Afstudeerscriptie ABI - Team 30

Bachelor Thesis ABI - Team 30

Datum: 29 mei 2015

Status: Definitief

Auteur	
Naam	Teun Theunissen, Joop van de Heijning & Alex Mekkering

In opdracht van	
Organisatie	Open Universiteit Nederland, faculteit Informatica
Cursus	T61327 - Afstudeerproject bachelor informatica
Begeleider	Rik Bos
Examinator	Marko van Eekelen
Opdrachtgever	Lloyd Rutledge

Inhoudsopgave

1. Samenvatting 3
2. Introductie 4
3. Requirements 9
4. Algemeen overzicht domeinen en technieken 11
5. Verslag onderzoek deeldomein en bijbehorende technieken (individueel) 18
6. Verslag analyse onderzoekscontext .....
7. Beschrijving van de opgeleverde softwareproducten .....
8. Procesverslag .....
9. Teamreflectie .....
10. Persoonlijke ervaringen en leermomenten .....
11. Conclusies en aanbevelingen .....

Referenties .....

Bijlagen .....

1. Samenvatting

Het Semantic Web staat voor open gegevensmodellen, interpreteerbaarheid door machine's van gegevens en deling van de gegevens op het web. Wiki's zijn gegevenssystemen waar gebruikers gegevens kunnen invoeren en verkennen. In de context van Rutledge's onderzoek naar Semantic Web en wiki's demonstreren wij een implementatie van Fresnel Forms, een tool die deze twee projecten samenbrengt. Met deze tool kan met een model-gedreven benadering eerst op basis van een semantisch gegevensmodel efficiënt een presentatiedefinitie gespecificeerd worden. Vervolgens kan met een mapping-functie hier automatisch een semantische wiki uit gegenereerd worden. In deze wiki kunnen via formulieren machine-interpreteerbare gegevens ingevuld en verkend worden en vervolgens gedeeld worden op het Semantic Web.

Fresnel Forms is een uitbreiding van MDD Fresnel, product van ABI-team 28, waar MDD Fresnel alleen de presentatiedefinitie genereerde. Fresnel Forms heeft naast de mapping functie naar wiki meer specificatiemogelijkheden toegevoegd. De plug-in is wat betreft de code en het bouwproces grondig refactored en kwalitatief verbeterd, wat het voor een eventueel volgend ABI team of anderen eenvoudig mogelijk moet maken om de ontwikkeling van Fresnel Forms voort te zetten.

De schaalbaarheid van Fresnel Forms heeft zich bewezen door het genereren van een wiki-user interface voor één van de grootste semantische gegevensmodellen ter wereld, namelijk de Wikipedia-Dbpedia-ontologie. Als *proof of concept* is te verwachten dat Fresnel Forms Rutledge's onderzoek op het gebied van automatisch gegenereerde user interfaces voor het Semantic Web zal bekrachtigen.

2. Introductie

Het Semantic Web[1] aan de ene kant biedt een kader voor het delen van gegevens. Wiki's zoals MediaWiki[2] aan de andere kant zijn platformen voor het invoeren en verkennen van gegevens. In deze scriptie leggen wij de implementatie van een tool (Fresnel Forms[3]) voor die laat zien hoe de twee projecten samengebracht kunnen worden door middel van een efficiënt te specificeren en genereren wiki-interface met functionaliteit voor en op basis van technieken van het Semantic Web.

Onderzoekscontext

Semantic Web. Het Semantic Web biedt een gemeenschappelijk kader waarmee gegevens worden gedeeld en hergebruikt over toepassings-, enterprise-, en gemeenschapsgrenzen heen (W3C, 2015). Het is geen apart web, maar een uitbreiding van het huidige, waarin gegevens op het web een door machine's te interpreteren betekenis worden gegeven door middel van tags (bv. `<span property="name">Tim Berners-Lee</span>`) op objecten zoals pagina's, tekst en data. Hierdoor kunnen computers en mensen op het web beter samenwerken.

Een programma dat informatie van twee verschillende bronnen met semantische tags wil vergelijken of combineren moet weten of dat deze twee termen in dezelfde betekenis

---

[1] `http://www.w3.org/2001/sw/`

[2] `http://www.mediawiki.org/wiki/MediaWiki`

[3] `http://is.cs.ou.nl/OWF/index.php5/Fresnel_Forms`

worden gebruikt. Een oplossing voor dit probleem wordt geleverd door gegevensmodellen, ofwel *ontologieën*. Een ontologie is een document waar in de code naar verwezen wordt dat formeel de relaties tussen termen definieert (Berners-Lee, Hendler, & Lassila, 2001). Op hoog niveau zijn ontologieën als alternatieven voor relationele database te beschouwen, voor toepassingen die schema's met een rijkere betekenis vereisen (Martinez-Cruz, Blanco, & Vila, 2012).

Kernpunten van het Semantic Web zijn dus ontologieën, interpreteerbaarheid van gegevens door machines en deling van de gegevens op het web.

Wiki's en het Semantic Web. Wikipedia[4] is een bekende online encyclopedie, DBpedia[5] extraheert gestructureerde informatie uit Wikipedia en publiceert deze op het web. Wiki's, waaronder Wikipedia, zijn gegevenssystemen waar gebruikers gegevens kunnen invoeren en verkennen. Wikipedia (draaiend op MediaWiki[6]) in combinatie met DBpedia heeft laten zien dat er op grote schaal gebruik gemaakt wordt van -en dus blijkbaar behoefte bestaat aan- gegevensinvoer en -verkenning voor en met behulp van technieken van het Semantic Web (Lehmann, et al., 2014). Wikipedia-DBpedia's workflow is echter omgekeerd aan die van een typisch gegevenssysteem. Gebruikers van Wikipedia beginnen met het handmatig intypen van gegevens als wiki-sjablooncode voor infoboxes[7] (fig. 1 H:3). DBpedia extraheert vervolgens gegevens uit deze infoboxen en vormt daarmee de DBpedia ontologie (inclusief instanties) (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013). Deze ontologie kan dan door machine's doorzocht worden en gedeeld worden op het Semantic Web.

De workflow van een typisch gegevenssysteem begint met een gegevensmodel of ontologie. Vervolgens wordt een user interface voor het invoeren en verkennen van de gegevens toegevoegd (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013). Rutledge doet onderzoek naar semantische gegevenssystemen gebaseerd op technieken van het Semantic Web (ontologieën, machine-leesbaarheid van gegevens, deling op het web) en bekijkt of het mogelijk is om de workflow voor Wikipedia (of een andere wiki) te laten beginnen met een ontologie waaruit dan een user interface gegeneerd wordt.

Semantic MediaWiki[8], een extensie van MediaWiki, ondersteunt het maken van systemen gebaseerd op technieken van het Semantic Web, zoals semantische tags en zoekopdrachten op gegevens. Ook kan het gegevens exporteren in een formaat voor het delen op het Semantic Web, RDF[9]. Semantic Forms[10], een extensie van Semantic MediaWiki, is een interface-driven tool waarmee tabellen en formulieren gemaakt kunnen worden, waarmee efficiënt gegevens ingevoerd en verkend kunnen worden (fig. 8 en 9).

---

[4] http://www.wikipedia.org

[5] http://dbpediawww.informatik.uni-leipzig.de/

[6] http://www.mediawiki.org/wiki/MediaWiki

[7] http://en.wikipedia.org/wiki/Help:Infobox

[8] https://semantic-mediawiki.org/

[9] http://www.w3.org/RDF/

[10] https://www.semantic-mediawiki.org/wiki/Semantic_Forms

OWL Wiki Forms en Fresnel. In 2013 introduceerde Rutledge OWL Wiki Forms[11] (OWF), een extensie van MediaWiki, Semantic MediaWiki en Semantic Forms. Gebruik makend van de functionaliteit van Semantic Forms genereert deze tool, op basis van gegeven ontologieën, semantische wiki's. Op deze manier wordt dus in feite met de workflow van een typisch gegevenssysteem een semantische wiki gecreëerd.

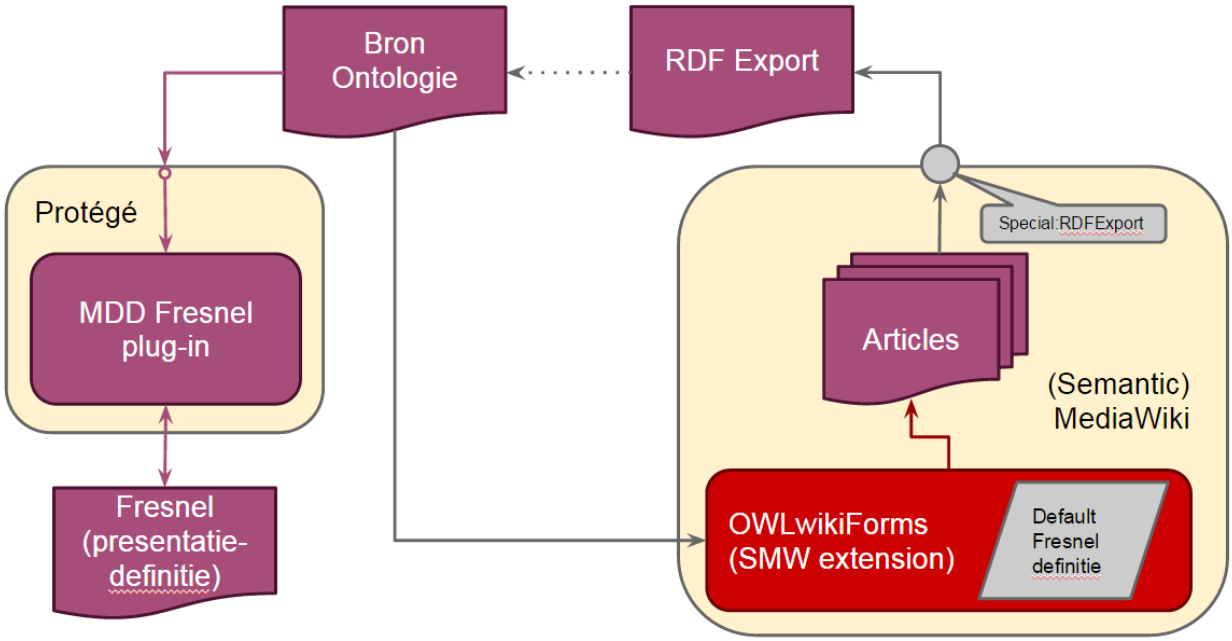
Het is belangrijk om de standaardpresentatie van user interfaces te kunnen aanpassen (Szekely, 1996). OWF maakt gebruik van een definitie in Fresnel[12] voor de standaardweergave van user interfaces.

In 2006 is Fresnel geïntroduceerd als ontologie voor het presenteren van gegevens uit Semantic Web ontologieën (Pietriga, Bizer, Karger, & Lee, 2006). Net zoals Cascading Style Sheets (CSS) bepalen hoe XML documenten moeten verschijnen op web-browsers, kan Fresnel gebruikt worden om trapsgewijs stijlinformatie aan informatieplatformen gebaseerd op semantische ontologieën toe te voegen (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013).

Na generatie van een wiki-interface met OWF kan door het handmatig toevoegen van Fresnel presentatiegegevens de interface trapsgewijs op maat gemaakt worden. De in 2014 gepresenteerde plug-in MDD Fresnel (Brenninkmeijer & Zwanenberg, 2014) geeft de mogelijkheid om met een GUI een presentatiedefinitie in Fresnel voor een ontologie te maken. Deze definitie zou bij de generatie van een wiki aan OWF meegegeven kunnen worden (Figuur 1).

Daar een Fresneldefinitie ook de relevante delen van het schema van de ontologie bevat kan het als een platformonafhankelijk model van een ontologie met toegevoegde presentatiegegevens in een Model Driven Development (MDD) proces beschouwd worden.

Model Driven Development. De voordelen van MDD zijn evident. Modelmatig ontwikkelen



Figuur 1. Situatie vóór vervanging MDD Fresnel door Fresnel Forms.

[11] `http://is.cs.ou.nl/OWF/index.php5/Main_Page`

[12] `http://www.w3.org/2005/04/fresnel-info/`

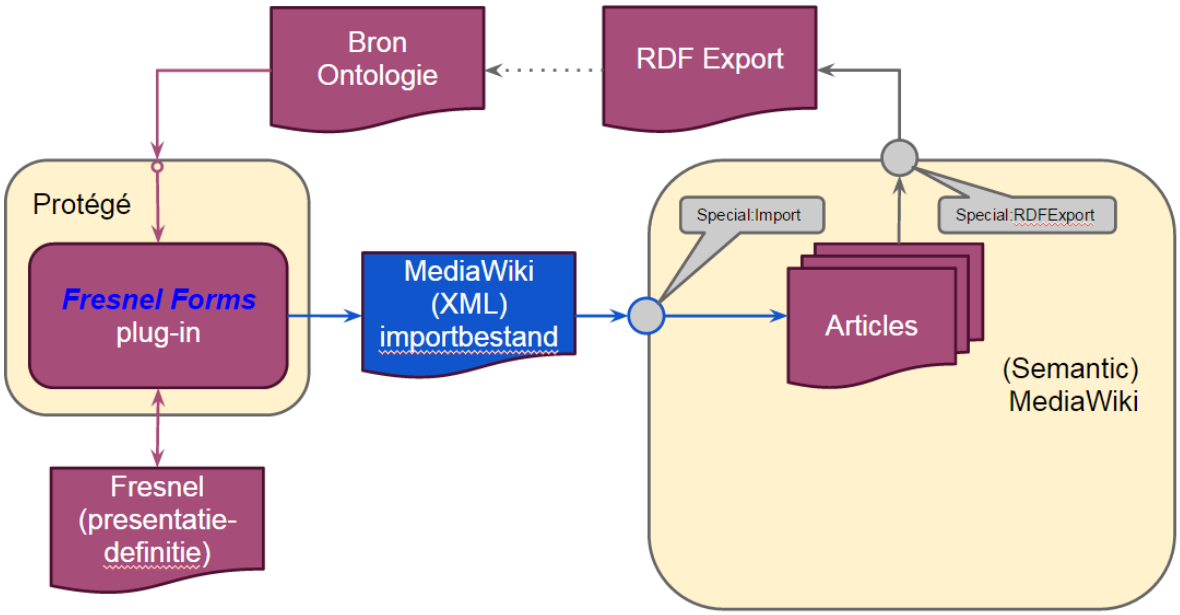
en het automatisch produceren van code leidt tot hogere productiviteit en betrouwbaarheid in het softwareproces (Selic, 2003). De concepten voor MDD van user interfaces worden steeds volwassener (Van den Berg, et al., 2010). Door een mapping functie van het platformonafhankelijke Fresnel-model naar platformspecifieke code voor een user interface kan MDD ingezet worden bij generatie van een user interface voor een gegevenssysteem gebaseerd op een ontologie.

Vraagstelling

Wikipedia in combinatie met DBpedia brengt het Semantic Web samen met het verkennen en invoeren van gegevens, maar de workflow van Wikipedia-DBpedia is het omgekeerde van de workflow van een typisch gegevenssysteem. Een typische gegevenssysteembenadering begint met een gegevensmodel of ontologie en voegt vervolgens een user interface toe voor het invoeren en verkennen van de gegevens (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013).

OWF genereert semantische wiki-interfaces op basis van ontologieën, maar is voor definitie van de user interface afhankelijk van apart bij te leveren presentatiegegevens in Fresnel. Ook is de tool als extensie van Semantic MediaWiki platformafhankelijk en net als MDD Fresnel direct afhankelijk van de bronontologie.

Rutledge heeft ABI-team 30 de opdracht gegeven MDD Fresnel uit te breiden met functionaliteit waarmee het platformonafhankelijk model in Fresnel efficiënt met presentatiegegevens aan te passen is en semantische user interface-code automatisch gegenereerd kan worden. Door een mapping functie van Fresnel naar wiki kan zo bekeken worden of het mogelijk is om Wikipedia's (of elke wiki's) workflow te laten beginnen met een ontologie waaruit een user interface gegeneerd wordt (Figuur 2).



Figuur 2. Situatie na vervanging MDD Fresnel door Fresnel Forms.

Mogelijke consequenties voor het onderzoek

Als *proof of concept* is te verwachten dat Fresnel Forms, de implementatie van bovenstaande vraagstelling, Rutledge's onderzoek op het gebied van automatisch gegenereerde user interfaces voor het Semantic Web zal bekrachtigen. Wikipedia-DBpedia heeft laten zien dat er op grote schaal gebruik gemaakt wordt van, en blijkbaar dus behoefte bestaat aan, Semantic Web browsing en gegevensinvoer (Lehmann, et al., 2014). Fresnel Forms kan laten zien hoe dit makkelijker kan met efficiënt te bouwen systemen en het geeft ook een platform voor het testen van de voorgestelde methode. Ook kan bewezen worden dat Fresnel goed te gebruiken en uit te breiden is voor deze doelstelling en wellicht maakt dat deze vocabulaire meer relevant.

De rest van de scriptie gaat dieper in op de requirements, bovenstaande context, op het eindproduct, het gevolgde proces bij implementatie en argumentering waarom dit een goed product is in de context.

3. Requirements

ABI-team 30 breidt MDD Fresnel (Brenninkmeijer & Zwanenberg, 2014) uit. MDD Fresnel slaat Semantic Web gegevens op in Fresnel. Deze Fresnelgegevens bepalen hoe een browserinterface voor een ontologie eruit zal zien (Figuur 3). Het algemene doel van dit project is het uitbreiden van deze plug-in met functionaliteit voor het exporteren van een wiki-interface gebaseerd op het Fresnel-model voor het verkennen en invoeren van gegevens (Figuur 22 en 23). Deze plug-in voor Protégé zal dan voortaan "Fresnel Forms" heten. Dit algemene doel is op te splitsen in globale requirements, waarvan de belangrijkste hieronder worden besproken. Deze zijn allen geïmplementeerd in Fresnel Forms.

http://dbpedia.org/ontology/#Tim_Berners-Lee

caption	Berners-Lee in 2012
name	Sir Tim Berners-Lee
depiction	http://commons.wikimedia.org/wiki/Special:FilePath/Tim_Berners-Lee_2012.jpg
homepage	http://www.w3.org/People/Berners-Lee/
occupation	http://dbpedia.org/ontology/#Computer_scientist

Figuur 4. Fresnelgegevens in Lena, een semantische webbrowser

Sir Tim Berners-Lee

OM, KBE, FRS, FREng, FRSA, DFBCS

Berners-Lee in 2014.

