Software Design Document

Mathieu Reymond, Arno Moonens

Mei 2015

Inhoudsopgave

1 Versiegeschiedenis

Versie	Commentaar
1	Design van eerste iteratie
2	Design van tweede iteratie
3	Design van derde iteratie
4	Design van vierde iteratie

2 Definities

Table 1: Definities

API	Application Programming Interface
REST	Representational state transfer
JSON	JavaScript Object Notation
SDD	Software Design Description
SQL	Structured Query Language

3 Introductie

3.1 Doel en Scope

Dit document heeft als doel om de software-architectuur van de WiseLib-applicatie uit te leggen aan de hand van een veralgemeende beschrijving van het systeem.

Dit document zal geraadpleegd worden door de testers en door programmeurs tijdens het implementeren van de applicatie.

WiseLib wordt op een iteratieve manier gemaakt. Dit betekent dat we in de eerste versie beginnen met slechts een beperkt aantal functionaliteiten te implementeren. In volgende iteraties wordt onze applicatie dan telkens uitgebreid en worden de functionaliteiten verbeterd en worden er nieuwe toegevoegd.

Dit document volgt de IEEE Std 1016-2006^{TM}"Standard for information technology — systems design — software design descriptions" standaard.

4 Logica

De Wiselib applicatie is onderverdeeld in twee grote onderdelen: de backend (4.1) en de frontend (4.3). De backend beheert de data van het systeem door gebruik te maken van een database en implementeert de core functionaliteiten van het programma. De frontend toont de data aan de gebruiker, en laat toe om onrechtstreeks met deze data te interageren. De communicatie tussen deze twee onderdelen gebeurt via een op voorhand afgesproken protocol: een RESTful[1] API[2] met als media type het JSON formaat.

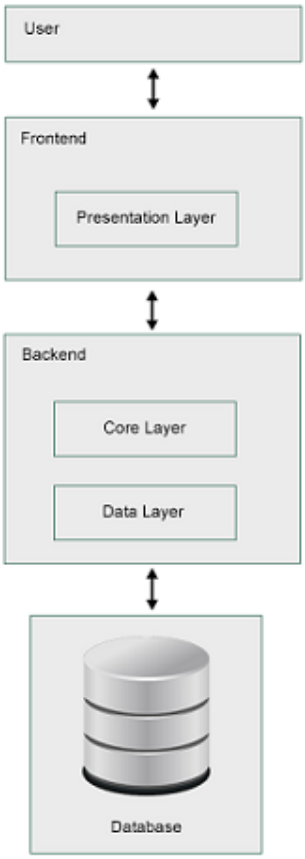

Figure 1: De verschillende layers van de applicatie

4.1 Backend

De backend bestaat uit drie grote modules:

- De database module (zie 4.1.1).
- De core module (zie 4.1.2).
- De communicatie module (zie 4.1.3).

De database module is verantwoordelijk voor de interactie tussen de applicatie en de database. Het biedt een abstracte klasse aan die de core-module kan gebruiken om met de database te communiceren.

De core module implementeert de functionaliteit van het systeem. De data is hier door verschillende klassen voorgesteld, waarmee de logica van het systeem uitgevoerd kan worden.

De communicatie module interageert met de buitenwereld door middel van de RESTful[1] API[2]. Geldige requests worden behandeld door gebruik te maken van de core module. De klassen van de module voeren eventuele berekeningen uit, voordat het antwoord omgezet wordt naar een JSON formaat en teruggestuurd wordt.

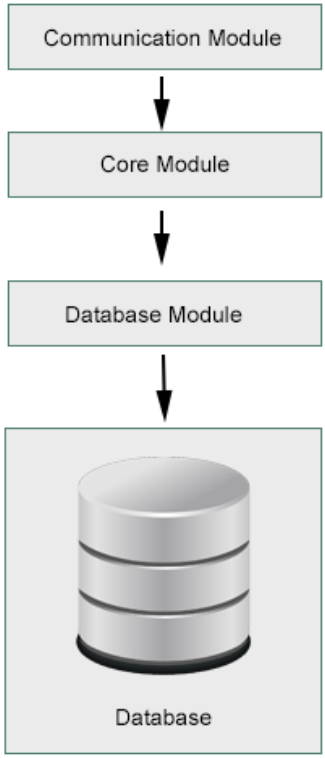

Figure 2: Het Module Schema

4.1.1 Database Module

De database module biedt twee abstracte klassen aan om met de database te interageren : objecten van de `Writeable` klasse kunnen opgeslagen, geupdate en verwijderd worden van de database, terwijl objecten van de `Searchable` klasse (een subklasse van `Writeable`) ook opgezocht kunnen worden in de database.

