SDD

Mathieu Reymond, Arno Moonens

November 2014

Inhoudsopgave

1 Versiegeschiedenis **2**

2 Introductie **3**
2.1 Doel en Scope 3

3 Referenties **4**

4 Design Viewpoints **5**
4.1 Use Cases 5
4.2 Logica 9

5 Conventies **11**
5.1 Taalregels 11
5.1.1 Variabelen 11
5.1.2 Statements 11
5.1.3 Methoden en eigenschap definities 11
5.1.4 Multiline string literals 11
5.2 Stijlregels 12
5.2.1 Benamingen 12
5.2.2 Code formattering 12
5.3 Databaseregels 13
5.3.1 Benamingen 13
5.3.2 Key velden 13

1 Versiegeschiedenis

Versie	Commentaar
1	Design van eerste iteratie

2 Introductie

2.1 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, die gebruik zullen maken van de use cases (zie 4.1), en door programmeurs, voor 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 er functionaliteiten verbeterd en worden er nieuwe toegevoegd.

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

3 Referenties

References

[1] *Crockords code conventions* `http://javascript.crockford.com/code.html`

[2] *Google Javascript Style Guide* `http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml`

4 Design Viewpoints

4.1 Use Cases

Een gebruiker kan het systeem anoniem of ingelogd gebruiken. De verschillende functionaliteiten die door deze gebruikt kunnen worden, worden beschreven door verschillende use cases en in het use case diagram samengevat.

Naam	Account aanmaken.
Samenvatting	De gebruiker maakt een nieuw account aan.
Actoren	Anonieme gebruiker.
Aannamen	
Beschrijving	De anonieme gebruiker vult zijn gegevens in (e-mail, paswoord, voornaam, achternaam, hiërarchische affiliatie en onderzoeksdomein). Het systeem kijkt na of er al een account met het ingegeven e-mailadres bestaat. Als het account al bestaat, treedt er een uitzondering op. Daarna kijkt het systeem na of er personen bestaan met de ingegeven naam en voornaam (dit is mogelijk indien een vorige gebruiker de naam en voornaam van een persoon die nog geen account heeft, in het systeem heeft ingegeven als coauteur van zijn paper). In dit geval moet de gebruiker zichzelf kiezen uit de gevonden lijst (als hij er tussen zit). Uiteindelijk maakt het systeem een nieuw account aan.
Uitzonderingen	
bestaand e-mailadres	Het systeem meldt aan dat het e-mailadres al in gebruik is. De gebruiker wordt gevraagd om een ander e-mailadres in te vullen.
Resultaat	De gebruiker heeft een account.

Naam	Inloggen.
Samenvatting	De gebruiker logt zich in.
Actoren	Anonieme gebruiker.
Aannamen	
Beschrijving	De anonieme gebruiker vult zijn paswoord en e-mailadres in. Het systeem gaat kijken of er al een account met het ingegeven e-mailadres bestaat. Als dit niet het geval is, treedt er een uitzondering op. Het systeem gaat daarna nakijken of het paswoord van het gevonden account correct is. Zo niet treedt er een uitzondering op. Uiteindelijk wordt de anonieme gebruiker ingelogd.

Uitzonderingen

verkeerd e-mail Het systeem vermeldt dat er geen account met het ingegeven e-mailadres bestaat. De gebruiker wordt gevraagd om zijn e-mailadres opnieuw in te vullen.

verkeerd paswoord Het systeem vermeldt dat het paswoord voor het gevonden account verkeerd is. De gebruiker wordt gevraagd om zijn paswoord opnieuw in te vullen.

Resultaat De gebruiker is ingelogd.

Naam	Publicatie uploaden.
Samenvatting	De gebruiker zet zijn publicatie online.
Actoren	Ingelogde gebruiker.
Aannamen	
Beschrijving	De gebruiker wordt gevraagd om ofwel een pdf bestand, ofwel een bibtex bestand te uploaden. De publicatie-velden worden door het systeem ingevuld indien mogelijk. De gebruiker kan deze zelf nog aanpassen of aanvullen. Als de gebruiker geen file heeft geüpload, wordt hij gevraagd om manueel alle velden in te vullen. Daarna wordt de publicatie door het systeem toegevoegd. Als niet alle velden ingevuld zijn treedt er een uitzondering op. Het systeem voegt de publicatie toe aan de gebruikers eigen publicatielijst.
Uitzonderingen	

lege velden Het systeem toont de niet-ingevulde velden aan, en vraagt de gebruiker om die in te vullen.

Resultaat De gebruiker heeft zijn publicatie geüpload en deze staat in zijn publicatielijst.

Naam	Publicaties opzoeken.
Samenvatting	De gebruiker zoekt publicaties op volgens bepaalde criteria.
Actoren	Gebruiker.
Aannamen	
Beschrijving	De gebruiker wordt gevraagd om keywords, disciplines en/of auteurs in te vullen in het zoeksysteem. Het systeem geeft dan de meest relevante publicaties terug.
Uitzonderingen	
Resultaat	De meest relevante publicaties worden aan de gebruiker getoond.

Naam Publicatie toevoegen aan gebruiker zijn bibliotheek.

