Liste kann er durch Auswahl des Studiengangs einschränken. Diese Suche wird mit der Klasse StundenplanForm durchgeführt, welche eine entsprechende Anfrage an die Datenbank richtet und anschließend die Liste (als Tabelle realisiert) aktualisiert.
Hat der Nutzer nun eine Lehrveranstaltung ausgewählt, so kann er diese durch Betätigung des Abonnieren-Buttons seinem Kalender hinzufügen lassen. Die Betätigung des Buttons teilt der TabTableController dem BlockLesson_Generator mit, welcher den Nutzer und den entsprechenden Lesson-Kalender in die Tabelle AbonnierteKalender einträgt. Nach Aktualisierung des Tabs, wird dem Nutzer angezeigt, dass er eine Lehrveranstaltung abonniert hat.

4.2 Datenstruktur der Veranstaltungen



ER-Diagramm

Einzelne Lehrveranstaltungen sind oft in Lehrveranstaltungsblöcke zusammengefasst. Daher werden die beiden Datenstrukturen Block und Lesson für die Speicherung verwendet. Informationen innerhalb der Objekte werden aus der OD übernommen, bzw. daraus abgeleitet.

Ein Objekt der Klasse Block besitzt die Attribute id (URI des Blockes aus der OD-Datenquelle), moduleNumber (Modulnummer), title (Titel des Blocks), responsiblePerson (Verantwortlicher), workload (aufzuwendende Zeit), olatCourse (ID eines OLAT Kurses zu dem verlinkt wird) und semester (empfohlene Semester). Das Attribut id identifiziert einen Block eindeutig.
Das Feld olatCourse kann null sein, wenn es keinen entsprechenden Kurs gibt (Verlinkung zum OLAT-Kurs wurde nicht realisiert).

Ein Objekt der Klasse Lesson besitzt die Attribute blockid (ID des übergeordneten Blocks), id (URI der Lehrveranstaltung), title (Titel der Lehrveranstaltung), tag (Tag der LV), startTime (zeitl. Anfang der LV), endTime (zeitl. Ende der LV), teacher (Dozent), location (Ort) und lessonCalender (ID des Kalenders).
Ein Lesson-Objekt wird durch das Attribut id eindeutig identifiziert.

Da ein Veranstaltungsblock aus einem oder mehr Veranstaltungen besteht, können einem Block-Objekt mehrere Lessons zugeordnet werden. Dagegen ist eine Lesson immer nur genau einem Block zugewiesen.

Die Datenstrukturen werden in der OLAT-internen Datenbank abgelegt und dort verwaltet. Da Hibernate für die Speicherung und das Lesen der Daten verwendet wird, müssen die Datenstrukturen vorher in Hibernate konfiguriert werden. Hierzu werden die Dateien Block.hbm.xml, Lesson.hbm.xml, Recommended.hbm.xml und AbonnierteKalender.hbm.xml eingesetzt.

4.3 SPARQL und OD-Datenquelle

Der SPARQLer stellt SPARQL-Anfragen an OD (od.fmi.uni-leipzig.de) und erhält ein XML-Dokument. Daraufhin werden die XML-Daten mit der in OLAT implementierten Bibliothek geparst und in ein Übergangsformat (OD_block für Lehrveranstaltungsblöcke und OD_lv für Lehrveranstaltungen) umgewandelt. Dieses wird an den BlockLesson_Generator weitergegeben, welcher die Daten auf konkrete Objekte (Block und Lesson) abbildet.

Die Klasse ist so aufgebaut, dass bezüglich Änderungen in der OD, nur Anpassungen innerhalb der SPARQLer-Klasse vorgenommen werden müssen.