Born	Timothy John Berners-Lee 8 June 1955 (age 59) London, England
Occupation	Computer scientist
Employer	World Wide Web Consortium University of Southampton Plessey MIT
Title	Professor
Spouse(s)	Rosemary Leith
Parent(s)	Conway Berners-Lee Mary Lee Woods
Awards	See full list of honours

Website

www.w3.org/People/Berners-Lee

Figuur 3. Wikipedia infobox van Tim Berners-Lee. Noot. http://en.wikipedia.org/Tim_Berners-Lee opgeroepen op 13 mei 2015

Functionele requirements

1. **Mapping en export van Fresnel naar wikicode voor het verkennen en invoeren van gegevens.**

Na specificatie met de GUI van het platformonafhankelijke model in Fresnel (requirements 2-5 en 7), moet het MDD-proces voltooid kunnen worden met een export van het

model naar een MediaWiki-platform met de extensies Semantic MediaWiki en Semantic Forms. Hiervoor moet een “save wiki” knop op de GUI geplaatst worden. Fresnel Forms moet dan de benodigde categorie-, eigenschap-, sjabloon- en formulierpagina’s voor het beoogde informatiesysteem genereren. Deze pagina’s voor het verkennen en invoeren van gegevens met Semantic Forms moeten in de wiki geïmporteerd kunnen worden (fig. 6 en 7). OWF PHP kan als voorbeeld gebruikt worden.

2. **Fresnel Forms kan infoboxes genereren die lijken op Wikipedia infoboxes.** Fresnel Forms moet het mogelijk maken om automatisch een infobox in standaard Wikipedia-stijl te genereren. Daarnaast moet het mogelijk zijn om de infobox verder te specificeren met de GUI, zoals bijvoorbeeld het vet gedrukt presenteren van de titel. Hiervoor moet de infobox op Tim Berners-Lee’s pagina op Wikipedia (Figuur 4) als leidend voorbeeld spelen voor benodigde te implementeren functionaliteit. Na specificatie moet de resulterende infobox dan zo precies mogelijk lijken op het voorbeeld op Wikipedia. Hoe beter dit lukt, hoe overtuigender het *proof of concept* van de hypothese dat de benadering van Fresnel Forms, wat betreft het presenteren van gegevens, een alternatief kan vormen voor bijvoorbeeld Wikipedia-DBpedia.

3. **CSS van Fresnel naar wikicode.** Fresnel Forms moet naast Fresnelgegevens ook standaard wiki-interfaces genereren (requirement 1). Daarnaast, om bijvoorbeeld een infobox te stileren naar voorbeeld van een pagina op Wikipedia (requirement 2), is het nodig om de gebruiker de interface te laten specificeren met de GUI. Fresnel refereert voor bepaalde stileringsaspecten aan CSS (bijlage III), bijvoorbeeld fresnel:labelStyle “color:blue”;. Deze CSS-waarden, die met de GUI van MDD Fresnel (Brenninkmeijer & Zwanenberg, 2014) al ingevoerd kunnen worden, moeten in wikicode geplaatst worden om de wiki-interface te stileren.

4. **Waar nodig, breidt de Fresnelvocabulaire uit.** Requirements 1 en 2 genereren behoeftes aan stileringsaspecten waarvoor het niet altijd mogelijk is om de CSS-eigenschappen van Fresnel (requirement 3) te gebruiken. Zulke te implementeren aspecten zijn het zetten van een minimum of maximum van een in te voeren waarde voor een semantische property[13], het verbergen of veranderen van een label, het bepalen van het tussenvoegsel bij een lijst van meerdere waarden[14] en andere (bijlage XV). Vaak heeft Fresnel de benodigde vocabulaire in huis, zoals `fresnel:label “Born”;` voor het specificeren van een label, maar voor bijvoorbeeld `rdfs:ranges`[15] voor de Semantic Forms autocompletion functionaliteit[16] bestaat geen Fresnel. In deze gevallen moet Fresnel uitgebreid worden met een statement uit onze owf_style.owl ontologie. Deze ontologie heeft de prefix “`owf:`”, als voorbeeld:

```
:employerFormat owf:autocompleteFromClass :Employer .
```

5. **Waar nodig, breidt de GUI uit.**Gebruikers van Fresnel Forms moeten met de GUI stileringsaspecten kunnen specificeren. Bijvoorbeeld moet aangegeven kunnen worden of

[13] `http://www.w3.org/2001/sw/wiki/Semantic_Web_terminology#property`

[14] `http://www.mediawiki.org/wiki/Extension:Semantic_Forms/Semantic_Forms_and_templates#.5Cn_delimiter`

[15] `http://www.w3.org/TR/rdf-schema/`

[16] http://www.mediawiki.org/wiki/Extension:Semantic_Forms/Defining_forms#Autocompletion

een in een formulier ingevoerde waarde, zoals een URL, als een plaatje of als een URL wordt getoond. Ook moeten gegenereerde invoerformulieren op basis van het type van een semantische property[17] automatisch het juiste invoertype laten zien, zoals radio-buttons of een datumveld.

6. Exporteren van gegevens in RDF. Als voorgesteld alternatief voor Wikipedia-DBpedia moet een door Fresnel Forms gegenereerde wiki ook gegevens kunnen exporteren in RDF om te kunnen delen op het Semantic Web.

7. Optie in de GUI voor heuristisch sorteer algoritme. Een andere requirement voor het benaderen van een Wikipedia infobox (requirement 2) is het GUI-matig sorteren van properties. Deze benadering wordt vereenvoudigd door het te implementeren heuristische sorteer-algoritme van Paul (Paul, 2014).

Niet-functionele requirements

1. Protégé plug-in. Fresnel Forms moet gebruik maken van de Java API's voor Protégé[18]. De Javacode en documentatie worden dus gemaakt en ingepakt als een Protégé plug-in.

2. Importeren en exporteren DBpedia-ontologie. Het zou goed zijn voor de impact van Rutledge's onderzoek als Fresnel Forms in staat is om de volledige DBpedia-ontologie te importeren in de GUI als Fresnel-model en vervolgens als wikicode te exporteren. Pas de GUI en back-end aan om dit mogelijk te maken.

3. Toekomstbestendigheid en kwaliteit. Naast de niet-functionele requirements, zoals opgesteld door de opdrachtgever, hebben we onszelf als doel gesteld om kwalitatief goede, platformonafhankelijke, en onderhoudbare, makkelijk uit te breiden software te maken. Dit alles om mogelijk te maken dat andere onderzoekers of studenten makkelijk voort kunnen bouwen op dit *proof of concept.*

4. Algemeen overzicht domeinen en technieken

In dit Hoofdstuk geven we een algemeen overzicht van de domeinen en technieken. Dit overzicht is opgedeeld in twee domeinen, namelijk:

- MediaWiki - het domein van het doel waarvoor het project een user interface genereert.
- Semantic Web - het domein van de bron waar de gegevens voor de user interface gedefinieerd zijn.

MediaWiki

MediaWiki[19] is een gratis wiki pakket, geschreven in de programmeertaal PHP. Het is ontworpen als basis voor de online encyclopedie Wikipedia[20], maar wordt tegenwoordig ook als basis voor vele andere wiki's gebruikt.

---

[17] `http://semantic-mediawiki.org/wiki/Help:Special_property_Has_type`

[18] `http://protege.stanford.edu/`

[19] https://www.mediawiki.org/wiki/MediaWiki

[20] http://nl.wikipedia.org/wiki/Hoofdpagina

Een wiki is een toepassing waarmee gebruikers eenvoudig gemeenschappelijk inhoud kunnen beheren en vormgeven. Om het gebruik te vereenvoudigen wordt vaak gebruik gemaakt van een eenvoudige markup taal. In de voorbeelden van dit hoofdstuk is de markup-taal gebruikt zoals van toepassing is op MediaWiki.

MediaWiki is een wiki welke bestaat uit pagina's welke onderverdeeld zijn in namespaces. Zo is er onder meer een namespace Help voor ondersteuningspagina's, een namespace Talk voor discussiepagina's en een namespace User voor gebruikerspagina's. Pagina's binnen een bepaalde namespace hebben een naam die begint met de namespace, gevolgd door een dubbele punt (:). Zo is de pagina met de naam 'Template:Welcome', de 'Welcome' pagina binnen de 'Template' namespace.

Voor dit project zijn de Template en Category namespaces van extra belang.

De Template namespace. De Template namespace wordt gebruikt voor pagina's welke op andere pagina's als sjabloon gebruikt kunnen worden.

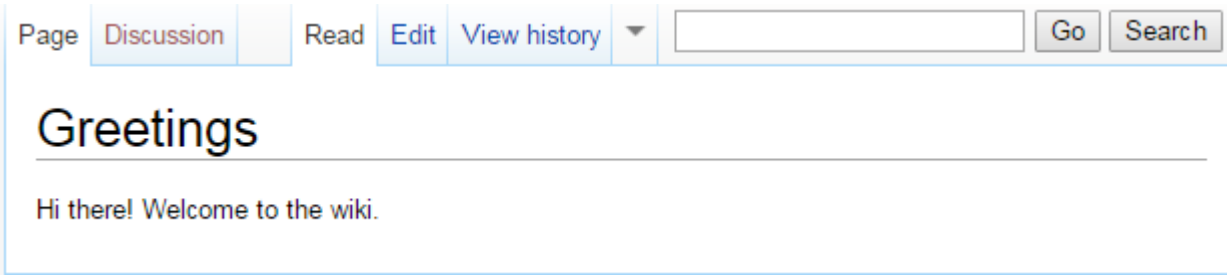
Wanneer we bijvoorbeeld de pagina 'Template:Welcome' creëren met de volgende broncode:

```
Welcome to the wiki.
```

En vervolgens een willekeurige pagina, bijv. 'Greetings', aanmaken met de volgende broncode:

```
Hi there! {{Welcome}}
```

Dan zal de pagina er als volgt uitzien wanneer het geraadpleegd wordt:



Figuur 5. Wikipagina 'Greetings'.

De aanroep van een sjabloonpagina kan ook parameters bevatten, waarmee de aanroepende pagina invloed heeft op de specifieke verwerking van de sjabloonpagina.

Wanneer we de inhoud van de 'Template:Welcome' pagina bijvoorbeeld wijzigen in:

```
{{{greetings}}} to the wiki.
```

En de 'Greetings' pagina de volgende inhoud geven:

```
{{Welcome|greetings=Hi There! Welcome}}
```

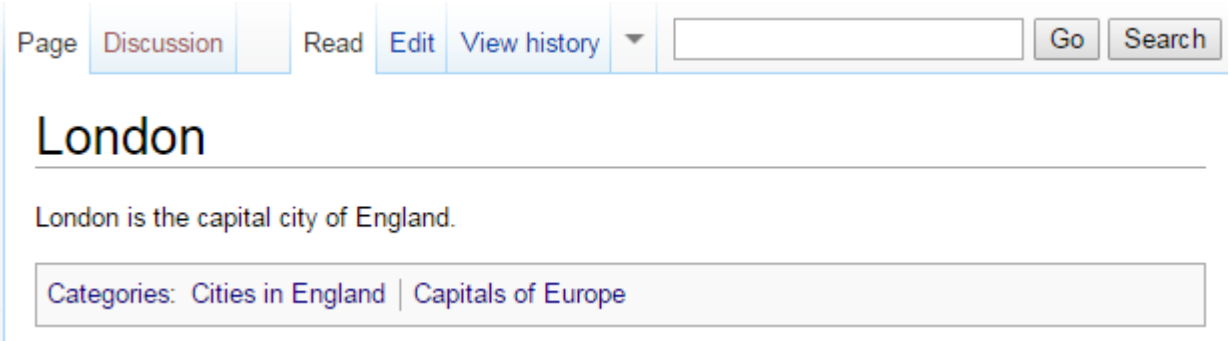
Dan wordt precies hetzelfde resultaat gegeven wanneer de pagina 'Greetings' wordt geraadpleegd, maar kan het sjabloon 'Template:Welcome' ook voor andere begroetingspagina's gebruikt worden door de parameter 'greetings' een andere waarde te geven.

De Category namespace. De Category namespace wordt gebruikt om pagina's te categoriseren.

Een Category pagina (een pagina welke tot de Category namespace behoort) bevat een automatisch samengestelde lijst van alle pagina's die tot die categorie behoren.

Wanneer we bijvoorbeeld een pagina 'London' aanmaken met de volgende broncode:

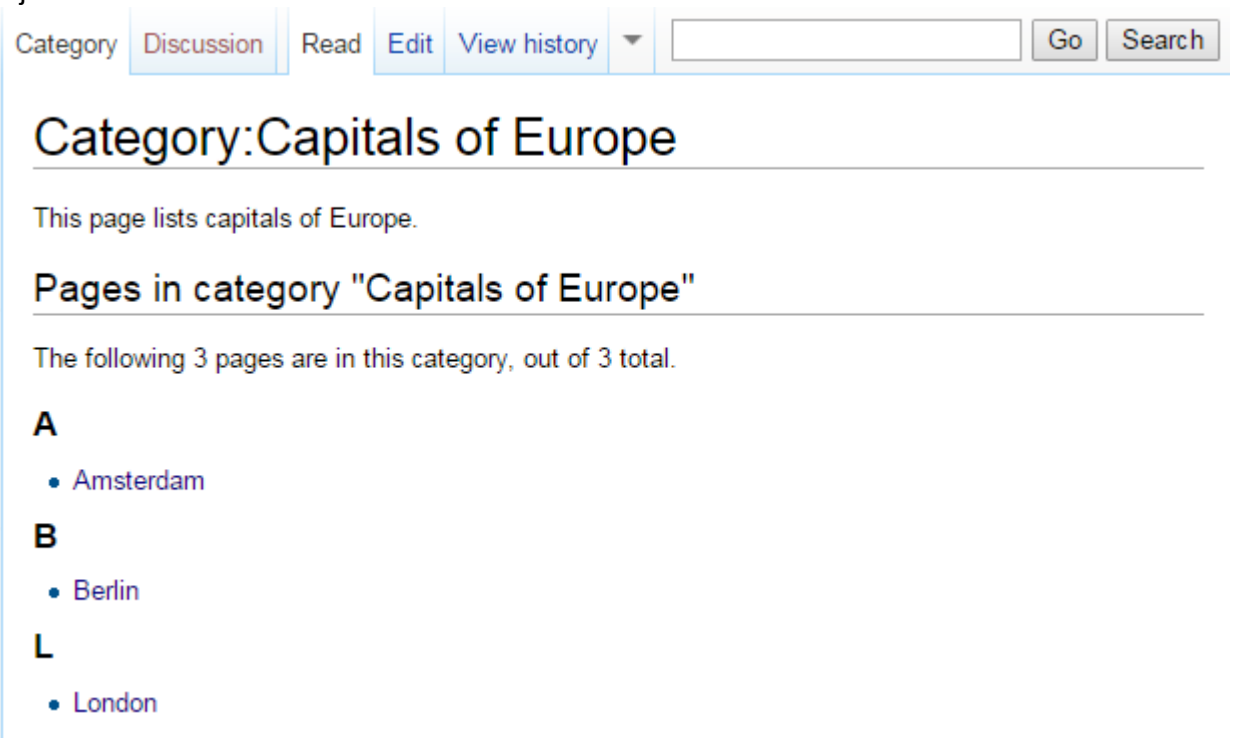
```
London is the capital city of England.

[[Category:Cities in England]]
[[Category:Capitals of Europe]]
```



Figuur 6. Wikipagina met categoriën.

Dan komt deze er uit te zien bij het raadplegen als in Figuur 6.

Daarnaast zal de 'Category:Capitals of Europe' pagina automatisch een lijst weergeven van alle pagina's welke `[[Category:Capitals of Europe]]` in de broncode hebben staan. Dus bijvoorbeeld:



Figuur 7. Wiki categorie overzichts pagina.

Semantic MediaWiki. Semantic MediaWiki[21] is een gratis en open source uitbreiding op MediaWiki waarmee MediaWiki in een uitbreidbaar en flexibel kennisbeheersysteem is te transformeren. Alle gegevens binnen Semantic MediaWiki kunnen eenvoudig via het Semantic Web gepubliceerd worden waardoor deze ook daar eenvoudig te gebruiken zijn.

Semantic MediaWiki breidt met name de Category functionaliteit van MediaWiki uit door een Property namespace te definiëren. Pagina's in de Property namespace representeren eigenschappen welke op andere pagina's kunnen gelden. Op een pagina kan een eigenschap dan een bepaalde waarde worden gegeven en de overeenkomstige pagina in de Property namespace bepaalt dan welk type de eigenschap heeft.

Als voorbeeld kunnen we eigenschappen aan het bovenstaand voorbeeld van de pagina 'London' toevoegen door bijvoorbeeld de eerste regel te wijzigen in:

```
[[Name::London]] is the capital city of [[Country::England]].
```

Hierbij heeft de eigenschap 'Name' de waarde 'London' gekregen en de eigenschap 'Country' de waarde 'England'.

Wanneer dan de 'Property:Name' pagina (dus de 'Name' pagina binnen de Property namespace) de volgende inhoud heeft, is de 'Name'-eigenschap van het type String:

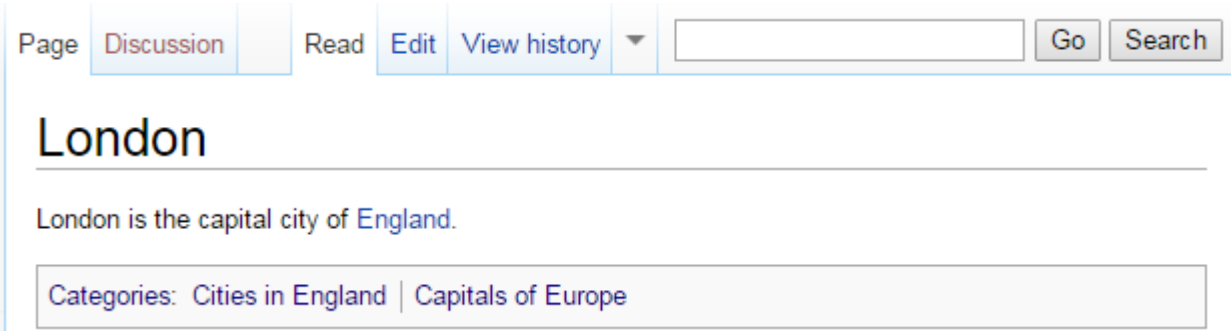
```
This property has type [[Has type::String]].
```

Wanneer dan de 'Property:Country' pagina de volgende inhoud heeft, is het van het type Page, wat tevens het default type voor een eigenschap is:

```
This property has type [[Has type::Page]].
```

Eigenschappen van type String worden als tekst weergegeven en eigenschappen van type Page worden weergegeven als hyperlink naar een pagina met als naam de waarde van de eigenschap.