De `Writeable` voorziet twee methoden:

- save() : deze methode voegt de `Writeable` toe aan de database, of update de database, in het geval dat de instance een `id` heeft.
- destroy() : deze methode verwijderdt de `Writeable` van de database, vermits een `id` gedefinieerd is.

De `Searchable` heeft een variable `q`, waarmee in de database opgezocht kan worden (door middel van het SQL commando `LIKE`). Het is ook mogelijk om naar specifieke variabelen te zoeken door gebruik te maken van tags. Een tag begint met `'@'`, gevold door de naam van de variable. Bijvoorbeeld, `q = 'John@firstName@lastName'` gaat zoeken naar alle 'firstName's of 'lastName's die gelijkaardig zijn met 'John'. Deze klasse voorziet twee methoden:

- fetch() : deze methode update de `Searchable` met de velden die in de database gevonden werden, vermits een `id` meegegeven wordt.
- fetchAll() : deze methode geeft een `Array` terug met alle `Searchable`s die met de instance corresponderen.

Onderliggend maakt het database module gebruik van *Bookshelf* [3], een Object Relational Mapper, om de SQL queries op te bouwen en met de database te interageren. De `Linker` zorgt ervoor dat elke `Writeable` object zijn `Representation` heeft, die de verschillende attributen van het object mapt op zijn database velden. De `Representation` klasse gebruikt `format` om de core representatie om te zetten naar de database representatie, en `parse` voor het omgekeerde proces. De `toQuery` methode bouwt een *bookshelf* query op, die de `DBManager` dan kan gebruiken om met de database te interageren, door gebruik te maken van *bookshelf* methoden `save`, `fetch` en `destroy`. Omdat al deze operaties asynchroon gebeuren, worden *Promises*[4] gebruikt door de `DBManager`.

De `Writeable` en `Searchable` klassen maken intern gebruik van de `DBManager` en zijn vier methodes, die overeenkomen met de REST architectuur:

- get(searchable, repr): Deze methode zoekt in de database.
  - `searchable`: de methode zoekt naar data die overeenkomt met de variabelen die in searchable gegeven worden.
  - `repr`: Initieel wordt de `Linker` representatie van het searchable object gebruikt, maar het is mogelijk om een custom representatie (repr) mee te geven als parameter. Deze gaat de correcte SQL queries opstellen die dan in deze methode uitgevoerd zullen worden.

Deze geeft een `Promise` terug met een lijst van alle corresponderende `Searchable`s.

- post(writeable, repr): Deze methode voegt nieuwe elementen (rijen, tabellen,... indien nodig) toe aan de database.
  - `writeable`: Een writeable object die toegevoegd wordt aan de database.
  - `repr`: Initieel wordt de `Linker` representatie van het writeable object gebruikt, maar het is mogelijk om een custom specifiekere (repr) mee te geven als parameter. Deze gaat de correcte SQL queries opstellen die dan in deze methode uitgevoerd zullen worden.

  Deze geeft een `Promise` terug met de aangepaste writeable.
- put(writeable, repr): Deze methode update bestaande elementen.
  - `writeable`: Een writeable object. De methode vervangt de velden van de database met diegene die in het object aanwezig zijn.
  - `repr`: Initieel wordt de `Linker` representatie van het writeable object gebruikt, maar het is mogelijk om een specifiekere representatie (repr) mee te geven als parameter. Deze gaat de correcte SQL queries opstellen die dan in deze methode uitgevoerd zullen worden.

  Deze geeft een `Promise` terug met de aangepaste writeable.
- delete(writeable, repr): Deze methode verwijdert bestaande elementen.
  - `writeable`: Een writeable object. De methode verwijdert het object uit de database door gebruik te maken van het `id` veld.
  - `repr`: Initieel wordt de `Linker` representatie van het writeable object gebruikt, maar het is mogelijk om een specifiekere representatie (repr) mee te geven als parameter. Deze gaat de correcte SQL queries opstellen die dan in deze methode uitgevoerd zullen worden.