Samenvatting	De gebruiker moet een publicatie die hij gevonden heeft met het systeem kunnen toevoegen aan zijn eigen bibliotheek.
Actoren	Ingelogde gebruiker.
Aannamen	
Beschrijving	De gebruiker zoekt naar publicaties (zie de *use case* "Publicaties opzoeken"). De gebruiker klikt op een gevonden publicatie. Een ingelogde gebruiker krijgt een knop te zien om de publicatie toe te voegen aan zijn bibliotheek. De gebruiker klikt hier op. Hierna verschijnt een kopie van de betreffende publicatie in de bibliotheek van de gebruiker.
Uitzonderingen	
Resultaat	De gekozen publicatie bevindt zich in de bibliotheek van de gebruiker.

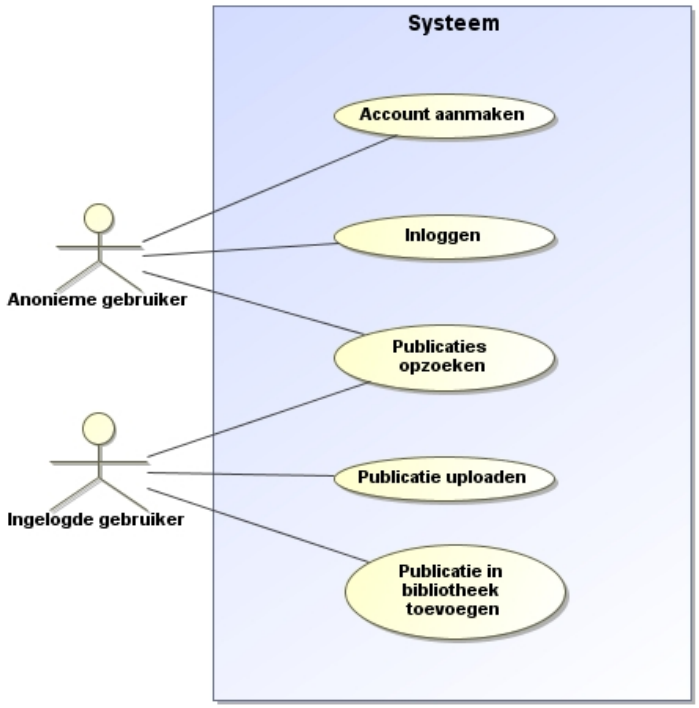

Figure 1: Use Case Diagram

4.2 Logica

Het Wiselib systeem is onderverdeeld in verschillende klassen. Anonieme gebruikers zijn voorgesteld door de klasse **User**. Er is nood om de anonieme gebruiker voor te stellen omdat die toegang heeft tot basisfunctionaliteiten zoals het opzoeken van publicaties. Eenmaal een gebruiker ingelogd is wordt hij door de klasse **RegisteredUser** voorgesteld. 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 ingeschreven is. 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 typen publicaties : **JournalPublication** en **ProceedingPublication**. Dit onderscheid is noodzakelijk omdat deze publicaties verschillende karakteristieken hebben. **ProceedingPublication**s hebben bijvoorbeeld publishers en editors, wat niet het geval is bij **JournalPublication**s.