De resulterende 'London' pagina zal er dan als volgt uitzien bij het raadplegen:



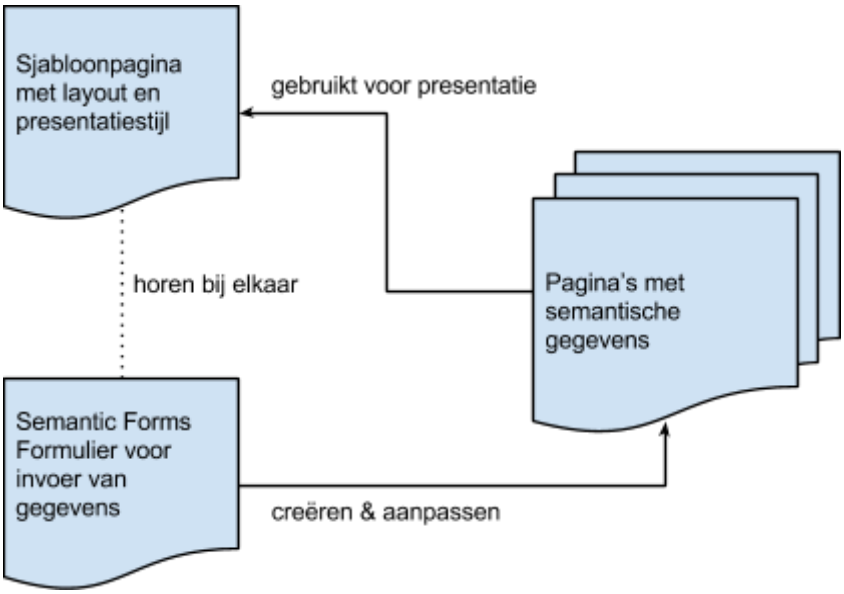
Figuur 8. Semantic MediaWiki pagina 'London'.

Hierbij wordt de tekst 'London', net als voorheen, gewoon als tekst en de tekst 'England' als hyperlink naar de pagina 'England' weergegeven.

De Category namespace wordt in Semantic MediaWiki gebruikt als equivalent van Semantic Web classes en de Property namespace wordt gebruikt als equivalent voor Semantic Web properties. Op deze manier wordt een semantisch systeem gecreëerd welke functionaliteit heeft die voor wat betreft (sub)klassen en (sub)properties equivalent is aan het Semantic Web.

[21] http://semantic-mediawiki.org/

Dit heeft als voordeel dat gegevens, welke zijn gedefinieerd in Semantic MediaWiki, eenvoudig gedeeld kunnen worden met het Semantic Web. Hierin wordt voorzien door de Special:RDFExport[22] pagina, welke door Semantic MediaWiki meegeleverd is. Door met name het verschil in typesysteem worden niet alle mogelijkheden van het Semantic Web ondersteund, maar dit is in de meeste gevallen niet beperkend (hoofdstuk 6).



Figuur 9. Informatie flow binnen Semantic MediaWiki.

Semantic Forms. Semantic Forms[23] is een uitbreiding op Semantic MediaWiki waarmee formulieren kunnen worden gedefinieerd waarmee op een eenvoudige manier gegevens binnen Semantic MediaWiki gecreëerd en aangepast kunnen worden. Hierbij worden formulierpagina's gebruikt waarmee voor bijbehorende sjabloonpagina's (binnen de Template namespace) gegevens onderhouden worden voor pagina's welke de betreffende sjabloonpagina's gebruiken voor de presentatie ervan. Figuur 9 laat zien hoe dit gebeurt.

Het Semantic Web

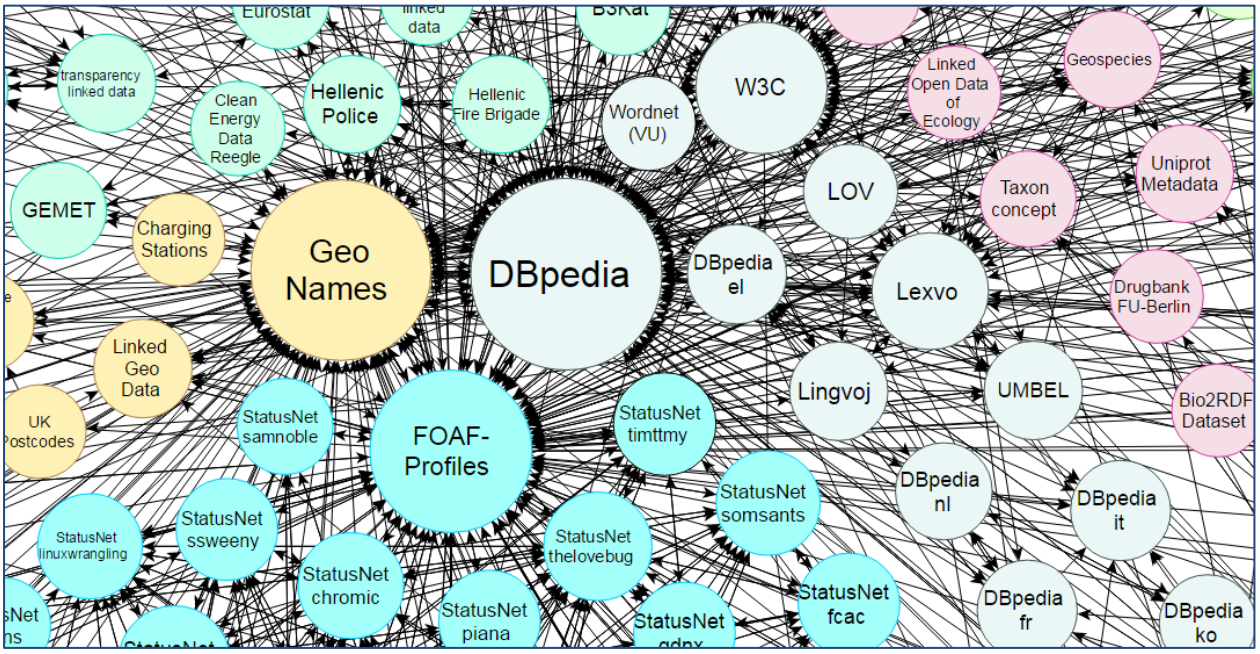
Het Semantic Web is een raamwerk voor het delen en hergebruiken van betekenisvolle kennis (Antoniou & van Harmelen, 2008).

In het Semantic Web worden entiteiten uniek gedefinieerd door de definitie ervan op basis van URI's; dezelfde entiteit kan wel bekend zijn onder verschillende URI's, maar een URI definieert altijd een unieke entiteit.

[22] `https://semantic-mediawiki.org/wiki/Help:RDF_export`

[23] `http://www.mediawiki.org/wiki/Extension:Semantic_Forms`

Wat ook ten grondslag ligt aan het Semantic Web is dat de vastgelegde kennis nooit volledig is; er is altijd de mogelijkheid dat er meer te weten valt over de informatie die vastgelegd is.



Figuur 10. De 'Linked Open Data Cloud'.

Daarnaast geldt, omdat het een Web betreft en daardoor een gemeenschappelijk goed is, dat iedereen iets kan beweren over elk gegeven in het Semantic Web.

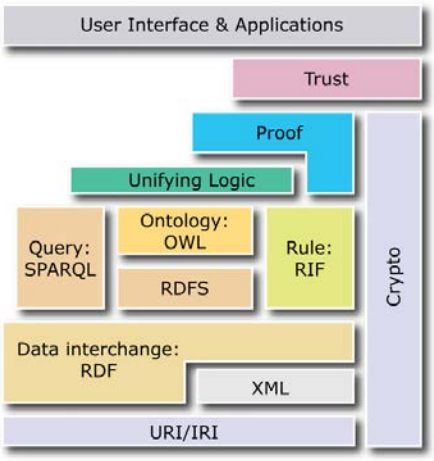
De gegevens binnen het Semantic Web zijn vastgelegd in ontologieen, wat formele beschrijvingen zijn van de werkelijkheid en welke verwijzingen naar elkaar kunnen hebben. Deze resulterende wolk van ontologiëen wordt ook wel de Linked Open Data Cloud genoemd. Figuur 10 laat een uitsnede zien van de Linked Open Data Cloud in aug. 2014.

Hierin is te zien dat er, naast ontologieën die vooral naar andere ontologieën verwijzen, ook een aantal ontologieën zijn waar erg veel naar verwezen wordt, zoals DBpedia, FOAF-Profiles en W3C. De complete Linked Open Data Cloud is te zien op http://lod-cloud.net/.

Het Semantic Web maakt gebruik van een aantal W3C standaarden en technieken. Deze zijn in figuur 11 weergegeven.

De belangrijkste Semantic Web standaarden voor dit afstudeerproject zijn URI/IRI, RDF, RDFS en OWL.

RDF is hierbij, als Resource Description Framework, de basis van het Semantic Web waarbij Uniform Resource Identifiers (URI's) gebruikt worden om objecten te identificeren (Allemang & Hendler, 2008). Een IRI is de geïnternationaliseerde variant van een URI en heeft een groter tekenset waardoor meer vreemde tekens weergegeven kunnen worden. De RDFS en OWL lagen voegen gaandeweg meer semantiek toe voor met name de classificering van resources.

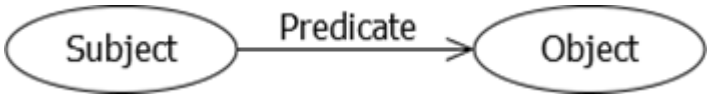


Figuur 11. De W3C standaarden waarop het Semantic web gebaseerd is.

RDF. Het Resource Description Framework is een

datamodel voor objecten en relaties daartussen welke zijn vormgegeven als uitspraken waarbij een subject, door een bepaald predicaat, of property, gerelateerd wordt aan een object. Deze uitspraken worden in RDF tripels genoemd. Deze tripels kunnen als gerichte graaf worden weergegeven (Figuur 12).

Hierbij worden de subjecten en predicaten geïdentificeerd door URI's. Het object van een tripel kan een specifiek object zijn met een bepaalde URI, of een letterlijke waarde van een bepaald type.



Figuur 12. De RDF triple.

Alle uitspraken (tripels) van een ontologie samen, vormen een gerichte graaf waarin de objecten en subjecten de vertices zijn en de predicaten (relaties) de kanten.

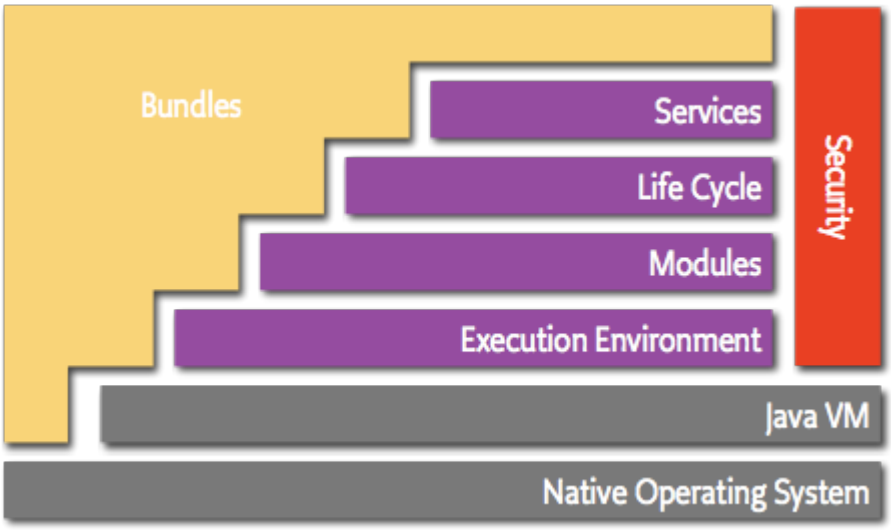
Fresnel. Fresnel[24] is een browseronafhankelijke vocabulaire voor het definiëren van een presentatie voor de gegevens van een willekeurige ontologie. Het bestaat uit de definitie van lenzen welke definiëren wat er getoond moet worden, en formats welke definiëren hoe de gegevens getoond moeten worden. Fresnel is zelf gedefinieerd in RDF.

Protégé. Protégé[25] is de meest gebruikte ontology editor. Het is opgebouwd als OSGi toepassing met een aantal modules en is uitbreidbaar met plug-ins. Het resultaat van dit afstudeerproject, Fresnel Forms, is een OSGi plug-in (bundle) voor Protégé.

Met de combinatie van Protégé en de Fresnel Forms plug-in is het mogelijk om een bepaalde ontologie te onderhouden én om er een user interface beschrijving voor te maken in abstracte zin (Fresnel) én specifieke zin (voor MediaWiki / Semantic MediaWiki / Semantic Forms).

OSGi

OSGi[26] is een raamwerk voor het samenstellen van modulaire, uitbreidbare Java toepassingen. De architectuur ervan bestaat uit de lagen zoals getoond in Figuur 13.



Figuur 13. Het OSGI raamwerk.

Zoals beschreven is Protégé gebouwd als OSGi toepassing. Ook Eclipse is een voorbeeld van een bekende OSGi toepassing waarbij plug-ins, welke in OSGi Bundles worden genoemd, de toepassing kunnen verrijken.

[24] http://www.w3.org/2005/04/fresnel-info/

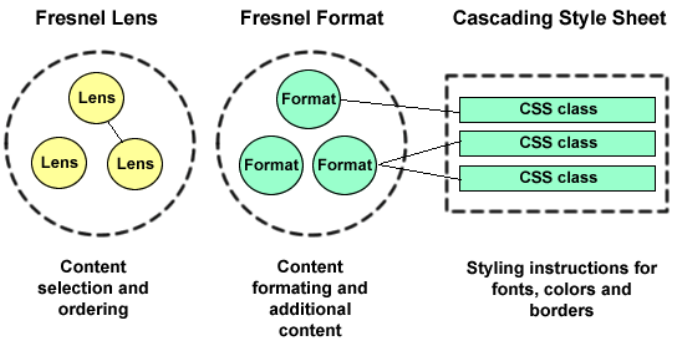
[25] http://protege.stanford.edu/

[26] http://www.osgi.org/

CSS

CSS[27] staat voor Cascading Style Sheets en is een eenvoudig mechanisme om stijl (lettertype, kleuren en plaatsing) toe te voegen aan Web documenten. Het is zodanig gelaagd opgebouwd dat web-browsers een default stylesheet kunnen definiëren welke door gebruikers kan worden aangevuld of overschaduwd met specifieke stijlinformatie voor (delen van) web documenten.

In Fresnel wordt binnen een Format de stijl van elementen gedefinieerd door de CSS attributen resourceStyle, propertyStyle, labelStyle en valueStyle.



Figuur 14. Kern begrippen van de Frenselvocabulaire.

5. Verslag onderzoek deeldomein en bijbehorende technieken (individueel)

[27] http://www.w3.org/Style/CSS/

6. Verslag analyse onderzoekcontext

Dit hoofdstuk bevat de verslaglegging van het consult voor de onderzoekscontext op vrijdag 1 mei 2015 van de auteurs met de opdrachtgever Rutledge. Voor dit consult waren, naar aanleiding van persoonlijke zoektochten naar de context van het onderzoek waarin Fresnel Forms een rol speelt, een aantal vragen voorbereid. De zoektochten en motivaties voor de vragen zijn gedocumenteerd in de bijlagen XVI (vragen 1-4), XVII (vragen 5-9) en XVIII (vragen 10-13). De vragen worden in dit hoofdstuk beantwoord.

Vraag 1

Gegeven het feit dat het Semantic Web en Semantic MediaWiki geen volledig equivalente semantische informatiesystemen zijn, constateren we dat gegevens die middels een door Fresnel Forms gegenereerde user interface voor Semantic MediaWiki ingevoerd zijn, niet in alle gevallen RDF tripels opleveren die aan de uitspraken in de bron-ontologie voldoen. Vindt u dit een beperking voor de gekozen oplossing?

Één van de grootste problemen die hieraan ten grondslag ligt, is het feit dat de typesystemen van Semantic MediaWiki en het Semantic Web anders zijn. Dit probleem bestaat uit twee delen:

1. Het geëxporteerde type kan anders zijn dan het beschreven type in de bronontologie.
2. De ingevoerde waarde kan ongeldig zijn voor het beschreven type in de bronontologie.

Om te bepalen of dit een beperking is, zou volgens Rutledge uitgezocht moeten worden óf deze problemen ook daadwerkelijk problemen zijn, door het bijvoorbeeld gewoon uit te proberen in Protégé.

Met behulp van Protégé en de HermiT reasoner is daarom onderzocht wat de gevolgen van deze deelproblemen zijn, met de volgende bevindingen:

1. Deelprobleem 1 heeft in principe geen gevolgen mits de waarde van de property binnen de intersectie van de lexicale ruimten van beide typen valt. Wanneer bijvoorbeeld een type van een property beschreven is als xsd:unsignedLong, maar het geëxporteerde type is een xsd:integer, dan is een property met een waarde groter dan, of gelijk aan nul, toch consistent met het beschreven type. Een waarde kleiner dan nul zal dan echter wel in een inconsistentie resulteren, het is dan geen onderdeel van de lexicale ruimte van het type xsd:unsignedLong.
2. Deelprobleem 2 heeft als gevolg dat, wanneer de geëxporteerde waarden in de bron-ontologie geïmporteerd zouden worden, deze dan inconsistent wordt. De HermiT reasoner geeft in dat geval foutmeldingen weer over gevonden inconsistenties.

Dit betekent dat wanneer de ingevoerde waarde (welke per definitie geldig moet zijn binnen de lexicale ruimte van het geëxporteerde datatype) geldig is binnen de lexicale ruimte van het datatype van de bronontologie, er geen consequenties zijn voor de geldigheid van de ingevoerde gegevens. Er zijn echter veel gevallen waarin het fout kan gaan, met name omdat numerieke waarden per definitie als xsd:double geëxporteerd worden en alle datatypen voor gehele getallen geen decimaal scheidingsteken ondersteunen. Er is dus sprake van een beperking. Vraag 3 gaat hier verder op in.

Vraag 2

Zouden, in aansluiting op vraag 1, de owf:list en owf:mandatory properties niet beter direct uit de bron-ontologie herleid kunnen worden (uit owl:min- en owl-maxCardinality) in plaats van aanpasbaar te zijn in Fresnel Forms en ruimte te geven voor discrepanties met de bron-ontologie?

In het geval van de owf:list en owf:mandatory uitbreidingen op Fresnel geldt in veel gevallen niet dat dit inconsistenties oplevert. Dit is omdat er in het Semantic Web de aanname geldt dat er altijd meer te weten valt.

Een owl:minCardinality restrictie voor een bepaalde property in de bron-ontologie betekent dan dat wanneer een gebruiker geen waarde heeft ingevuld, dit het nog niet inconsistent maakt met de uitspraken in de bron-ontologie. Er kan altijd iemand anders zijn die het de verplichte waarde geeft. Een owl:minCardinatlity restrictie geeft slechts aan dat er voor de betreffende klasse minimaal één waarde voor die property bestaat.

Als gevolg hiervan wil je als gebruiker van de plug-in de mogelijkheid hebben om aan te geven of iets verplicht ingevuld moet worden of niet, met een geschikte default op basis van de uitspraken in de bron-ontologie.

De conclusie van deze vraag is dat onze onderliggende aanname, dat de betreffende owf properties tot inconsistenties zouden kunnen leiden, niet geldt binnen het Semantic Web.

Vraag 3

Het lijkt haalbaar, doch niet triviaal, dat middels een samengestelde extensie voor Semantic MediaWiki een betere aansluiting gevonden wordt van Semantic MediaWiki met de datatypes zoals die in het Semantic Web gebruikt worden. Is het volgens u de moeite waard om dit verder te onderzoeken in een volgend ABI project?