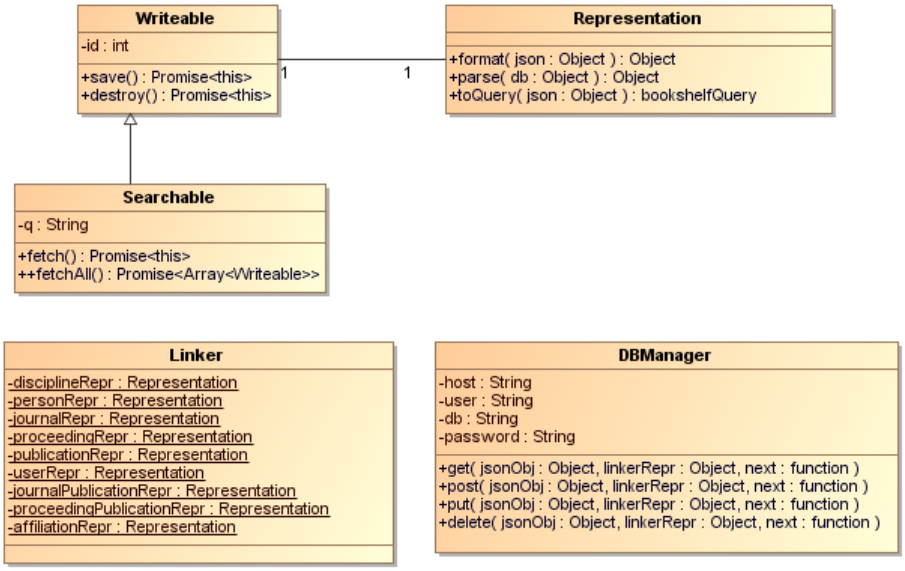

Figure 3: Het Database Klasse diagram

4.1.2 Core Module

De core-module bestaat uit verschillende klassen, die elk subklassen zijn van `Searchable` (aangeboden door de database module). Gebruikers zijn voorgesteld door de klasse `User`. De applicatie maakt een onderscheid tussen een gebruiker en een persoon `Person`. Het is namelijk mogelijk dat een auteur in de gegevensbank opgeslagen is, maar niet als gebruiker is ingeschreven. Dit kan bijvoorbeeld gebeuren wanneer een gebruiker een publicatie met co-auteurs uploadt. Publicaties zijn voorgesteld door de klasse `Publication`. Er wordt een onderscheid gemaakt tussen twee type publicaties: `JournalPublication` en `ProceedingPublication`. Dit onderscheid is noodzakelijk omdat deze publicaties verschillende karakteristieken hebben. `ProceedingPublications` hebben bijvoorbeeld publishers en editors, wat niet het geval is bij `JournalPublications`.

Omdat de rank van een journal of een proceeding een invloed heeft op de rank van een publicatie worden zij ook door hun eigen klasse gerepresenteerd (respectievelijk `Journal` en `Proceeding`).

Journals en proceedings zijn gespecialiseerd in specifieke onderwerpen, voorgesteld door de `Discipline` klasse. Deze kunnen onderdeel zijn van andere onderwerpen, `Disciplines` hebben dus een boom-structuur. Gelijkaardig hebben de affiliaties (`Affiliation` klasse) van auteurs ook een boom-structuur, die structuur is door de klasse `Tree` voorgesteld.

Elke klasse die een rank heeft, namelijk `Journal`, `Proceeding`, `Person` en `Publication` en zijn subklassen zijn allemaal de subklasse van `Rankable`, die de methode `calculateRank()` aanbiedt.