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

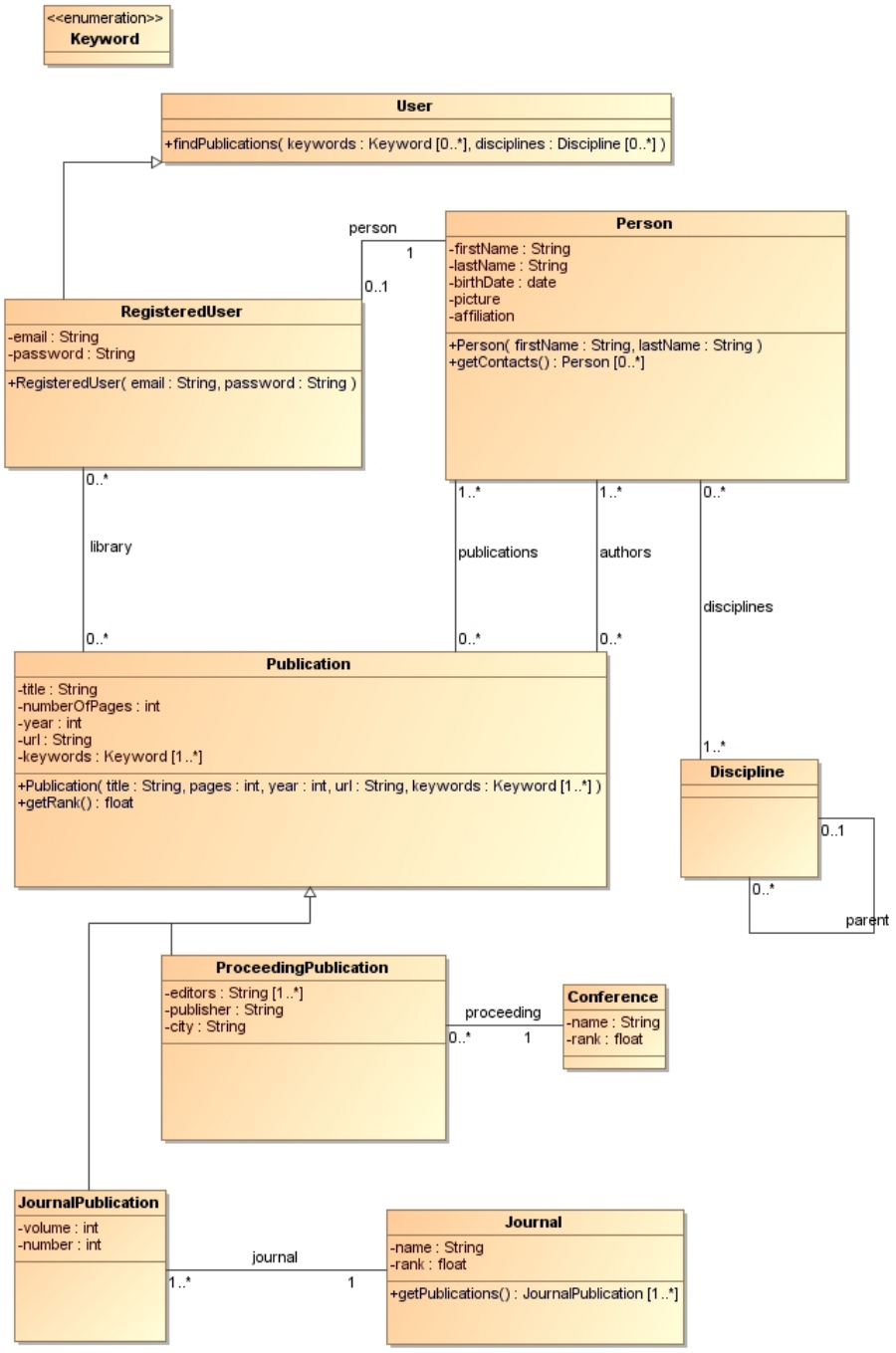

Figure 2: Klasse Diagram

5 Conventies

Om overzichtelijke, uniforme en duidelijke code te hebben is het nodig om een set van conventies te volgen. Deze conventies zijn gebaseerd op Crockords code conventions[1] en de Google Javascript Style Guide[2].

5.1 Taalregels

5.1.1 Variabelen

Alle variables worden door middel van "var" gedeclareerd : zonder een "var" worden deze in de globale omgeving geplaatst.

5.1.2 Statements

- Elke lijn heeft maximum een simpele statement.
- ";" wordt altijd gebruikt om het einde van een statement aan te duiden.

5.1.3 Methoden en eigenschap definities

De verkozen stijl voor methodes is :

```
Foo.prototype.bar = function() {
    /* ... */
};
```

De verkozen stijl voor andere eigenschappen is om ze in de constructor te initialiseren :

```
/** @constructor */
function Foo() {
    this.bar = value;
}
```

5.1.4 Multiline string literals

Er worden geen strings over meerdere lijnen geschreven :

```
var myString = 'A rather long string of English text, an error message \
               actually that just keeps going and going';
```

In plaats daarvan worden ze geconcateneerd :

```
var myString = 'A rather long string of English text, an error message ' +
    'actually that just keeps going and going';
```

5.2 Stijlregels

5.2.1 Benamingen

Er wordt *camelCase* gebruikt voor het benoemen van variabelen, functies,... (behalve voor de naam van een bestand) :

- functionNamesLikeThis
- variableNamesLikeThis
- ClassNamesLikeThis
- EnumNamesLikeThis
- methodNamesLikeThis
- CONSTANT_VALUES_LIKE_THIS
- foo.namespaceNamesLikeThis.bar
- filenameslikethis .js

5.2.2 Code formattering

Indentatie Omdat "TAB" verschillende resultaten geeft afhankelijk van het gebruikte programma wordt deze niet gebruikt. Indentatie gebeurt met vier spaties.

Gekrulde haakjes De gekrulde haakjes worden op dezelfde lijn als wat ze openen gezet :

```
if (something) {
    // ...
} else {
    // ...
}
```

Spaties Er zijn geen spaties tussen linker en rechter haakjes :

```
var arr = [1, 2, 3];  // No space after [ or before ].
```

Na "," wordt er altijd een spatie gezet.

5.3 Databaseregels

5.3.1 Benamingen

Er wordt *snake_case* gebruikt voor het benoemen van tabellen, kolommen,... Ze worden ook steeds met kleine letters geschreven:

- account
- first_name

5.3.2 Key velden

Primary key Deze worden steeds "id" genoemd. Dit is kort, simpel en makkelijk te onthouden.

Foreign key De velden in een *foreign key* tabel moeten een combinatie zijn van de naam van de gerefereerde tabel en van de gerefereerde velden:

```
CREATE TABLE publication_written_by_person (
  publication_id        bigint NOT NULL REFERENCES publication(id),
  person_id      bigint NOT NULL REFERENCES person(id),
  CONSTRAINT publication_written_by_person_pkey PRIMARY KEY (publication_id,
  person_id));
```