Hoewel het een interessante uitbreiding is van het systeem, en het de semantische systemen zeker dichter bij elkaar brengt, lijkt de matige omvang van zo'n uitbreiding niet in aanmerking te komen als ABI project. Daarnaast zal het geen aanleiding zijn voor een nieuw paper. Wat wel interessant is om te onderzoeken is hoever je kunt gaan door het definiëren van een ontologie waarin bijvoorbeeld voor kommagescheiden getallen aparte velden gebruikt worden waarin de cijfers vóór en ná de komma van het getal afzonderlijk ingegeven worden, soortgelijk de invoer van getallen bij websites van sommige banken. Hiermee zou dan ook consistentie afgedwongen kunnen worden.

Het is idee is zeker interessant, maar binnen onze planning echter niet haalbaar om dit verder te onderzoeken. Het is derhalve als aanbeveling voor de toekomst opgenomen in hoofdstuk 11.

Vraag 4

Fresnel is slechts geschikt als presentatiedefinitie en niet als invoerdefinitie; de presentatie van een formulier in Semantic Forms kan, afgezien van de volgorde van velden, niet middels Fresnel gedefinieerd worden. De benodigde extra invoerproperties zijn nu als kleine uitbreidingen op de mogelijkheden van een Fresnel Format gedefinieerd, maar vindt u ook dat dit een meer prominente plaats verdient in een platformonafhankelijke user interface ontologie bijvoorbeeld op basis van W3C normen als XForms? Zo ja, komt ook dit dan in aanmerking om verder te onderzoeken in een volgend ABI project?

In het paper is aangetoond dat de huidige methode werkt en voldoende uitbreidbaar is. Het kan natuurlijk altijd anders, maar in dit geval hoeft een form niet slechts uit invoerelementen te bestaan (denk aan velden welke slechts gelezen kunnen worden). Alles wat we nu willen kan met Fresnel Forms en nieuwe uitbreidingen kunnen toegevoegd worden, dus een ander ontwerp van de presentatie- of formulierontologie heeft weinig toegevoegde waarde. Daarnaast is de huidige oplossing backward compatible en kan door een willekeurige Fresnel browser worden omgezet in een user interface, hoewel er niet veel Fresnel browsers meer beschikbaar zijn.

Deze vraag heeft derhalve geen consequenties voor Fresnel Forms.

Vraag 5

Een verstandige en praktische manier om nieuwe technieken zoals MDD te introduceren is als extensie van een bestaand project. Dit verlaagt de risico's en maakt gebruik van al gemaakte investeringen (Selic, 2003)*. Fresnel Forms maakt handig gebruik van het beheren van een ontologie door Protégé. Wellicht dat deze succesvol gebleken aanpak nieuwe wegen kan openen voor onderzoek naar nieuwe tools die hierop voortbouwen?*

Rutledge merkt in het consult op dat deze aanpak onderzoekstechnisch niets nieuws oplevert, wel dat Frensel Forms in combinatie met Protégé een belangrijke aanwinst voor de praktijk is.

Vraag 6

Standaardisatie van modelformaat heeft voordelen zoals het faciliteren van gebruik door verschillende tools, het vastleggen van best practices, aanmoedigen van hergebruik en specialisatie (Selic, 2003)*.Zal het onderzoek van Rutledge in een vorig artikel* (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013) *en het verwachte artikel op basis van onder andere dit project bijdragen aan een verdere acceptatie van Fresnel als standaard vocabulaire voor UI's van informatiesystemen gebaseerd op het Semantic Web, zoals Fresnel Forms produceert?*

Fresnel is in 2006 ontwikkeld en was toen kortstondig populair. Hoewel er een paar Semantic Browsers zijn geweest die Fresnel ondersteunden is het nooit echt opgepakt en geen standaard geworden, wat vreemd is omdat er ook geen alternatieven zijn.

Fresnel is voor het Semantic Web wat CSS voor XML is: een style sheet. In werkelijkheid verkennen mensen het Semantic Web echter anders dan het World Wide Web. Vooraf ingevoerde data wordt geanalyseerd en gerapporteerd, het volgen van hyperlinks wordt niet veel gedaan. Wikipedia is in die zin een belangrijke Semantic Browser en de aanpak met Fresnel Forms maakt dit efficiënter doordat op MediaWiki gebaseerde user interfaces sneller gedefinieerd kunnen worden vanuit ontologieën uit het Semantic Web. Fresnel Forms bewijst dat Fresnel goed te gebruiken en uit te breiden is voor deze doelstelling en wellicht maakt dat deze vocabulaire meer relevant.

Vraag 7

Voor het produceren van een prototype van een wiki-UI met Fresnel Forms is het nu nog nodig om het geproduceerde platformspecifieke model handmatig te importeren op een

MediaWiki server met special:import[28]. Het is te verwachten dat het versnellen van dit proces met behulp van een API endpoint[29] de productiviteit van eindgebruikers van Fresnel Forms zal verhogen. Is dit interessant voor een vervolgonderzoek of -project?

Dit is voor het onderzoek niet van belang, maar heeft zeker praktische waarde wanneer het hele proces efficiënter kan.

Vraag 8

Niet alle informatie gerelateerd aan UI objecten kunnen bevat worden in een specifieke tool die alle doeleinden bestreikt. Daarom is er behoefte aan enige steun voor design gebaseerd op annotaties (Vanderdonckt, 2008)*. Fresnel Forms geeft deze mogelijkheid niet. Het is te overwegen om een optie in de GUI toe te voegen die eindgebruikers de mogelijkheid geeft om commentaar toe te voegen aan lenzen of properties. Deze zouden kunnen worden doorgegeven aan het platformspecifieke model (Fresnel met OWF) in de form van een OWF formatting property. Is dit een interessante aanvulling voor een vervolgonderzoek of -project?*

Zulke annotatie properties bestaan al, RDFS heeft bijvoorbeeld label en comment, ook XML heeft comments. Er is waarschijnlijk geen behoefte aan een optie in de GUI, andere MDD tools hebben deze namelijk ook niet. Een ontwikkelaar kan, indien nodig, annotaties handmatig in de code toevoegen.

Vraag 9

De mogelijkheden tot individualisatie van UI's die Fresnel Forms biedt zijn beperkt tot opties in de GUI en het invullen van CSS. Er bestaan extensies van MediaWiki die het toelaten om JavaScript code te laden, zoals de JavaScript extensie[30]. Zou het interessant zijn om te onderzoeken in hoeverre dit kan helpen om het Fresnel Forms platformspecifieke model verder te individualiseren?

JavaScript zou eventueel net als CSS gestructureerd toegevoegd kunnen worden aan Fresnel. Maar in feite heeft Rutledge zelf al een oplossing voor het probleem gepresenteerd in 2013. In een gegenereerde MediaWiki sjabloon wordt namelijk gecontroleerd op aanwezigheid van een "InformboxTop". Indien een wiki-ontwikkelaar deze al had gecreëerd, wordt deze bijgevoegd (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013).

Vraag 10

Ons project heeft enkele mappingen zoals gedefinieerd in de feature-tabel in Bijlage XIV geïmplementeerd. Wat betekent het voor uw onderzoek als alle mappingen zijn geïmplementeerd?

Voor het onderzoek is het meeste al bereikt en de aanpak is bewezen. Het toevoegen van nog meer functionaliteit voegt dan niet zoveel meer toe en bovendien is de paper

---

28 `http://www.mediawiki.org/wiki/Manual:Importing_XML_dumps#Using_Special:Import`

29 `http://www.mediawiki.org/wiki/API:Main_page`

30 `http://www.mediawiki.org/wiki/Extension:Javascript`

submission al geweest. Veel producten zijn op deze manier gemaakt en er zijn er dus ook veel die nooit zijn afgemaakt.

De verwachting van de auteurs was dat aangetoond moest worden dat een complete mapping mogelijk was maar dat blijkt niet zo te zijn. Ons project heeft aangetoond dat een mapping die niet mogelijk is met standaard Fresnel wel mogelijk is door de Fresnel ontologie uit te breiden. Dit is dus een belangrijke stap in het onderzoek.

Vraag 11

Wat zijn de voordelen van de aanpak zoals voorgesteld door Rutledge in 2013 (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013) *met Semantic MediaWiki t.o.v. het OntoWiki project?*

Semantic MediaWiki is als doel een veel breder platform dan OntoWiki met een grotere gebruikersgroep en veel meer extensies. Waar Semantic MediaWiki zich richt op het delen van kennis in het algemeen heeft OntoWiki als hoofddoel een directe export naar het Semantic Web, maar veel meer niet.

De verwachting van de auteurs was dat het OntoWiki project veel technische voordelen zou bieden omdat het functionaliteiten heeft specifiek voor het semantische web, zoals een exportfunctionaliteit naar RDF. Maar met zijn antwoord geeft Rutledge aan dat het doel om een groter publiek te bereiken, door een populair platform te gebruiken, belangrijker is dan de export naar RDF.

Ook gaf Rutledge aan dat het Mediawiki platform veel mogelijkheden biedt voor het ontwikkelen van extensies. Het ontwikkelen van de RDF exportfunctionaliteit behoort tot de mogelijkheden.

Vraag 12

Wat is de reden geweest om het onderzoek van Paul (Paul, 2014) *te starten?*

De reden is om zo efficiënt mogelijk user interfaces te bouwen. Wanneer dit automatisch kan gebeurt het het meest efficiënt. Het onderzoek van Paul bewijst dat er een methode is die in meer gevallen automatisch de juiste volgorde (voor Wikipedia infoboxes) geeft dan bij een alfabetische volgorde. Hoewel MDD-systemen doorgaans ook zo'n methode kennen is het daar van minder belang aangezien de aanpasbaarheid van gebruikers daar voorop staat.

Wat nog ontbreekt aan het onderzoek is of de methode voor mensen ook betere resultaten oplevert, dus of mensen tevreden zijn met de volgorde zoals die automatisch gegenereerd wordt. Een tevredenheidsonderzoek zou een goed vervolgonderzoek zijn.

De auteurs hadden geen verwachtingen maar waren verbaasd dat er zo weinig te vinden was over dit onderwerp op het internet. Het antwoord geeft duidelijk aan dat de behoefte aan een aanpasbare user interface-definitie, zoals gebruikt bij MDD, veel groter is dan een automatisch gegenereerde interface.

Vraag 13

Waarin onderscheidt het onderzoek van Rutledge zich ten opzichte van het onderzoek van Ławrynowicz (Ławrynowicz & Filipiak, 2014)*?*

Rutledge was zeer verbaasd over dit onderzoek. Hij was niet bekend met het onderzoek , maar wel bijzonder geïnteresseerd. Zijn wedervraag was dan ook om het door te lezen en te vergelijken met zijn onderzoek.

De verwachting van de auteurs was dat het onderzoek van Lawrynowicz minder interessant zou zijn omdat het een hele specifieke oplossing is en de aanpak van Rutledge veel algemener van opzet is. Rutledge was echter bijzonder geïnteresseerd in dit onderzoek omdat het wel heel erg lijkt op zijn onderzoek en het gestart is na de presentatie van zijn onderzoek. Dit voorval maakt duidelijk hoe belangrijk het is om een paper te publiceren over een onderzoek om de ideeën en het eigenaarschap vast te leggen.

Hieronder een korte samenvatting en vergelijking met het onderzoek van Rutledge.

De conversie van ontologie naar wiki in de tool gepresenteerd door Ławrynowicz (Ławrynowicz & Filipiak, 2014) geschiedt op basis van drie configuratiebestanden en een of meerdere ontologie bestanden. De configuratiebestanden dienen handmatig opgesteld te worden. Het sjabloonbestand bepaald welke attributen meegenomen worden. De mapping file koppelt de sjablonen aan entiteit-URI's. De conversie genereert artikel, categorie en sjabloon pagina's.

Voor de artikelpagina's wordt de categorie bepaald aan de hand van de klasse, de equivalente klasse en de ouderklasse. Fresnel Forms daarentegen doet dat niet, een lens staat dan voor een klasse en zijn equivalente klassen. Fresnel Forms maakt per lens een categorie.

Per individual wordt een artikel pagina aangemaakt. Dat doet Fresnel Forms niet, individuals worden niet meegenomen. Fresnel Forms genereert wel een forms pagina voor de invoer van individuals in de Semantic Media Wiki.

7. Beschrijving van de opgeleverde software producten.

Code en installatie

De releases. Het eindproduct van het ABI 2014 project is te vinden op GitHub projectpagina : Fresnel forms[31]

De GitHub projectpagina is gebruikt als release en als issue management platform.

GitHub biedt de mogelijkheid om per issue een discussielijn op te zetten. Hiervan is gebruik gemaakt om onderling te discussiëren en met de opdrachtgever Rutledge.

De final release is te vinden op Final Release Fresnel Forms[32].

De final release pagina biedt de download aan van de Fresnel [33]Forms plug-in als jar bestand. Het sourcecode zip bestand bevat enkele ontologie bestanden.

- nl.ou.cs.is.protege.plugin.fresnelforms.jar
- Source code (zip)

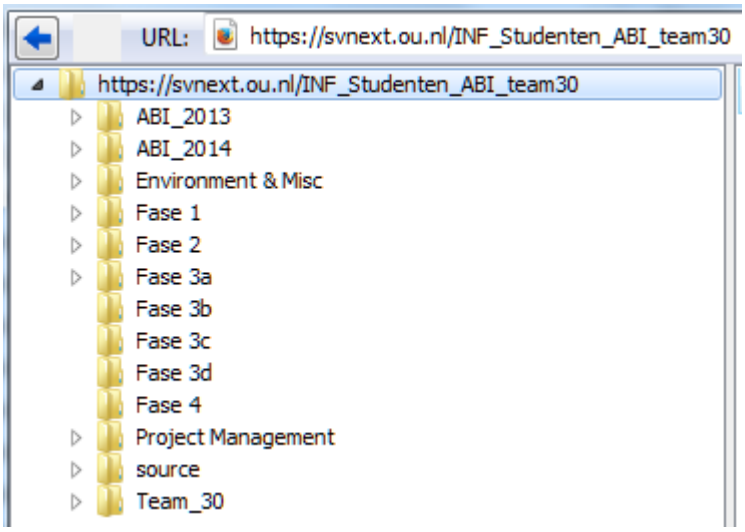
[31] https://github.com/ABI-Team-30/Fresnel-Forms

[32] https://github.com/ABI-Team-30/Fresnel-Forms/releases/tag/1.0.10

[33] http://www.w3.org/2005/04/fresnel-info/

In de issue sectie van de GitHub projectpagina zijn alle issues te vinden van het project. De issues die opgelost zijn in de final release staan onder ‘Closed’. De issues bestaan uit de vraag van de opdrachtgever en discussies die daaruit voortvloeiden. Voor het gebruik van de Fresnel Forms plug-in binnen de Protégé[34] ontologie editor is enkel het .jar bestand nodig. Zie sectie ‘Installatie Fresnel Form plug-in in protege’ voor de instructie.

Voor kennisborging is de ABI-2013 Semantic Media Wiki-pagina van de Open Universiteit (OU) gebruikt. Voor deze website is een login account noodzakelijk dus zijn de de pagina’s opgenomen in bijlage XV.



Figuur 15. De mappen structuur van de Subversion repository.

De sources

Bij de ontwikkeling van de Fresnel Forms plug-in is gebruik gemaakt van het versiebeheersysteem Subversion[35]. Alle sources, de Java code, de documentatie en de ontologieën zijn te vinden in de Subversion repository[36] op de Open Universiteit server.

De Subversion repository voor ABI team 30 bestaat uit een aantal mappen die documenten per onderwerp groeperen. Zie Figuur 15 voor een overzicht van de mappenstructuur, de mappen worden besproken in Tabel 1.

Tabel 1. De mappenstructuur van de repository.

Map	Beschrijving van de inhoud van de map
ABI_2013	De eindproducten van ABI team 28.
ABI_2014	De midterm presentatie.
Environment & Misc	Verschillende soorten documenten die niet onder een andere map vallen zoals ontologieën en documentatie over aanverwante onderwerpen.
Fase 1	De projectvoorstel en projectaanvraag documenten.
Fase 2	Het project plan

[34] http://protege.stanford.edu/

[35] https://subversion.apache.org/

[36] https://svnext.ou.nl/INF_Studenten_ABI_team30/

Fase 3a	De domeinanalyse documenten
Fase 4	De afstudeerscriptie
Project Management	Planningdocumenten
Source	De broncode van het project
Source\..\Design	De ontwerpdocumenten
Team 30	De competentiekeuze documenten

Installatie Fresnel Forms plug-in in Protégé. De plug-in wordt door Protégé herkend als het “nl.ou.cs.is.protege.plugin.fresnelforms.jar” bestand naar de “plugins” map onder de Protégé installatie map wordt gekopieerd. De plug-in Fresnel Forms wordt dan vermeld onder het menu “Windows - Tabs”.

Voorbeeld-URL van de installatiemap:

“D:\Program Files (x86)\Protege_5.0_beta\plugins”.

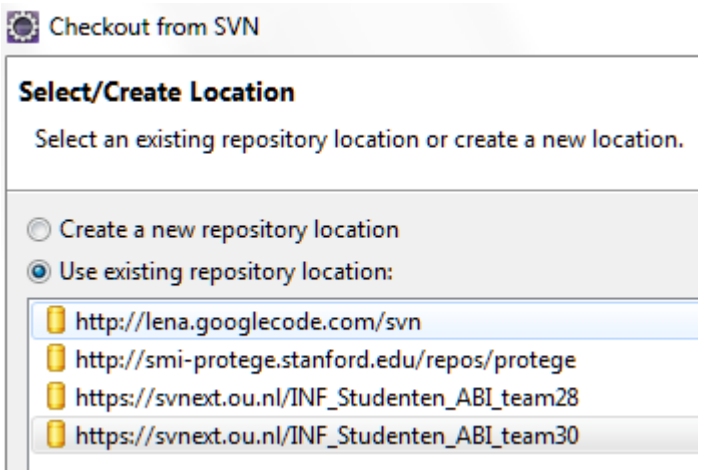
Installatie van de source code in Eclipse. De broncode van het project kan via Subversion naar het lokale bestandssysteem worden gekopieerd door middel van een checkout, zie Figuur 16.

In Eclipse kan ook een nieuw project gestart worden door te verwijzen naar de Subversion repository. Kies voor “File - New - Other - SNV - Checkout Projects from SNV”.

Kies de “ABI Team 30” repository.

Het bouwen van de plug-in. Fresnel Forms kan gebouwd worden met het commando ’`mvn clean install`‘ op de command line.

Na uitvoer hiervan kan het bestand nl.ou.cs.is.protege.plugin.fresnelforms.jar gevonden worden in de map “target”. Door het te kopiëren naar de map ‘plugins’ in de map waar Protégé geïnstalleerd is, kan de plug-in gebruikt worden.



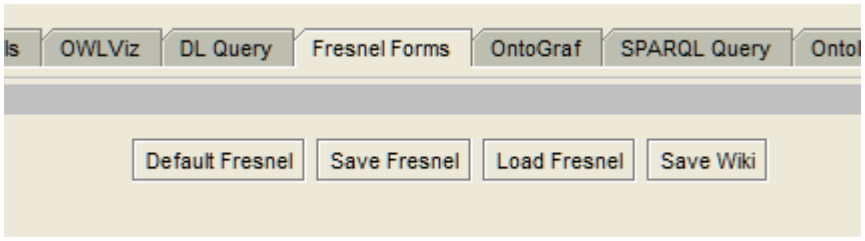
Figuur 16. Checkout scherm van Subversion.