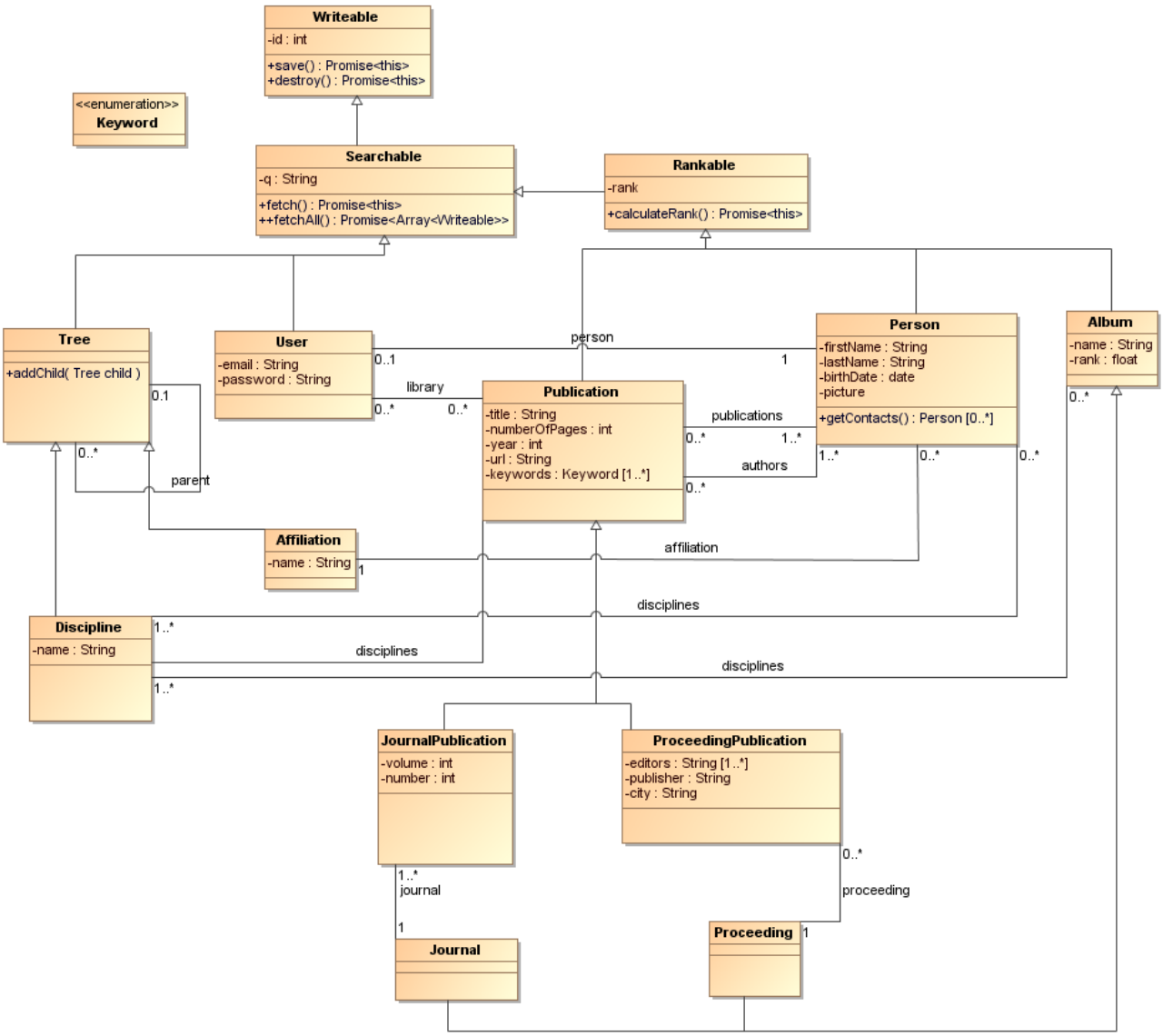

Figure 4: Het Core Klasse diagram

4.1.3 Communicatie Module

De communicatie module handelt requests af van de buitenwereld. De requests zijn gedefineerd door de API[2]. Elk URL heeft een `Route`, die de `get`, `post` `put`, `delete` HTTP requests afhandelt via zijn gelijk genoemde methode.

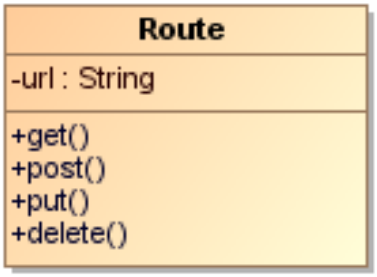

Figure 5: Het Communicatie Klasse diagram

4.2 Database

Het figuur zes toont het database diagram.
De tabellen 'publication', 'person' en 'user' zijn de belangrijkste onderdelen van de tabel.
Een publicatie kan slechts door één 'user' geüpload zijn, maar kan door meerdere personen geschreven en bewerkt zijn. Publicaties kunnen ook naar meerdere andere publicaties refereren en kunnen meerdere sleutelwoorden bevatten.