Naast de plug-in zelf, wordt de projectdocumentatie ook gegenereerd tijdens uitvoer van bovenstaand commando. Deze is dan te vinden in de “target/site” map. Het bestand index.html kan dan gebruikt worden als startpunt.

Selecteer in deze repository de map “Source\trunc\FresnelForms” en klik op finish.

Functionele beschrijving van de Fresnel Forms plug-in.

Nadat Protégé is opgestart, is de Fresnel Forms plug-in aanwezig als tab zoals in Figuur 17. Mocht dat niet het geval zijn dan is de tab te activeren via menu “Windows - Tabs - Fresnel Forms” met een vinkje.



Figuur 17. De Fresnel Forms tab in de Protégé editor.

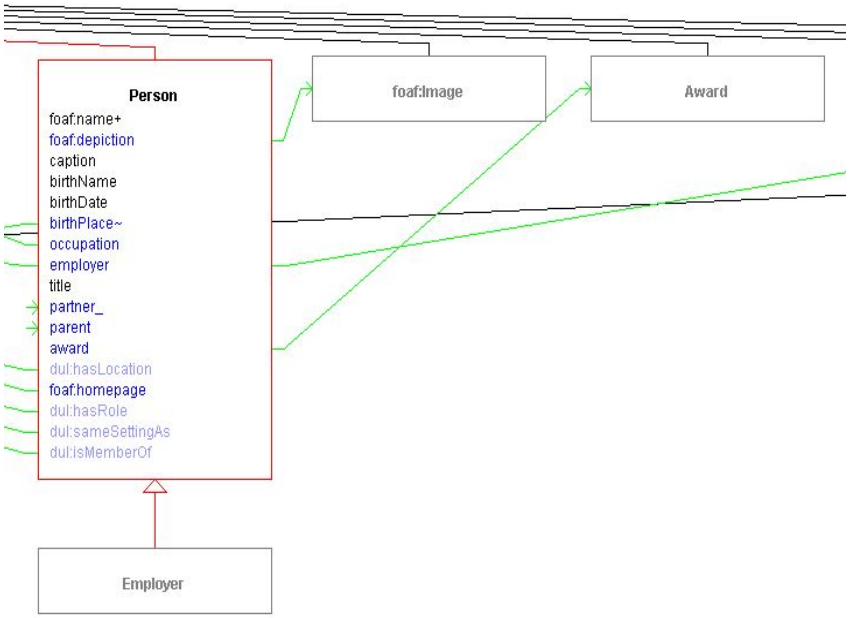
Nadat een ontologie in Protégé geladen is kan met de knop “Default Fresnel” een default Fresnel user interface-definitie gegenereerd worden. De definitie kan met de knop “Save Fresnel” bewaard worden in een “.n3” bestand. Met de knop “Load Fresnel” kan een Fresneldefinitie weer geladen worden. Met de knop “Save Wiki” wordt de Fresnel user interface-definitie vertaald naar een bestand in het Semantic Media Wiki Formaat. Dit bestand kan dan in de Semantic MediaWiki geladen worden via het menu “special pages - import pages”.

Voor een uitgebreide handleiding van de Fresnel Forms plug-in zie de bijlage map voor document: “Userinterface instructie.docx”.

De Fresnel Forms plug-in is een onderdeel van een MDD proces. In het MDD proces wordt van een bronontologie een user interface gegenereerd en vanuit deze user interface kan de bronontologie weer van data voorzien worden. Dit proces zal hieronder doorlopen worden met de demonstratie ontologie “owf_style_TBL_extract.owl”. Deze ontologie is een uittreksel van de DBpedia ontologie met klassen die betrekkingen hebben op personen.

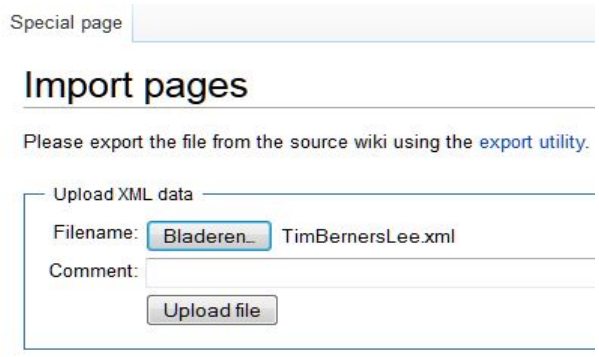
De bronontologie “owf_style_TBL_extract.owl” wordt geladen in de Protégé ontologie editor.

In de Fresnel Forms tab wordt met de knop “Default Fresnel” een user interface-definitie gegenereerd van onder andere bovenstaande de klassen gepresenteerd in Figuur 18. Figuur 18 laat zien dat voor de klasse Person de lens Person is gedefinieerd. Met het rechtermuismenu kunnen de eigenschappen van de properties nog aangepast worden. Voor de Person lens is voor de property foaf:name aangegeven dat het een verplicht attribuut is (+). Het “~” teken achter de property birthPlace geeft aan dat het om een auto-completion veld gaat. Het “_” teken achter partner geeft aan dat het om een enkelvoudig attribuut gaat.



Figuur 18. Lens weergave van de klasse 'Person'.

De Fresnel user interface-definitie zoals in Figuur 19 getoond, kan geëxporteerd worden naar Semantic MediaWiki via de knop “Save Wiki”. Deze Fresneldefinitie is naar het bestand TimBernersLee.xml geëxporteerd. Het xml bestand bevat de form- en template-definities. Dit bestand kan met de Semantic MediaWiki pagina “special pages - import pages” ingelezen worden (Figuur 19).



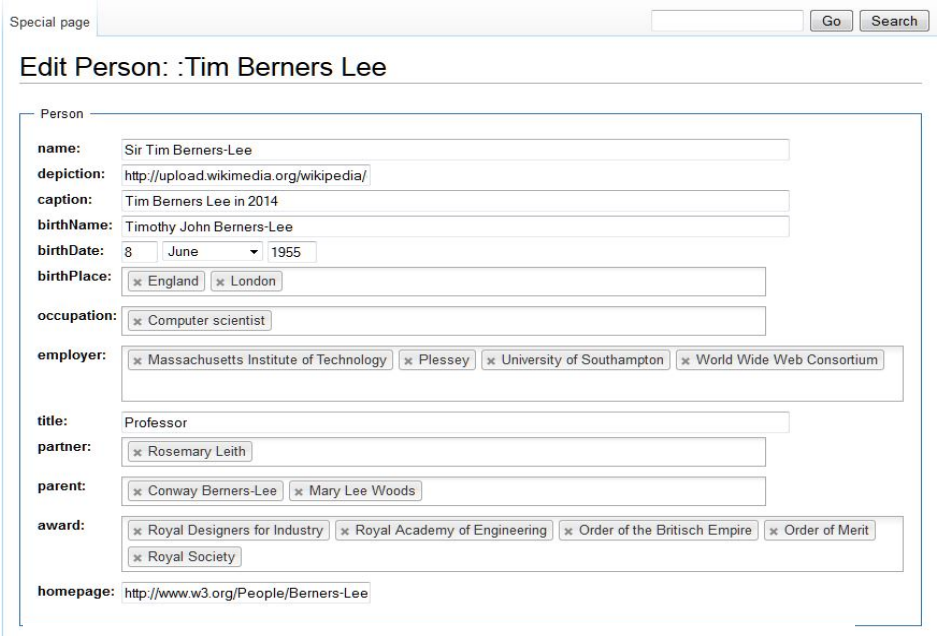
Figuur 19. De Semantic MediaWiki-importpagina

Nadat op de knop “Upload file ” van Figuur 19 geklikt is wordt het bestand ingelezen. Het resultaat van de import is de generatie van property-, template- en form-pagina’s (Figuur 20).

Fiauur 20. De Semantic MediaWiki

Via de gegenereerde formulieren kunnen gegevens ingevoerd worden in Semantic MediaWiki. Als voorbeeld wordt het formulier “Person” gebruikt om de gegevens voor Tim Berners Lee in te voeren (Figuur 21).

De ingevoerde gegevens worden getoond in een infobox die automatisch gegenereerd is bij de importeeractie (Figuur 22).



Figuur 21. Het Semantic MediaWiki Person invulformulier.

Als laatste stap kunnen de ingevoerde gegevens geëxporteerd worden naar een RDF bestand via “special pages - export pages to RDF”. Hoewel de export in deze vorm nog niet geschikt is om rechtstreeks in de bronontologie in te lezen geeft het wel aan dat de export mogelijk is.

Figuur 23 geeft een uittreksel van de geëxporteerde RDF.

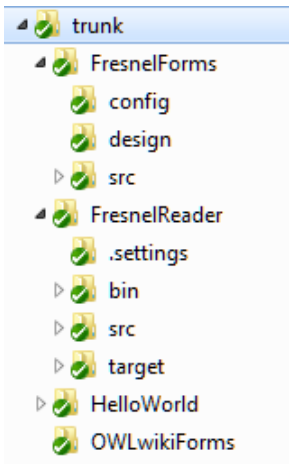
```
<rdf:RDF>
<owl:Ontology>
        <swivt:creationDate rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2015-05-
        16T23:13:37+02:00</swivt:creationDate>
        <owl:imports rdf:resource="http://semantic-mediawiki.org/swivt/1.0"/>
</owl:Ontology>
<swivt:Subject rdf:about="abiteam30.lukylx.orgTim_Berners_Lee">
<rdf:type rdf:resource="abiteam30.lukylx.orgCategory-3APerson"/>
<rdfs:label>Tim Berners Lee</rdfs:label>
<swivt:page rdf:resource="http://abiteam30.lukylx.org:3080/mediawiki/index.php/Tim_Berners_Lee"/>
<rdfs:isDefinedBy
rdf:resource="http://abiteam30.lukylx.org:3080/mediawiki/index.php/Special:ExportRDF/Tim_Berners_Lee"/>
<swivt:wikiNamespace rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">0</swivt:wikiNamespace>
<foaf:depiction rdf:resource="http://upload.wikimedia.org/wikipedia/commons/thumb/9/9d/Sir_Tim_Berners-
Lee.jpg/1280px-Sir_Tim_Berners-Lee.jpg"/>
<foaf:homepage rdf:resource="http://www.w3.org/People/Berners-Lee/"/>
<foaf:name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Sir Tim Berners-Lee</foaf:name>
<owf_style_tim_berners_lee_extract:award rdf:resource="abiteam30.lukylx.orgRoyal_Designers_for_Industry"/>
<owf_style_tim_berners_lee_extract:award rdf:resource="abiteam30.lukylx.orgRoyal_Academy_of_Engineering"/>
<owf_style_tim_berners_lee_extract:award rdf:resource="abiteam30.lukylx.orgOrder_of_the_Britisch_Empire"/>
<owf_style_tim_berners_lee_extract:award rdf:resource="abiteam30.lukylx.orgOrder_of_Merit"/>
<owf_style_tim_berners_lee_extract:award rdf:resource="abiteam30.lukylx.orgRoyal_Society"/>
```

Figuur 23. De Semantic MediaWiki-RDF-exportcode

Figuur 22. De door Fresnel gegenereerde infobox.

Algemene Beschrijving van de software

De trunk. In Figuur 24 staat de mappenstructuur van de trunk zoals die in de Subversion repository staat. De trunk bevat de mappen van het hoofdproject FrenselForms en het subproject FresnelReader. Het FresnelReader project betreft een controle-toepassing die een Fresnelbestand controleert op syntactische correctheid.



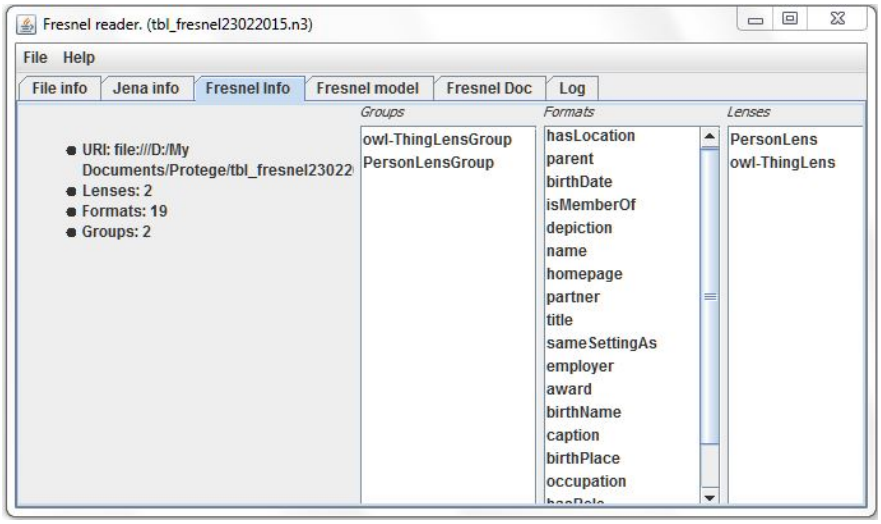
Figuur 24. Mappenstructuur van de trunk.

De map “FresnelForms” bevat het hoofdproject, de submap “config” bevat de configuratiebestanden. De submap “design” bevat de ontwerp documenten. De submap “src” bevat alle sourcecode bestanden.

De map FresnelReader bevat het subproject, de submap “bin” de .class-bestanden, de submap “target” het .jar-bestand en de de submap “src” de sourcecode bestanden.

De map “HelloWorld” bevat een standaard plug-inproject.
De map “OWLwikiFforms” bevat het PHP project van Rutledge dat een Fresnel bestand converteert naar Semantic MediaWiki formaat.

Het Fresnel Reader project. Het “Fresnel Reader”-subproject vult een gat dat ontstaan is door het ontbreken van Fresnelbrowsers. Op de W3C Fresnelinformatiepagina[37] worden vijf Fresnelbrowserprojecten beschreven, Het betreft de volgende projecten “IsaViz”, “LongWell”, “Horus”, “Lena” en “OAT”. Maar geen van deze projecten voldoet. De projecten “IsaViz” en “Lena” tonen niet de juiste informatie. De projecten “LongWell”, “Horus” en ”OAT” zijn niet meer actief; noch de website noch de software is te achterhalen. Een Fresnel-browser is voor ons project noodzakelijk om aan te tonen dat de Fresnelcode die door onze software wordt produceert geldig is.



Figuur 25. Fresnel Reader validatie applicatie.

Als alternatief is de Fresnel Reader toepassing geschreven op basis van de JFresnel API[38]. De JFresnel API implementeert de Fresnelspecificatie en wordt kan gebruikt worden met de standaard bibliotheken voor RDF zoals Jena[39] en Sesame[40].

[37] http://www.w3.org/2005/04/fresnel-info/

[38] http://jfresnel.gforge.inria.fr/

[39] https://jena.apache.org/

[40] http://rdf4j.org/

De “Fresnel Reader” toepassing bouwt een intern Fresnelmodel op vanuit een opgegeven bestand en toont de informatie over dit model op verschillende tabbladen. De gebruiker kan nu verifiëren of de informatie overeenkomt met hetgeen uitgedrukt moest worden met de Fresnelcode.

In Figuur 25 wordt het resultaat getoond van de gegenereerde Fresnelcode voor de demonstratie ontologie “owf_style_TBL_extract.owl”. Het tabblad “Fresnel Info” toont het aantal lenzen, formats en groepen die gevonden zijn in het Fresnelbestand. In de lijsten aan de rechterkant worden de afzonderlijke groepen, formats en lenzen getoond zodat de gebruiker het resultaat kan controleren.

Het Fresnel Forms project. Hierboven is de toepassing Fresnel Reader kort beschreven omdat de rol van de toepassing beperkt is tot het controleren of de ontworpen Fresneloplossingen syntactisch correct zijn.

De rest van dit hoofdstuk gaat over Fresnel Forms. Als eerst volgt een introductie op de structuur van het Fresnel Forms project en vervolgens een paragraaf over het functionele ontwerp met de ontwerpkeuzes die gemaakt zijn.

De structuur van het Fresnel Forms project.

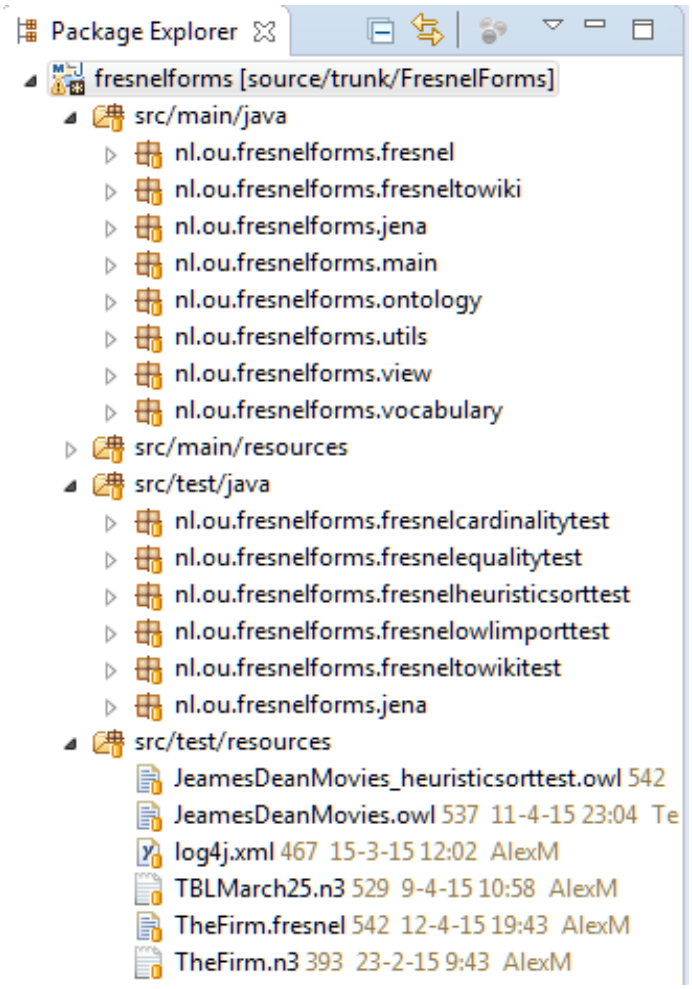
Figuur 26 toont de structuur van het Java EE project in Eclipse Luna 4.4.1. Het project “fresnelforms” bestaat uit vier mappen met daarin Java packages en resource-bestanden.

De map “src/main/java” bevat de Fresnel Forms plug-in packages.

De map “src/test/java” bevat de unit test packages.

De map “src/main/resources” bevat de resource-bestanden voor de Fresnel Forms plug-in en de map “src/test/resources bevat de resource-bestanden voor de unit testen.

Om inzicht te geven over de opbouw van het project zal hieronder de functie van de verschillende packages beschreven worden.



Figuur 26. De project package structuur in Eclipse.

De map “src/main/java”.

- De package “fresneltowiki” bevat de Javacode om het Fresnelmodel te vertalen naar SMW-syntax.
- De package “jena” bevat de Java code om het Fresnelmodel om te zetten naar een Jena-ontologie model.
- De package “main” bevat de Java code voor de Fresnel Forms plug-in.

- De package “ontology” bevat de Java code om een bronontologie-model op te zetten.
- De package “utils” bevat de Java code voor huishoudelijke taken.
- De package “view” bevat de Java code voor de presentatie van de plug-in.
- De package “vocabulary” bevat de Javacode die de Fresnel en OWF-ontologie-syntax definiëren.

De map “src/test/java”. De package “fresnelcardinalitytest” bevat de unittesten voor de implementatie van de min. en max. cardinaliteit.

- De package “fresnelequalitytest” bevat de unittesten voor de object equality testen
- De package “fresnelheuristicsort” bevat de unittesten om de heuristische sorteermethode van de lens te testen.
- De package “fresnelimporttest” bevat de unittesten om de fresnel-importfunctionaliteit te testen.
- De package “fresneltowiki” bevat de unittesten om de Fresnelexport naar Semantic Media Wiki syntax te testen.
- De package “jena” bevat de unittesten om de omzetting van het Fresnelmodel naar het Jenamodel te testen.

Het functionele ontwerp

Het Fresnel Formsproject is opgezet volgens het Agile ontwikkelproces.De wensen van de opdrachtgever werden verzameld op de issue-lijst op GitHub, deze lijst functioneerde als de to-do-lijst. De issues werden opgelost in sprints van twee weken.

Voor de issues die nieuwe functionaliteit toevoegden aan het project zijn functionele en technische oplossingen bedacht die beschreven zijn in ontwerpdocumenten.

Hieronder worden de ontwerpbeslissingen opgesomd met een korte omschrijving van het probleem en de oplossing. De gedetailleerde beschrijvingen staan in het ontwerpdocumenten en zijn opgenomen in de bijlage. De volgorde van de onderwerpen komt globaal overeen met de projectvolgorde zoals beschreven in het procesverslag.

Tabel 2. De procesverslagonderwerpen met de ontwerpdocumenten.

	Onderwerp	**Ontwerpdocumenten**
1	Omzetten OWF-PHP naar Java.	Formatting Props Fresnel2Wiki Syntaxis van Infoboxes CSS van Fresnel naar wiki code
2	Opzetten en uitbreiden van de Fresnelontologie.	Fresnelgeneratie Fresneluitbreiding met OWF_Style properties. Overzicht fresnel2wiki-functionaliteit.
3	Implementeren van de uitbreidingen.	Prioritering van Property Formats Overzicht geïmplementeerde fresnel2wiki-functionaliteitIntegratie van Namespaces Cardinaliteit in Fresnel Forms DBPedia in FresnelForms

4	Export naar RDF.	RDF export Fresnel2Wiki
5	De innovatieve verbetering.	Heuristic based ranking of properties

Formatting Props Fresnel2Wiki. De eindstap om met Fresnel Forms een informatiesysteem op een wiki te maken is het converteren van een ontologie in Fresnel naar wikicode in XML. Dit wordt in de tool gedaan door de fresneltowiki package bestaande uit de klassen Fresnel2wiki, Article en de enumeration SMWType. Het skelet van Fresnel2wiki is ruwweg overgenomen van de PHP code van OWF.

De inhoud is toch aanzienlijk anders, mede door het gebruik van het Jena Framework in Fresnel Forms. Bijlage I geeft uitleg over hoe Fresnel2wiki de formatting properties (properties met stijlkenmerken voor semantische properties) ophaalt met Jena.

Syntaxis van Infoboxes. Het doel van bijlage II is om een syntaxis te definiëren waarmee SMW-infoboxes vanuit de Protégé plug-in 'Fresnel Forms' eenvoudig en flexibel gevuld en vormgegeven kunnen worden.

In de Protégé plug-in 'Fresnel Forms' is een infobox één van de doelen om semantische gegevens weer te kunnen geven binnen Semantic MediaWiki. Een infobox is een MediaWiki-term voor een tabel met een vast formaat welke toegevoegd wordt aan de rechterkant van een pagina. Deze tabel heeft als doel om belangrijke feiten en statistieken gemeenschappelijk met gerelateerde artikelen weer te geven.

Vanuit FresnelForms wordt voor elke Fresnellens een SMW infobox gegenereerd. Hierbij geldt de lens zelf als selectiemechanisme voor de properties welke getoond moeten worden. De Fresnel Formats bepalen hoe de properties getoond dienen te worden.

Eenzelfde infobox kan in SMW echter op veel manieren worden geimplementeerd met verschillende combinaties van Mediawiki, HTML & CSS syntaxes.

In bijlage II wordt bekeken welke infobox-syntaxis past bij de manier waarop Fresnel Forms, met Fresnel als tussenlaag, de presentatiegegevens opslaat en laat weergeven.

CSS van Fresnel naar wiki code. ABI Team 30's Fresnel Forms plug-in zal ontologieën inclusief aangepaste stijlinformatie omzetten van Fresnel naar wiki-pagina's, met de nadruk op infoboxen. Daarom is het belangrijk om te weten hoe stijlinformatie vertaald kan worden van Fresnel naar wikicode. Wiki accepteert HTML en CSS opmaak, bijlage III concentreeert zich op CSS.

RDF export Fresnel2Wiki. Een belangrijke requirement voor Fresnel Forms is dat het mogelijk is om van de naar een wiki geüploade ontologie de RDF triples te exporteren met de Semantic MediaWiki (SMW) special page special:ExportRDF voor RDF export[41] (Semantic MediaWiki). Bijlage XI geeft een beknopte uitleg van het proces.

Prioritering van Property Formats. Bij het toepassen van Fresnel Formats op Properties welke geselecteerd zijn door Fresnellenzen kan het zijn dat er meerdere Fresnel Formats geldig zijn voor een betreffende Property voor die lens. Deze Fresnel Formats kunnen echter

[41] `https://semantic-mediawiki.org/wiki/Help:RDF_export`

tegenstrijdige presentatie-informatie bevatten waardoor er een noodzaak is tot het eenduidig selecteren van presentatie-informatie.

Om de presentatie-informatie eenduidig te kunnen bepalen uit een collectie van geldige formaten, wordt hiervoor in dit document een algoritme ontworpen.

Bijlage VII geeft uitleg over het algoritme voor het prioriteren van Fresnel Property Formats.

Fresnel generatie. Bijlage IV analyseert de gegenereerde Fresnelcode en doet een alternatief voorstel.

We gebruiken Jfresnel als ‘gouden standaard’ en laten zien dat het alternatief geparset kan worden door Jfresnel.

Fresnel uitbreiding met OWF_Style properties. Bijlage V is geschreven om te onderzoeken hoe de Fresnelontologie uitgebreid kan worden.

Daartoe wordt in kaart gebracht welke informatie de wiki moet tonen. Daarna wordt bekeken of deze informatie al in de Fresnelontologie bestaat en hoe deze dan gebruikt kan worden.

Bestaat de informatie niet dan wordt bekeken hoe de Fresnelontologie uitgebreid kan worden zodat de Fresnel-to-wiki-conversie alsnog de informatie kan tonen op de wiki.

Overzicht fresnel2wiki functionaliteit. Bijlage VI geeft een high level overzicht van de fresnel2wiki-functionaliteiten van de Fresnel Forms plug-in. Daarnaast wordt een overzicht gepresenteerd van de uitbreidingen op de Fresnelontologie.

Integratie van Namespaces. Bijlage VIII legt uit hoe we namespaces van elementen uit de bronontologie kunnen integreren in Fresnel Forms. Het is hierbij uitdrukkelijk niet de bedoeling om het huidige ontwerp in zijn geheel te herzien; slechts de delen die van belang zijn voor het correct integreren van URI namespaces worden aangepakt, mede om de tijdsbesteding voor deze taak niet uit de hand te laten lopen.

Cardinaliteit in Fresnel Forms. De cardinaliteitrestricties in de bronontologie dienen geïmplementeerd te worden in Fresnel Forms. De min-en-maxcardinaliteiten met waarden voor 0 en 1 dienen vertaald te worden naar de Semantic Wiki Forms eigenschappen “isList” en “isMandatory”. Bijlage IX beschrijft de analyse, de oplossing en de implementatie in Fresnel Forms.

DBPedia in Fresnel Forms. Om de vraag van de opdrachtgever te kunnen beantwoorden of DBpedia ingelezen kon worden in de Fresnel Forms plug-in hebben we dit getest. Het bleek dat de DBpedia ontologie wel ingelezen kon worden maar ook dat de user interface onwerkbaar traag werd. In Bijlage X wordt de analyse en de oplossing voor dit probleem beschreven. Nadat de oplossing de user interface weer werkbaar maakte zijn nog enkele kleine aanpassingen doorgevoerd die het werken met zeer grote ontologieën vergemakkelijkt. Bijlage X geeft een overzicht van de oplossingen.

Heuristic based ranking of properties. Bijlage XII beschrijft hoe de “heuristic sort” methode zoals gepresenteerd door Paul (Paul, 2014) geïmplementeerd is in de Fresnel Forms plug-in. Paul evalueert verschillende sorteeralgoritmes om te bekijken welke het

beste lijkt op de sorteervolgorde die aangehouden wordt voor de infoboxes op wikipedia[42]. De “heuristic sort”methode komt hier het dichtste bij in de buurt.

8. Procesverslag

Dit hoofdstuk beschrijft het gevolgde proces en de stappen die genomen zijn tijdens het ontwikkelen van Fresnel Forms.

Het gevolgde proces

Bij het ontwikkelen van Fresnel Forms hebben we het volgende proces gevolgd.

Nagenoeg elke twee weken hadden we via Skype een SCRUM meeting met de opdrachtgever waarin de gemaakte oplevering, voortgang, en nieuwe eisen en wensen besproken werden. In het weekend voorafgaand aan de meeting werd hiervoor een nieuwe oplevering van Fresnel Forms gemaakt en gedeeld via GitHub.

De issues werden beheerd in GitHub en op basis van de prioriteit, zoals bepaald in samenspraak met de opdrachtgever, geïmplementeerd. De issues werden hierbij effectief gebruikt als discussieforum wanneer er onduidelijkheid was of actie vereist was van andere teamleden. Het toewijzen van een issue aan een persoon werd hierbij veelvuldig gebruikt om het issue bij de juiste persoon onder de aandacht te brengen.

Bij het verdelen van de issues werd afhankelijk van het functionele deelgebied gekeken welk teamlid het issue het best als eerst kon oppakken. In sommige gevallen werd eerst een haalbaarheidsonderzoek gedaan wanneer de impact of haalbaarheid van een issue in eerste instantie onduidelijk was.

Minstens eens per week hadden we een teambijeenkomst via Skype. Hierin werden de voortgang en de planning besproken evenals andere punten die op de agenda stonden die ervoor was opgesteld.

Bij het ontwikkelen werd naast de implementatie van een issue, unit tests ervoor gemaakt welke de implementatie testten. Voor grotere wijzigingen werd eerst een ontwerp gemaakt welke vastgelegd werd in een document (zie Bijlagen). Elke wijziging, ofwel als gevolg van de implementatie van een issue of als gevolg van refactoring, werd structureel ter review aangeboden aan een ander teamlid. Een issue werd slechts opgenomen in een oplevering wanneer deze gereviewed was en het commentaar daarop verwerkt was. Bij het reviewen werden zowel de implementatie als de unit tests en Checkstyle meldingen bekeken en beoordeeld. Ook werd hierbij handmatig steevast een korte functionele integratietest uitgevoerd in Protégé.

Alle wijzigingen werden in Subversion bijgehouden en waar nodig werd voor grotere wijzigingen in een branch gewerkt om de stabiliteit van de hoofdbranch (trunk) voor een volgende oplevering te waarborgen. Voor het maken van een oplevering was een procesbeschrijving gemaakt waarin het versienummer van Fresnel Forms structureel werd opgehoogd.

[42] http://www.wikipedia.org/

De genomen stappen

De stappen die genomen zijn tijdens het ontwikkelen van Fresnel Forms worden in dit hoofdstuk beschreven. De beslissingen die bij deze stappen genomen zijn en argumenten daarvoor komen hierin uitgebreid aan bod. De genomen stappen worden zoveel mogelijk in chronologische volgorde weergegeven en zijn onderverdeeld in de volgende thema's:

- De intake van de erfenis van Team 28
- De Fresnelontologie als basis
- Conversie van Fresnel naar Semantic MediaWiki
- Oplossingen voor structurele tekortkomingen
- 'Forms paper'-mijlpaal
- Innovatieve verbeteringen

De intake van de erfenis van ABI team 28. Vrijwel aan het begin van het project is één van ons reeds begonnen met de intake van de erfenis van ABI team 28, namelijk MDD Fresnel. Dit, om al in een vroeg stadium een beeld te krijgen van de ontwikkelomgeving en om dit voor de andere teamleden waar nodig duidelijk en reproduceerbaar vast te leggen zodat we er gezamenlijk gelijk mee aan de slag konden zonder veel oponthoud. Naar aanleiding van bevindingen en problemen die we hierbij tegenkwamen hebben we een aantal stappen genomen om de ontwikkelomgeving beter beheersbaar te maken en beter aan te laten sluiten op onze ontwikkelmethode zoals beschreven in het in fase 2 van het afstudeerproject geschreven plan van aanpak. Deze stappen worden in dit hoofdstuk beschreven.

Keuze voor een Buildtool. Team 28 had de ontwikkelomgeving gebouwd m.b.v. Eclipse en de standaard buildtools die daarin aanwezig zijn. Aangezien dit voor ons niet eenvoudig onderhoudbaar bleek en omdat Protégé zelf ook gebruik maakt van Apache Maven[43] als buildtool, zijn wij Apache Maven gaan gebruiken voor het bouwen van de software en de documentatie. Daarnaast integreert Apache Maven naadloos met het OSGi framework, zoals beschreven in hoofdstuk 4, waarin Fresnel Forms als plug-in acteert.

Oplossen van Checkstyle meldingen. Om een stabiele basis te hebben waarbij er op elk moment een snel en goed overzicht is op de kwaliteit van de software is vroeg in het project de keuze gemaakt om alle openstaande Checkstyle meldingen (volgens onze ontwikkelstrategie), zoals deze aanwezig waren in de code uit de svn repository, op te lossen alvorens echt te beginnen met ontwikkelen.

Het debuggen van code. Aangezien het niet triviaal was om de ontwikkelde plug-in te kunnen debuggen onder Protégé, hebben we gekozen om dit middels het gebruik van unit tests te doen. Het enige nadeel wat hieraan zat was dat we de user interface-componenten (in de view klasse) niet konden debuggen.

Logging. Naast het debuggen via unit tests hebben we gekozen om logging in te bouwen middels de log4j API. De log4j API was de meest voor de hand liggende keuze aangezien deze ook binnen Apache Jena en Protégé gebruikt wordt en de configuratiebestanden ervoor al aanwezig waren.

Het voordeel van logging boven het printen van meldingen naar de command line is dat geconfigureerd kan worden welke meldingen getoond moeten worden en welke niet. Zo kan

[43] https://maven.apache.org/

tijdens normale omstandigheden geconfigureerd worden dat alleen fouten en waarschuwingen getoond moeten worden en kan voor het debuggen van code tijdens de uitvoer in Protégé toch gebruik gemaakt worden van meldingen op debug niveau.

De Fresnelontologie als basis. Bij de aanvang van de ontwikkeling en nadat de ontwikkelomgeving beheersbaar was opgezet, hebben we ervoor gekozen om verder te bouwen op de functionele basis die ABI team 28 gelegd had. Dit, om zoveel mogelijk gebruik te kunnen maken werk wat reeds verricht was en de resultaten ervan. Hierbij is de door MDD Fresnel resulterende Fresnelontologie als generiek uitgangspunt gebruikt voor het genereren van de specifieke Semantic MediaWikicode. Omdat uit één van de domeinanalyses bleek dat de verwerking van de Fresnelontologie sub-optimaal was en om een goed onderhoudbare basis hiervoor te hebben, zijn er een aantal stappen genomen welke in dit hoofdstuk worden beschreven.

Apache Jena als RDF API. Zoals onderbouwd wordt in 'Verbeteren door hergebruik - Domeinanalyse over het opslaan en inlezen van Fresnel ontologieën binnen de Protégé Fresnel Forms plug-in' (Mekkering, 2014), bleek het serialiseren en parsen van de Fresnelontologie sub-optimaal te zijn. Om dit probleem op te lossen is Apache Jena als meest geschikte API voor het genereren en inlezen van RDF gebruikt.

Het Jena Model als uitgangspunt voor verdere conversies. Nadat het parsen en serialiseren van de Fresnel RDF refactored was naar het gebruik van de Apache Jena API, hebben we besloten om het resulterende Apache Jena Model als uitgangspunt te gebruiken voor de conversie naar Semantic MediaWiki code. De belangrijkste argumenten hiervoor waren:

- Apache Jena is een generieke API voor het raadplegen van een RDF-graaf, dus alle RDF-constructies worden ondersteund, ook degene die we nog niet voorzien hebben.
- Doordat Apache Jena alle RDF-constructies ondersteunt, kunnen we deze ook gebruiken om Semantic MediaWiki conversies te maken op basis van handmatig aangepaste Fresnelontologieen, los van de mogelijkheden van de rest van de plug-in.
- Doordat de SMW-conversie op deze manier volledig ontkoppeld is van de rest van de plug-in, kunnen verschillende ontwikkelaars, los van elkaar, op verschillende vlakken werken aan ondersteuning voor één probleem. Zo kan de generatie van SMW-code voor een specifieke RDF-constructie tegelijkertijd plaatsvinden met ontwikkeling van de GUI uitbreidingen daarvoor.

Een utility-klasse voor het interpreteren van de Fresnelsemantiek. Om de conversie van de (Fresnel) RDF-graaf naar SMW-code te ontlasten van Fresnelsemantiek hebben we besloten om een utility-klasse te maken, specifiek voor het interpreteren van de Fresnelsemantiek van de RDF-graaf. De argumenten hiervoor waren:

- De rolverdeling binnen het team resulteert in het feit dat verschillende ontwikkelaars expert zijn op het gebied van Fresnel en Semantic MediaWiki.
- Door deze contructie hoeft de Ontology Guru (hoofdstuk 9) geen diepgaande kennis te hebben van Semantic MediaWiki
- De Semantic MediaWiki Guru (hoofdstuk 9) hoeft geen diepgaande kennis te hebben van Fresnel.

Voor het genereren van SMW-code wordt nu aan de utility-klasse gevraagd om de (meest specifieke) eigenschappen van properties op lenzen volgens de Fresnelsemantiek. Het hoeft zich dus zelf niet meer bezig te houden met de semantische complexiteit van Fresnel.

Conversie van Fresnel naar Semantic MediaWiki. Nadat we een onderhoudbare basis hadden voor het verwerken en raadplegen van de Fresnel ontologie hebben we ons bezig gehouden met het functionele hoofddoel van de opdracht, namelijk het genereren van de Semantic MediaWiki code als uitgangspunt voor de user interface. De stappen die hiervoor genomen zijn, worden in dit hoofdstuk beschreven.