Het type van een publicatie ('proceeding' of 'journal' publicatie) wordt bepaald door de overeenkomstig genoemde tabellen. Deze bevatten steeds het id van de publicatie in de 'publication' tabel, samen met nog extra informatie die kenmerkend is voor dat soort publicatie.

Een 'proceeding' publicatie is steeds gelinkt aan een 'conference', die op zijn beurt eventueel gelinkt is aan academische disciplines.
Een 'journal' publicatie is gelinkt aan een 'journal', en kan ook gelinkt worden aan academische disciplines.
Een persoon maakt deel uit van een affiliatie en doet onderzoek omtrent academische disciplines.

Academische disciplines kunnen aan andere disciplines gelinkt worden door middel van de 'part_of' relatie. Dit betekent dus dat de ene discipline een onderdeel is van de andere. Dit is ook het geval bij affiliaties.

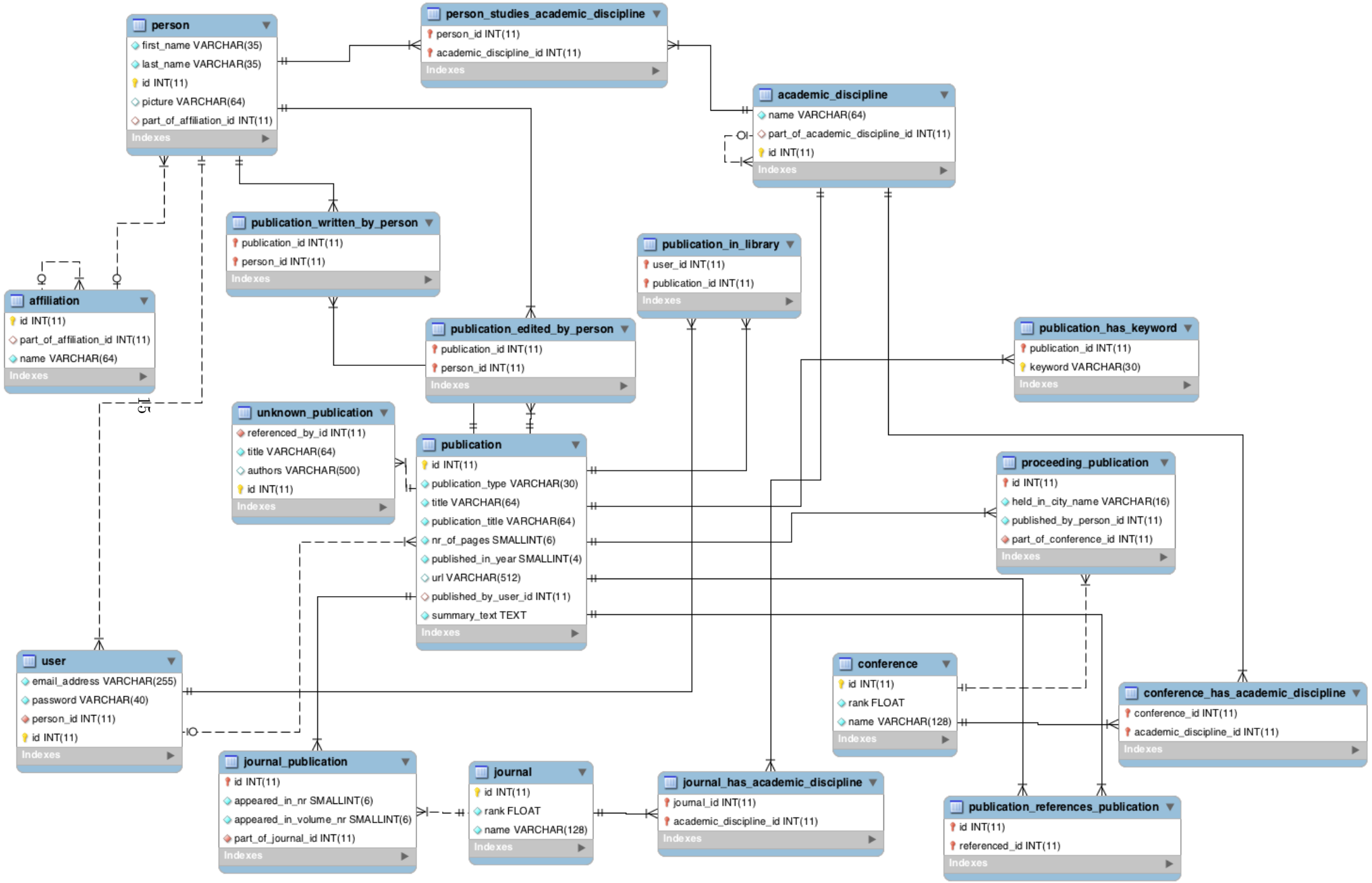

Figure 6: Het EER diagram

4.3 Frontend

Voor de frontend hebben we gebruik gemaakt van *AngularJS* en *Angular-Material*. De frontend zorgt voor interactie met de gebruiker en bestaat grofweg uit 3 delen: *models*, *views* en *controllers*. Zoals te zien in figuur zeven, zorgen *controllers* voor de verbinding tussen een *view* en een *model*.
Een *view* bevat html-pagina's en css code. Een *model* (bij *AngularJS* zogenaamde *services* en *factories*) voert bepaalde *bussiness*-logica uit die te complex of uitgebreid is om in een *controller* te definiëren.
De data die meegegeven wordt aan de presentatielaag en ontvangen wordt van de *server* is telkens in *JSON* formaat.
Als een gebruiker een actie uitvoert op een pagina (een *view*) wordt dit doorgegeven aan de bijhorende controller. Deze gaat dan een functie uitvoeren van een bepaald *model*. Dit *model* kan indien nodig communiceren met de *server*. De controller gaat de *view* dan aanpassen met het resultaat van het *model*.