Porteren van OWF PHP naar Java. Om snel en beheersbaar functionaliteit van de OWF-plug-in binnen Fresnel Forms te krijgen, hebben we besloten om de reeds bestaande PHP-code zoveel mogelijk te porteren naar Java. Dit heeft de volgende voordelen:

- Doordat het meeste denkwerk al is gedaan, is er snel resultaat te halen.
- De functionaliteit zal in één keer op nagenoeg hetzelfde niveau zijn als de eerdere oplossing.
- Een review van de code is eenvoudig aangezien de PHP-code er naast gehouden kan worden en de verschillen eenvoudig inzichtelijk kunnen worden gemaakt.

Het beperken van SPARQL. Nadat het porteren van de PHP-code naar Java had plaatsgevonden en de utility-klasse voor het interpreteren van de Fresnel Semantiek klaar was, hebben we besloten om alle resterende SPARQL queries te refactoren naar het direct raadplegen van het Apache Jena Model (grotendeels via de utility-klasse). De nadelen hiervan zijn:

- De complexiteit van een specifieke SPARQL query zal expliciet geprogrammeerd moeten worden.

De voordelen hiervan zijn:

- Er is slechts één techniek nodig voor het bevragen van de Fresnelontologie waardoor onderhoud aan de code eenvoudiger wordt.
- Het direct bevragen van het Apache Jena Model is eenvoudiger te debuggen wanneer dat nodig is.
- Het direct bevragen van het Apache Jena Model is sneller dan het (laten) parsen van de SPARQL query, het voorbereiden en uitvoeren van de query en het interpreteren van de geleverde resultaten.
- De complexiteit van de aanwezige SPARQL queries was niet erg groot en eenvoudig om te zetten naar een directe bevraging van het Apache Jena Model.

Apache Commons Lang voor het escapen van Strings voor gebruik in XML. Om te garanderen dat in alle gevallen well-formed XML geproduceerd wordt, hebben we de keuze gemaakt uit:

- Het opzetten van een XML DOM om vervolgens een geldig XML-document uit te (laten) genereren.
- Het blijven samenstellen van XML strings middels StringBuilder en door escape-functies van Apache Commons Lang[44] te laten verwerken tot geldige XML.

We hebben in Fresnel Forms voor het gebruik van de StringEscapeUtils.escapeXml10-functie gekozen omdat:

[44] https://commons.apache.org/proper/commons-lang/

- Door de portering van PHP- naar Javacode alles reeds in Strings wordt opgebouwd.
- Het toevoegen van een DOM de schaalbaarheid van de plug-in voor wat betreft het geheugengebruik niet ten goede zou komen.
- Een soortgelijke functie toch al nodig was om door gebruikers ingevoerde gegevens ook op een correcte manier in SMW-code om te zetten. Er is dus maar één functie nodig om van een arbitraire string geldige XML content te maken.
- Apache Commons Lang een veel gebruikte Library is voor het escapen van arbitraire tekst voor verschillende syntaxes.

De syntaxis van infoboxes. Zoals in bijlage II wordt verantwoord, hebben we voor het genereren van infoboxes gekozen om hiervoor XHTML div-elementen te gebruiken waarvan de structuur overeenkomt met die van het Abstract Box Model zoals in Fresnel gebruikt wordt.

Oplossingen voor structurele tekortkomingen. Gaandeweg het verloop van het project, kwamen we een aantal structurele tekortkomingen tegen in de bestaande implementatie. De stappen die genomen zijn om deze op te lossen, worden in dit hoofdstuk beschreven.

De volgorde van properties. De volgorde van properties op een lens bleek niet in alle gevallen bewaard te kunnen worden. Met name het verbergen van properties zorgde hierbij voor problemen omdat deze in een aparte lijst werden bijgehouden, disjunct met de lijst van properties die getoond dienen te worden.

Om de volgorde van properties op een lens in alle gevallen te bewaren, dus ook voor hidden properties, hebben we gebruik gemaakt van de mogelijkheid in Fresnel om hidden properties zowel in fresnel:showProperties als in fresnel:hideProperties te vermelden. In fresnel:showProperties staan dan alle properties op volgorde terwijl de properties, welke vermeld staan in fresnel:hideProperties, verborgen worden weergegeven op de lens (en derhalve niet doorgegeven aan SMW).

Gebruik van namespaces voor zowel Fresnel als SMW. De MDD Fresnel plug-in hield geen rekening met de namespace van properties, klassen, lenzen en formats. Hierdoor ontstonden de volgende problemen:

- De Fresnelontologie was eigenlijk niet van toepassing op de elementen uit de bronontologie.
- Gelijkgenoemde properties of klassen werden vertaald naar dezelfde entiteit, waardoor geen onderscheid te maken was tussen bijv. `foaf:name` en `myown:name`.
- Ook op de GUI was een verschil in namespace niet te zien.

We hebben hiervoor in alle lagen van Fresnel Forms de URI's van entiteiten (welke ook de namespace bevat) als uitgangspunt gebruikt omdat die een entiteit identificeren. Ook in SMW hebben we een gecodeerde vorm van de URI gebruikt om een entiteit te benoemen zodat ook op het target platform alle entiteiten uniek benoemd zijn.

'Forms paper'-mijlpaal. Ongeveer één maand na de start van de implementatiefase van het project kreeg opdrachtgever dr. Rutledge de uitnodiging om een vervolg te schrijven op een eerder artikel waarin hij OWF introduceerde (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013). In dit vervolg wilde Rutledge onze verwachte plug-in Fresnel Forms introduceren. Vanwege de Agile-ontwikkelmethode was het geen probleem om de focus in het project te verleggen, dit werd vormgegeven door de 'Forms paper'-mijlpaal, welke

hiervoor was gedefinieerd. Terwijl het algemene doel van het project hetzelfde bleef, namelijk het efficiënt genereren van user interfaces voor Semantic Web-ontologieën, kregen sommige requirements, specifiek van belang voor de Forms paper, een hogere prioriteit en ook nieuwe requirements werden hiervoor geïntroduceerd. Één zo'n requirement was het het inlezen van de DBpedia-ontologie als voorbeeld van een zeer grote ontologie, om de schaalbaarheid van de oplossing aan te tonen. We hebben dit mogelijk gemaakt door de schermafhandeling van de plug-in efficiënter te laten verlopen. De invulling van deze requirement is aangetoond door de DBpedia-ontologie daadwerkelijk te laten verwerken door Fresnel Forms.

Innovatieve verbeteringen. Ter ondersteuning van de onderzoekscontext hebben we in Fresnel Forms de innovatieve, door Paul ontworpen heuristische methode voor het bepalen van de volgorde van properties (Paul, 2014) geïmplementeerd.

Volgens Paul, levert een property-volgorde, volgens de door hem voorgestelde heuristische methode, infoboxes op die beter voldoen aan die van Wikipedia dan bijv. een alfabetische of arbitraire sortering (Paul, 2014). Wij hebben daarom gekozen om deze methode te implementeren in Fresnel Forms om nog eenvoudiger en sneller een goede user interface-definitie te kunnen leveren.

9. Teamreflectie

In dit hoofdstuk wordt het project zelf en de gekozen aanpak geëvalueerd.

Het project zelf

Voor de planning van ons afstudeerproject hebben we gekozen om de voorbeeldplanning, zoals die op studienet staat één op één over te nemen. Dit was mogelijk omdat we gezamenlijk een besteedbare tijd van vijftien uren per week waren overeengekomen. Deze planning hebben we precies gevolgd en daardoor zijn we niet uitgelopen. We zijn van mening dat, met name voor de implementatiefase, de Agile-methodiek die we gevolgd hebben hier ook in positieve zin toe heeft bijgedragen.

De overeengekomen vijftien uren studiedruk hebben we allen iets overschreden.

Fase 3b, de onderzoekscontext, welke nu eigenlijk achteraan de implementatiefase van start gaat, had naar ons idee beter gelijktijdig met de domeinanalyse kunnen aanvangen. Zo is er nog ruimte om de ideeën die daarin opgedaan worden, eventueel te verwerken in de implementatie.

De samenwerking met de opdrachtgever is ons goed bevallen. Opdrachtgever dr. Rutledge was net als wijzelf altijd erg enthousiast en droeg daardoor zeker in positieve zin bij aan onze motivatie. Ook gedurende een sprint, waar veel op GitHub gediscussieerd werd, nam hij vaak de tijd en de ruimte om ook in de discussies deel te nemen. Mede daardoor is het project volgens ons geslaagd.

Het samenwerken op afstand is ons ook goed bevallen, mede door de tools en technieken waarmee dit mogelijk wordt gemaakt. We hadden wekelijks een vaste bijeenkomst gepland en eens per twee weken een SCRUM sessie met de opdrachtgever. Tegen het einde van het project hadden we bijna dagelijks een bijeenkomst om de voortgang voor de scriptie te bespreken opdat de deadline gehaald zou worden.

De midterm meetings waren een welkome aanvulling. Naast het feit dat we elkaar dan eens zagen (we werkten immers slechts op afstand samen) waren de bijeenkomsten nuttig om

tips van anderen mee te krijgen. Omdat ons project volgens planning verliep, was voor ons het wellicht licht sturende doel minder van toepassing.

Rolverdeling

Bij de rolverdeling hebben we het project in functionele deelgebieden verdeeld en aan elk van deze deelgebieden een expert, door ons “Guru” genoemd, gekoppeld. De verdeling was als volgt:

Ontology Guru: Teun - Aangezien Teun de cursus Semantic Web had gevolgd was hij de aangewezen persoon om in de loop van dit project als expert op het gebied van het Semantic Web door het leven te gaan. Teun was primair verantwoordelijk voor alles wat met Semantic Web ontologieen te maken had, inclusief de uitbreidingen op de Fresnel ontologie.

Wiki Guru: Joop - Joop was erg geïnteresseerd in kennisbeheersysteem achter MediaWiki en is derhalve opgestaan als Wiki Guru. Joop was primair verantwoordelijk voor het ontwerp van de conversie van Fresnel naar Semantic MediaWiki code.

Protégé Guru: Alex - Aangezien Alex professioneel gezien veel ervaring had met het ontwikkelen van software, hoewel niet specifiek Java, was hij de aangewezen persoon als brug tussen de focusgebieden van de Ontology Guru en de Wiki Guru. Alex was derhalve primair verantwoordelijk om de ontologiekant en de wiki-kant vloeiend op elkaar aan te laten sluiten.

Bijtaken

Naast de primaire verantwoordelijkheden waren er ook een aantal bijtaken. Zoals voor alles hebben we ook hiervoor gezamenlijk een primaire verantwoordelijke bepaald.

- Joop zorgde voor het algemene overzicht, de academische focus en het structureren van de vergaderingen.
- Teun zorgde voor bijkomende ontologie-gerichte zaken zoals een onderzoek naar Fresnel browsers en de ontwikkeling van een onafhankelijke tool om de gegenereerde Fresnel ontologie te analyseren en valideren. Ook enkele mooie oplossingen zoals aanpassing van de GUI voor de schaalbaarheid (wikipedia-ontologie) kwamen van zijn hand.
- Alex zorgde voor de techniek zoals het opzetten van de ontwikkelomgeving, installeren van de gezamenlijke testserver, en beheer van het subversion versiebeheersysteem. Hij had wellicht het beste inzicht in de algehele problematiek.

Naast gedefinieerde bijtaken hebben we ieder ook de ruimte genomen en gekregen om eens wat taken van een ander op te pakken. Mede hierdoor kreeg ieder van ons een goed overzicht over het geheel en werd de kennis goed gedeeld.

Als team hebben we waarschijnlijk meer voor elkaar gekregen dan wat we elk afzonderlijk hadden kunnen doen. Wanneer één van ons drie maal zoveel tijd gekregen had om het project uit te voeren, dan was het resultaat waarschijnlijk minder geweest dan nu het geval is.

Gekozen aanpak

Gekozen tools. Door het feit dat één van ons Subversion als versiebeheer omgeving goed kende hebben we deze tool efficiënt kunnen benutten. Zo hebben we goed gebruik gemaakt van de mogelijkheid om branches en tags aan te maken waar dat van toepassing was. Hoe

aantrekkelijk het gebruik van Git ook leek bij de start van dit project, een keuze hiervoor had dit proces toch wel weer wat complexer gemaakt en had dan wellicht meer tijd in beslag genomen om het goed te doen.

GitHub bleek een heel fijn platform om de issues bij te houden in samenspraak met de opdrachtgever. Er zijn veel mogelijkheden om de issues van labels te voorzien waardoor prioriteiten volgens de door ons gekozen MoSCoW-methode (van Vliet, 2008) goed vorm gegeven konden worden. Daarnaast bleek het een zeer effectief discussieplatform en is daarvoor ook veel gebruikt. Het gebruik van GitHub vereist echter wel de nodige discipline om de issues goed bij te houden. Zo was er na het invoegen van de 'Forms paper'-mijlpaal onduidelijkheid over welke issues nu als eerst moesten worden opgepakt, waardoor de focus niet bij iedereen in het team dezelfde was.

Ook zijn we benieuwd naar de mogelijkheden die een verdergaande integratie van versiebeheer (met Git) en issue-management (in GitHub) te bieden heeft en wellicht is het voor een volgend team een goed idee om, alvorens te beginnen, een halve week de tijd te nemen om deze combinatie te proberen.

Google Docs en Google Drive waren voor ons erg nuttige tools om mee samen te werken naar een gezamenlijk eindresultaat. Zo hebben we Google Documents gebruikt voor de basis van deze scriptie, Google Drive voor het delen van mappen en bestanden en Google Presentaties voor het gezamenlijk samenstellen van de presentatie.

Gedurende de loop van het project bleek dat er een plug-in voor Eclipse beschikbaar was welke eenvoudig UML-diagrammen kan destilleren uit de code die gemaakt is. Deze hebben we dan ook veel gebruikt waardoor Dia als ontwerptool wat meer naar de achtergrond schoof.

Gekozen ontwikkelmethoden. De Agile ontwikkelmethode met sprints van twee weken is ons goed bevallen. Er was genoeg tijd om naast triviale issues ook wat meer structurele zaken aan te pakken en de frequentie was hoog genoeg om zo af en toe wat bij te sturen.

Checkstyle heeft ons geholpen om de complexiteit van de implementatie laag te houden en tips gegeven over hoe de implementatie nog beter kon. Door in het begin van het project, bij de intake van code van team 28, alle meldingen reeds op te lossen was er altijd een goed overzicht van welke kwalitatieve verbeterpunten er resteerden.

Hoewel de unit tests in onze ogen wellicht nog iets verder uitgewerkt hadden kunnen worden hebben we de code coverage weten te verbeteren van nul naar 62,8%, waarvan de nl.ou.fresnelforms.jena, nl.ou.fresnelforms.fresnel2wiki, nl.ou.fresnelforms.vocabulary en nl.ou.fresnelforms.utils klassen een code coverage hebben van 90% en hoger. De nl.ou.fresnelforms.view-klasse was met 14,7% een negatieve uitschieter, maar dit package bevat voornamelijk user interface-elementen welke erg lastig zijn te dekken door unit tests.

Kijkend naar deze resultaten, vinden we dat de kwaliteit van de code hierdoor goed te noemen is.

De unit tests zijn ook deels ingezet als integratietesten voor met name de testen voor de wikicode. Hierbij werd vanuit een vaste bronontologie en bijbehorende Fresnel stylesheet de gegenereerde wikicode vergeleken met een verwachting. Hierdoor werd bijna de gehele functionele keten doorlopen, met uitzondering van de user interface (view package).

Het consequent reviewen van elkaars code heeft ook zeker bijgedragen aan een goed eindresultaat. De reviews werden serieus genomen en er tegelijkertijd gebruikt om inzicht te krijgen in elkaars werk. Daarnaast droeg het positief bij aan het gezamenlijk eigenaarschap.

Door het ontbreken van use case scenarios hebben er met name in het begin van het traject weinig user tests plaatsgevonden, dus hier had wellicht wat meer aandacht aan gegeven kunnen worden. Voor de 'Forms paper'-mijlpaal is er met name door de opdrachtgever met de plug-in gewerkt waardoor er genoeg vertrouwen was dat het juiste ontwikkeld was.

Toch is er een bug geweest die pas gevonden werd bij het samenstellen van de demo van de presentatie. Deze bug was niet lastig op te lossen, maar door beter te testen tegen een verse wiki-omgeving had deze wellicht eerder gevonden kunnen worden.

10. Persoonlijke ervaringen en leermomenten

11. Conclusies en aanbevelingen

Het eindresultaat

Fresnel Forms heeft alle voor opdrachtgever van belang zijnde 'Must'-requirements geïmplementeerd en een aantal Coulds en Shoulds, waaronder de implementatie van de heuristische sortering, zoals voorgesteld door Paul (Paul, 2014).

De schaalbaarheid van Fresnel Forms heeft zich bewezen door het genereren van een Semantic MediaWiki user interface voor één van de grootste ontologieën ter wereld, namelijk de DBpedia-ontologie (Lehmann, et al., 2014).

Kijkend naar de bevindingen van Checkstyle, de code coverage van de unit tests en de beheersbaarheid en reproduceerbaarheid van het bouwproces is de plug-in kwalitatief verbeterd, wat het voor een eventueel volgend ABI team of anderen eenvoudig mogelijk moet maken om de ontwikkeling van Fresnel Forms voort te zetten.

De platformonafhankelijkheid is op zowel een raspberry pi met 32-bits Java 7 SDK onder Linux als een modern Intel platform met 64-bits Java 8 SDK onder Windows aangetoond. Hierdoor is de plug-in op de meeste platformen probleemloos te ontwikkelen en gebruiken en dus voor een breed publiek bruikbaar.

Voor wat betreft het resultaat van het eindproduct is dit ABI project dus zeker geslaagd.

De weg er naartoe

De Agile-ontwikkelmethode is ons als Team 30 goed bevallen. Doordat verandering omarmd werd, konden we ons in samenspraak met de opdrachtgever focussen op de requirements die van belang waren voor de paper submission 'From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies' (Rutledge, From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies, 2015 verwacht) waarin wij als co-auteurs deelnamen.

Het eindresultaat heeft er ook niet onder geleden en wellicht was hierdoor zelfs de focus voor een goed en bruikbaar eindresultaat wel extra aanwezig.

Doordat we als team samenwerkten konden we de taken goed verdelen, toegespitst op de functionele deelgebieden waar ieder van ons een specifieke expert-rol (guru) in vervulde.

Doordat we soms wat taken van een ander overnamen werd bovendien de kennis goed gedeeld waardoor we gezamenlijk en ieder afzonderlijk een compleet beeld kregen van alle aspecten van het project.

De weg naar het eindresultaat is wat ons betreft dus ook zeker geslaagd.

Academische aspecten

Het afstudeerproject vond plaats in het kader van onderzoek van een staflid van de faculteit Management, Science & Technology van de Open Universiteit dr. Rutledge als opdrachtgever. Het was verhelderend en leerzaam om een academisch onderzoekproces van dichtbij mee te hebben kunnen maken, ook omdat academische aspecten in de rest van de opleiding volgens ons meer op de achtergrond staan.