Naast gewone *views* (om een bepaalde pagina te tonen), zijn er ook *directives* die op verschillende pagina's kunnen getoond worden. Dit zijn een soort templates waarbij bijvoorbeeld info over een bepaalde persoon of publicatie kan weergeven zonder elke keer de expliciet te formuleren wat er moet worden weergegeven.

De eerste *controller* die uitgevoerd wordt als de gebruiker de applicatie bezoekt is de *routeProvider*. Deze gaat de juiste *view* (*html*-pagina) voor de presentatielaag laden, samen met de bijpassende *controller*.

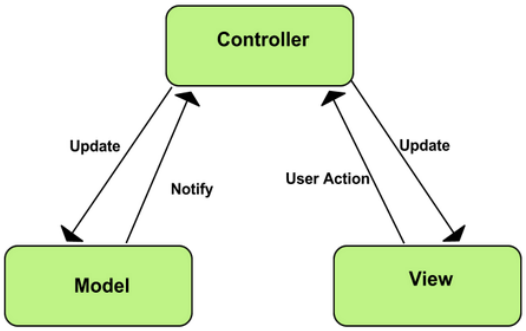

Figure 7: Het *model-view-controller* schema

We beschrijven nu elke pagina apart.

4.3.1 Startpagina en menu

De startpagina zelf toont slechts een welkomsttekst.
Het menu past zich aan naargelang je ingelogd bent of niet. Als je niet ingelogd bent zie je knoppen om je in te loggen, te registreren, en om publicaties op te

zoeken en te bekijken.
Als je wel ingelogd bent zie je naast de reeds genoemde knoppen ook een knop om je uit te loggen, een knop om je accountgegevens te wijzigen, je profiel te bekijken, je publicaties te zien, je bibliotheek te bekijken, je geüploade publicaties te bekijken en je sociale netwerk te bekijken. De knop om je profiel te bekijken verwijst slechts naar een 'Persoon' pagina die ook via andere pagina's toegankelijk is en waarvoor men niet moet ingelogd zijn om ze te bekijken.

Onderaan het menu bevinden zich 2 knoppen, 1 om de taal op de website in het Nederlands te zetten en 1 om de taal in het Engels te zetten. Het veranderen van de taal gebeurt onmiddelijk, zonder dat de pagina herladen moet worden. De laatst gekozen taal wordt onthouden als je de website opnieuw bezoekt.

4.3.2 'Log in' pagina

Hierbij wordt simpelweg het email adres en wachtwoord via de controller en de inlog service naar de server gestuurd en verwacht men een token of een error als antwoord. De token wordt dan in de huidige sessie opgeslagen. Hierna wordt de gebruiker naar de startpagina gestuurd en wordt er een bericht getoond dat men succesvol is ingelogd. Een error wordt rechtsboven op het scherm getoond in geval van een foutief emailadress of paswoord.

4.3.3 'Registreer' pagina

Allereerst wordt er naar een voor- en achternaam gevraagd. Als er voldoende karakters worden ingegeven wordt er naar personen met de ingegeven naam in de database gezocht. Indien er personen gevonden zijn worden deze in een lijst weergegeven en kan men er een van selecteren. Als men geen persoon geselecteerd heeft kan men ook een profiel afbeelding meegeven.
Vervolgens dient men een geldig email adres en paswoord (met minstens acht tekens) mee te geven.
Als er een plobleem was met het registreren krijg je net als bij een fout op de loginpagina een bericht te zien met een precieze foutmelding. Anders wordt de gebruiker naar de startpagina gestuurd en krijg je een bericht dat de registratie succesvol was.

4.3.4 'Upload publicatie' pagina

Deze pagina bevat een formulier om als ingelogde gebruiker een publicatie te kunnen uploaden. Allereerst kan je een *PDF* bestand uploaden met een knop om de titel, jaar van publicatie, aantal pagina's en url van de publicatie automatisch te laten invullen. Je kan deze velden echter ook manueel invullen. De samenvatting (*abstract*) kan niet uit een pdf gehaald worden en moet manueel worden ingevuld.
Hierna kan je co-auteurs meegeven. Deze worden net als een persoon bij het registreren opgezocht terwijl je typt. Je kan gevonden personen dan makkelijk

toevoegen of verwijderen door er op te klikken.
Daarna kiest men wat voor publicatie men wil uploaden: een *journal* publicatie of een *proceeding* publicatie.
Bij een *journal* publicatie geef je mee in welke *journal* de publicatie is verschenen en in welk volume en nummer precies.
Voor een *proceeding* publicatie geef je mee voor welke *proceeding* de publicatie is verschenen, door welke *editor* en *publisher* en in welke plaats de *proceeding* plaats vond.
Tenslotte kan je referenties toevoegen door het uploaden van een *BIBTEX* bestand.

4.3.5 'Update profiel' pagina

Op deze pagina kan je zowel je gebruikersgegevens (email en wachtwoord) als je persoonsgegevens (voor- en achternaam) aanpassen.
Voor elk deel van gegevens bevindt er zich een knop om de wijzigingen te bewaren. Deze worden pas aanklikbaar als je minstens 1 element hebt ingevuld en alle ingevulde elementen ook geldige informatie bevatten (bijvoorbeeld dat het email-veld wel degelijk een geldig email adres bevat).
Na het klikken op de knoppen krijg je een bericht of de wijzigingen al dan niet goed zijn uitgevoerd.
Men moet om deze pagina te bekijken ingelogd zijn en men krijgt dan ook een foutmelding als dit niet het geval is.

4.3.6 'Mijn publicaties' pagina

Hier staan alle publicaties die de gebruiker die de pagina bezoekt geüpload heeft.
Men moet dan ook ingelogd zijn om deze pagina te bekijken, anders krijgt men een foutmelding.
Voor elke publicatie is de titel, het aantal pagina's, het jaar van publicatie en het *abstract* te zien.
Rechts van de publicatie bevindt zich een knop om de publicatie te verwijderen. Als je hier op klikt verschijnt een venster om te bevestigen alvorens de publicatie daadwerkelijk verwijderd is.
Bij het klikken op de titel van een publicatie kom je op de 'publicatie' pagia van die publicatie terecht.

4.3.7 'Publicatie' pagina

Deze pagina is opgesplitst in 3 tabbladen.
Het eerste tabblad bevat algemene informatie over de publicatie. Hier kan men naast het abstract ook zien wie de publicatie heeft geüpload, de auteurs, de editeurs, het aantal pagina's, het jaar van publicering en de rank. Bij een persoon is de profiel afbeelding, de voornaam en de achternaam te zien. Bij het klikken op een persoon gaat men naar de 'persoon' pagina van die persoon.
Onderaan dit tabblad is er een knop om de publicatie toe te voegen aan de

bibliotheek van de gebruiker waar men mee ingelogd is. Als men niet ingelogd is of als deze publicatie reeds in de bibliotheek aanwezig is, is deze knop niet aanwezig.
Op het 2e tabblad kan men de publicatie bekijken.
Op het laatste tabblad ziet men publicaties waar deze publicatie naar verwijst. Deze ziet er hetzelfde uit als de 'Mijn publicaties' pagina, zonder de knop ernaast om de publicatie te verwijderen.

4.3.8 'Persoon' pagina

Naast de profielafbeelding en de voor- en achternaam bevat deze pagina 3 tabbladen.
De eerste bevat de affiliatie waar de persoon aan verbonden is, de academische disciplines waar de persoon onderzoek over doet en de rank van de persoon.
Het tweede tabblad bevat de publicaties die de persoon geschreven heeft. Deze lijst ziet er hetzelfde uit als de 'mijn publicaties' pagina, ook zonder de knop ernaast om de publicatie te verwijderen.
Het derde en laatste tabblad bevat een lijst met contactpersonen van de persoon. Deze zijn personen die aan dezelfde affiliatie verbonden zijn of personen die samen met de persoon waarvan je de pagina bekijkt een publicatie geschreven hebben.

4.3.9 'Zoek een publicatie' pagina

Hier kan men een publicatie zoeken uit de database of via het internet.
Allereerst moet men een sleutelwoord invoeren. Daarna kan men kiezen of men wil zoeken op titel, auteur, journal en/of conference. Tenslotte kan men met de 2 knoppen kiezen of men publicaties wil zoeken in de database of op het internet.
Voor het zoeken in de database moet op zijn minst een sleutelwoord zijn ingegeven. Voor het zoeken op het internet moet men naast het sleutelwoord ook kiezen op wat men wil zoeken.
Na het klikken worden er resultaten opgezocht. Indien dit niet gelukt is wordt er een foutmelding weergegeven. Anders is er voor elke gevonden publicatie de titel te zien, de naam van de uploader, de auteurs, het aantal pagina's, het jaar van publicatie en het abstract.

4.3.10 'Mijn bibliotheek' pagina

Deze pagina bevat publicaties die toegevoegd zijn (via de 'publicatie' pagina) aan de bibliotheek van de gebruiker die ingelogd is. Als men niet ingelogd is krijgt men dan ook een foutmelding.
Als men op de titel van een publicatie klikt komt men op de 'publicatie' pagina van die publicatie terecht.
Rechts van elke publicatie is er een verwijder-knop te zien om de publicatie (na het bevestigen) uit de bibliotheek van de gebruiker te verwijderen.

4.3.11 'Netwerk' pagina

Op deze pagina is een graf te zien van het netwerk van een persoon. Deze graf bestaat uit volgende *nodes*:

- Publicaties van de persoon
- Co-auteurs van deze publicaties
- De affiliatie van de persoon
- Personen die aan diezelfde affiliatie verbonden zijn

Publicaties worden voorgesteld door een rode ellips, personen door een donkergrijze driehoek en affiliaties door een paarse rechthoek. Verbindingen tussen publicaties/affiliaties en personen worden weergegeven door een dubbele pijl.
De graf zelf past zich aan aan de grootte van de pagina. Het is ook mogelijk om in en uit te zoomen.
Als men klikt op een node krijgt men meer info te zien die afhangt van het type. Bij een affiliatie krijgt men slechts de naam te zien, bij een persoon krijgt men naast de naam ook de profielafbeelding te zien en bij een publicatie kan men de titel zien en de abstract van de publicatie. Bij het klikken op de titel van de publicatie of de naam van de persoon komt men respectievelijk op de publicatie pagina of persoon pagina terecht.

5 Referenties

References

[1] *RESTful* http://en.wikipedia.org/wiki/Representational_state_transfer

[2] *Wiselib API* http://wilma.vub.ac.be/~se2_1415/api.html

[3] *Bookshelf* http://bookshelfjs.org/

[4] *Bluebird* https://github.com/petkaantonov/bluebird/blob/master/API.md