In het begin van het project richtten wij ons bijvoorbeeld op de functionaliteit van de tool als doel op zich. Gaandeweg in het project, na geregelde interactie met de opdrachtgever en inlezen in de context, begonnen wij te begrijpen dat de implementatie van de tool slechts onderdeel was van een groter doel, namelijk het bekrachtigen van Rutledge's onderzoek naar systemen voor het invoeren en verkennen van gegevens voor en met behulp van technieken van het Semantic Web. Tijdens het project vergrootten wij, naast onze ervaring met het verrichten van elementair onderzoek (met name in de fasen domeinanalyse en onderzoekcontext), ook onze inzichten dat er op verschillende gebieden al veel kennis bestaat of nog ontwikkeld wordt, waar op voort te bouwen is. Zo kregen we gevoel voor welke functionaliteiten of aspecten van belang waren en waarom.

Wij hopen dat wij met de implementatie van Fresnel Forms, als *proof of concept* in het verwachte artikel van Rutledge (Rutledge, From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies, 2015 verwacht) een kleine bijdrage hebben kunnen leveren aan de acceptatie van Rutledge's stelling dat de workflow van een semantische wiki efficiënter kan. Wat ons betreft is de kennismaking met academische aspecten in dit project ook geslaagd.

Aanbevelingen voor de toekomst

Ondanks het feit dat dit afstudeerproject zowel in technische als persoonlijke zin zeer geslaagd was, hebben we wel een aantal aanbevelingen voor de toekomst.

Uitzoeken hoe te debuggen onder Protégé. Om de ontwikkeling in de toekomst nog doeltreffender te kunnen uitvoeren zou automatische deployment en het debuggen van Fresnel Forms in Protégé een welkome aanvulling zijn. Aanpassingen aan de code zouden dan eenvoudiger vanuit eclipse gedebugged kunnen worden door vanuit eclipse de plugin uit te voeren binnen Protégé.

Versiebeheeromgeving omzetten naar Git(Hub). Door het versiebeheer om te zetten naar Git en de reeds aanwezige GitHub repository hiervoor als master repository in te richten zou het versiebeheer nog beter aansluiten bij het issue management. Hierdoor kunnen commits rechtreeks aan issues gekoppeld worden, en omgekeerd. Daarnaast wordt het definieren van Release duidelijker omdat GitHub weet welke commits er tussen twee releases in een bepaalde branch zijn uitgevoerd.

Implementeren van meertaligheid. Gedurende de loop van het project, werd een wens voor ondersteuning van meertaligheid duidelijk. Er was echter niet genoeg tijd meer om deze wens te implementeren. Er is wel een beschrijving gemaakt in het desbetreffende issue over

hoe de implementatie ervan aangepakt kan worden. Een eventueel volgend ABI team zou dit mee kunnen nemen bij de implementatie.

Gebruik maken van de MediaWiki API om de Wiki pagina's te maken. De huidige implementatie van Fresnel Forms exporteert de gedefinieerde User Interface als een importbestand voor MediaWiki. Dit importbestand zal dan door de gebruiker middels een Special page geimporteerd moeten worden voordat de resultaten ervan zichtbaar worden op de wiki.

Zoals uit het verslag van de onderzoekscontext blijkt, zou het gebruik van de MediaWiki API de ontwikkeling van een User Interface bij een willekeurige ontologie kunnen versnellen doordat er minder gebruikersacties nodig zijn om het eindresultaat te kunnen beoordelen.

Het maken van een MediaWiki extensie voor meer aansluiting bij het Semantic Web.

Semantic MediaWiki is tot op een bepaalde hoogte equivalent met het Semantic Web. Zoals uit het verslag van de onderzoekscontext blijkt, is één van de grootste en meest problematische verschillen het verschil in typesysteem. Om dit op te lossen zou een Semantic MediaWiki extensie gemaakt kunnen worden om ondersteuning te bieden voor ten minste XML Schema datatypen zoals deze binnen het Semantic Web gebruikt worden.

Referenties

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlagen

Inhoudsopgave

1. Bijlage I: Formatting properties Fresnel2Wiki.....47
2. Bijlage II: Syntaxis van SMW infoboxes.....50
3. Bijlage III: CSS van Fresnel naar wikicode 57
4. Bijlage IV: Fresnel generatie voor het koppelen van formats aan lenzen en properties. 64
5. Bijlage V: Fresnel uitbreiding met OWF_Style properties.....72
6. Bijlage VI: Overzicht van de fresnel2wiki functionaliteit. 77
7. Bijlage VII: Prioritering van Property Formats 82
8. Bijlage VIII: Integratie van namespaces.....89
9. Bijlage IX: Cardinaliteit in Fresnel Forms. 98
10. Bijlage X: DBpedia in Fresnel Forms plugin. 105
11. Bijlage XI: RDF export Fresnel2Wiki.....112
12. Bijlage XII: Heuristic based ranking of properties. 118
13. Bijlage XIII: Fresnel Forms user interface 130
14. Bijlage XIV: Statustabel Fresnel Forms.....134
15. Bijlage XV: Kennisdeling Wiki 137
16. Bijlage XVI: Onderzoekscontext Alex.....151
17. Bijlage XVII: Onderzoekcontext MDD 163
18. Bijlage XVIII: De onderzoeks context van het ABI project Fresnel Forms 170

Bijlage I: Formatting properties Fresnel2Wiki

Joop van de Heijning
Open universiteit

Introductie

De laatste stap om met Fresnel Forms een informatiesysteem op een wiki te maken is het converteren van een ontologie in Fresnel naar wikicode in XML. Dit wordt in de tool gedaan door de fresneltowiki package bestaande uit de klassen Fresnel2wiki, Article en de enumeration SMWType. Het skelet van Fresnel2wiki is ruwweg overgenomen van de PHP code van OWF (Rutledge). De inhoud is toch aanzienlijk anders, mede door het gebruik van het Jena Framework in Fresnel Forms (The Apache Software Foundation). Dit verslag geeft uitleg over hoe Fresnel2wiki de formatting properties (properties met stijlkenmerken voor semantische properties) ophaalt met Jena. Motivatie voor bovenstaande keuzes wordt in hoofdstuk 8 Procesverslag van deze scriptie gegeven.

Afhankelijkheden

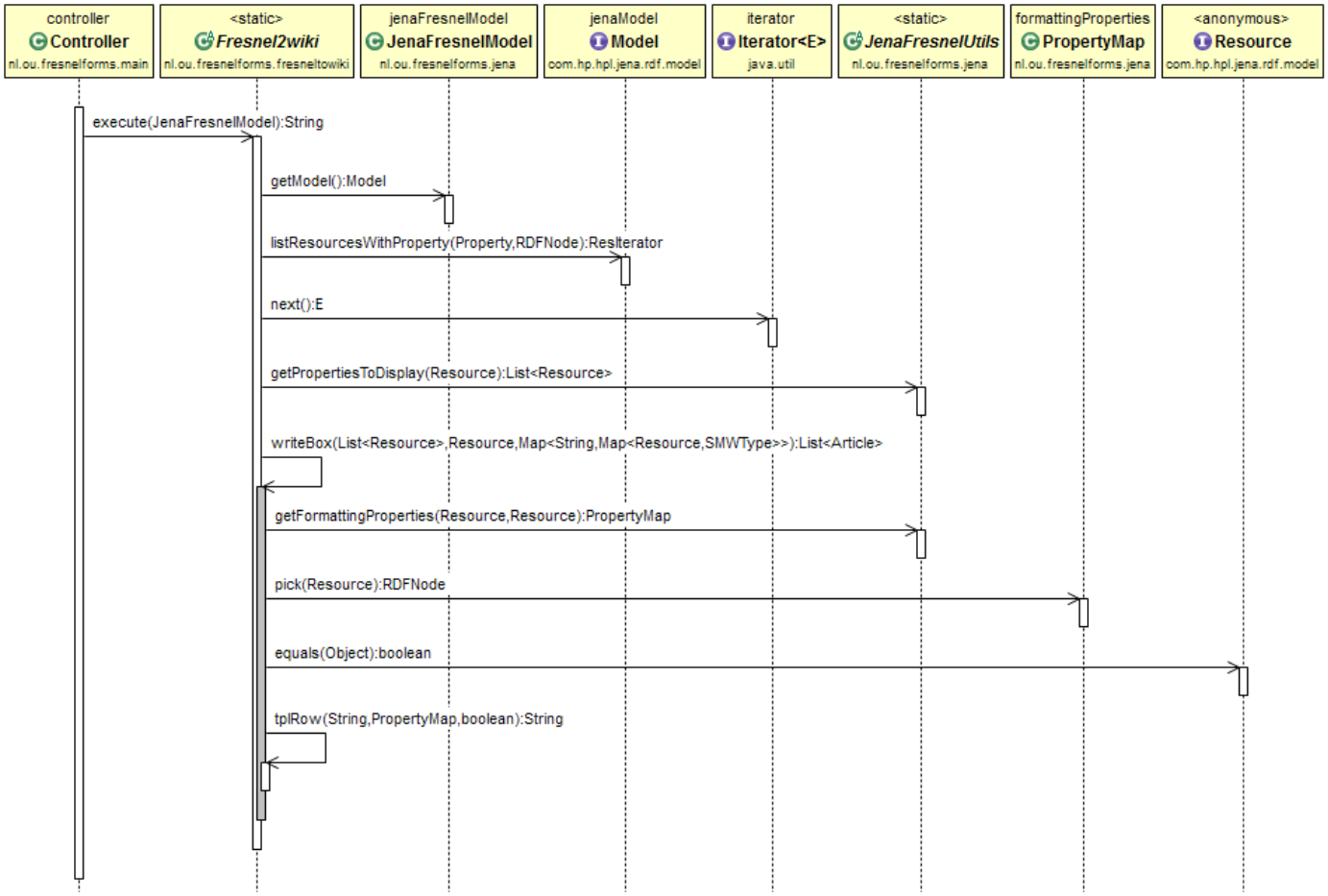
De controller krijgt de opdracht “save wiki” van de view en geeft deze door aan Fresnel2wiki. Fresnel2wiki maakt verder gebruik van de klasse JenaFresnelModel voor het ophalen van de Fresnel lenzen, van JenaFresnelUtils voor het ophalen van de (formatting) properties en van Article voor het tijdelijk opslaan van wiki-pagina’s. JenaFresnelUtils maakt gebruik van een PropertyMap voor het bewaren van formatting properties voor een semantische property.

Code

Methode execute(JenaFresnelModel) haalt eerst de lenzen op uit het Jena Fresnel model en loopt deze af. In deze lus wordt per lens de properties opgehaald (getPropertiesToDisplay()) en writeBox() aangeroepen. Binnen writeBox() worden de properties bij de lens afgelopen. Per property wordt een propertyMap met de formatting properties opgehaald bij de abstracte klasse JenaFresnelUtils. Uit de propertyMap wordt de desbetreffende formatting property opgehaald met methode pick(). Deze methode heeft als argument het soort formatting property dat je op die plek nodig hebt. Deze namen zijn te vinden in de klassen FRESNEL en OWF van de vocabulary package. Het volgende fragment code geeft een voorbeeld van het bepalen of een property het type image heeft:

```
boolean isImage =
   FRESNEL.IMAGE.equals(formattingProperties.pick(FRESNEL.VALUE));
```

De propertyMap wordt later in writeBox() ook doorgegeven aan hulpmethoden zoals tplRow(), die dezelfde methode gebruikt om andere formatting properties op te halen. Onderstaande figuur 1 geeft de stappen weer in een sequence map.



Figuur 1 **Fragment Fresnel Forms vereenvoudigd**

Referenties

Rutledge. (n.d.). *OWLwikiForms v0.2.1.zip*. Retrieved December 6, 2014, from OWL Wiki Forms v0.2 installation: http://is.cs.ou.nl/OWF/files/OWLwikiForms_v0.2.1.zip

The Apache Software Foundation. (n.d.). *Apache Jena - Home*. Retrieved May 2, 2015, from Apache Jena: https://jena.apache.org/

Bijlage II: Syntaxis van SMW infoboxes

Definitie van de te gebruiken syntaxis voor SMW infoboxes in Fresnel Forms.

Alex Mekkering

Revisiehistorie

Datum	Beschrijving
2015-02-10	Initiële versie
2015-02-15	Tabel met mogelijke oplossingen voor veel voorkomende stijlwensen
2015-02-20	Revisiehistorie bijgewerkt
2015-05-12	Uiteindelijke versie

Doel

Het doel van dit document is om een syntaxis te definiëren waarmee SMW infoboxes vanuit de Protégé plugin ‘Fresnel Forms’ eenvoudig en flexibel gevuld en vormgegeven kunnen worden.

Inleiding

In de Protégé plugin ‘Fresnel Forms’ is een infobox één van de doelen om semantische gegevens weer te kunnen geven binnen Semantic MediaWiki. Een infobox is een MediaWiki term voor een tabel met een vast formaat welke toegevoegd wordt aan de rechterkant van een pagina. Deze tabel heeft als doel om belangrijke feiten en statistieken gemeenschappelijk met gerelateerde artikelen weer te geven.

Vanuit Fresnel Forms wordt voor elke Fresnel lens een SMW infobox gegenereerd. Hierbij geldt de lens zelf als selectiemechanisme voor de properties welke getoond moeten worden. De Fresnel Formats bepalen hoe de properties getoond dienen te worden.

Voor het bepalen van het uiterlijk wordt in Fresnel gebruik gemaakt van de fresnel:resourceStyle, fresnel:propertyStyle, fresnel:labelStyle en fresnel:valueStyle properties. Bij deze properties kan de gebruiker CSS invullen welke vervolgens gebruikt kan worden om een Resource (infobox zelf), Property, Label of Value door de browser van styling te laten voorzien.

Eenzelfde infobox kan in SMW echter op veel manieren worden geïmplementeerd met verschillende combinaties van Mediawiki, HTML & CSS syntaxes.

In dit document wordt bekeken welke infobox syntaxis past bij de manier waarop Fresnel Forms, met Fresnel als tussenlaag, de presentatiegegevens opslaat en laat weergeven.

Achtergrond

Huidige syntaxis

De huidige syntaxis voor infoboxes, of eigenlijk die voor de templates voor infoboxes, is die van een MediaWiki table **Invalid source specified.** waarbij het wanneer een regel begint met:

- ‘{|’ – een tabel begint
- '|+’ – een tabel bijschrift definieert
- ‘|-‘ – een rij begint
- ‘!’ – een header cel beschrijft
- ‘|’ – een data cel beschrijft
- ‘|}’ – een tabel eindigt

Elke markup hiervan, met uitzondering van de beëindiging van de tabel, accepteert tevens XHTML attributen zoals bijvoorbeeld class=””, style=”” en colspan=””.

Deze syntax is het equivalent van een HTML table.

Hoewel een table syntaxis als deze de aangewezen kandidaat lijkt voor het weergeven van infoboxes, is het, net als bij een XHTML table, met CSS niet mogelijk om inhoud te verdelen over meerdere kolommen en/of rijen; hiervoor zijn de XHTML attributen colspan en rowspan vereist. Ook is in het bijzonder de breedte van de labels en values (dus de breedte van de 1^{e}

en 2e kolom) niet afzonderlijk in te stellen met CSS. Hierdoor wordt een gedeelte van de flexibiliteit die CSS biedt, teniet gedaan door de structuur die door een tabel wordt opgedrongen.

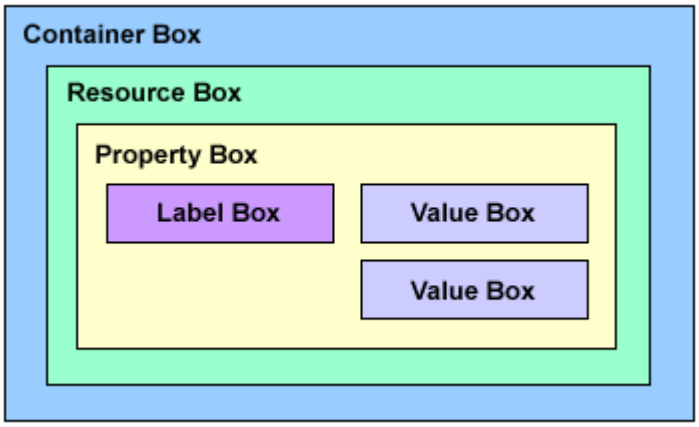
Wanneer dus gebruik gemaakt wordt van een tabel om de gegevens vorm te geven, is er dus vanuit de plugin gezien minder controle over de presentatie van gegevens dan op het eerste oog lijkt.

Abstract Box Model

Om een beeld te krijgen van de presentatiemogelijkheden van Fresnel Forms, kijken we naar het presentatiemodel wat eraan ten grondslag ligt, namelijk het Abstract Box Model.

Fresnel

In Fresnel (Bizer, Lee, & Pietriga, 2015) staat het volgende model centraal voor weergave van Properties welke geselecteerd zijn door een Lens:



Figuur 27: Fresnel Abstract Box Model

Hierbij geldt het volgende:

- De Container Box is geen onderdeel van de standaard Fresnel vocabulaire en is bedoeld om bijvoorbeeld de achtergrond van een gehele HTML pagina of SVG diagram vorm te geven. Vooralsnog biedt Fresnel Forms geen ondersteuning hiervoor.
- De Resource Box is het gebied dat alle properties van een enkele resource omvat en kan daarom gezien worden als het Fresnel equivalent van een Infobox.
- De Property Box is het gebied dat de gegevens van een enkele property omvat.
- De Label Box is het gebied dat het label van een enkele property omvat.
- De Value Box is het gebied dat een enkele waarde van een property omvat.

XHTML equivalent

Om hetzelfde resultaat in XHTML te bereiken, kan voor iedere box in het Abstract Box Model een div-element gebruikt worden om de volgende redenen.

- Het div element wordt als één van de weinige XHTML elementen standaard ondersteund door MediaWiki.

- Het div element is een XHTML block element waardoor de presentatie ervan volledig kan worden bepaald door CSS, dit in tegenstelling tot inline elementen waarvoor bijv. margins en padding geen invloed hebben.
- Ze kunnen genest worden, wat betekent dat label- en value-boxes binnen property boxes binnen een resource box gedefinieerd kunnen worden.
- Het div element voegt standaard geen presentatie informatie toe, anders dan het feit dat het standaard een block element is.

CSS in MediaWiki

Om de presentatie van gegevens vorm te geven wordt in MediaWiki, net als in Fresnel, ondersteuning geboden voor CSS. Deze ondersteuning wordt geboden middels stylesheets op verschillende niveaus en inline XHTML style-attributen.

Stylesheets

De Styleheets welke in MediaWiki standaard worden ondersteund, zijn als volgt op verschillende niveaus in te stellen:

- MediaWiki:Common.css – algemene stylesheet voor de gehele MediaWiki installatie (wanneer configuratieparameter $wgUseSiteCss is ingesteld)
- MediaWiki:skinname.css – algemene skin-specifieke stylesheets
- User:Username/skinname.css – gebruiker- en skin-specifieke stylesheets (wanneer configuratieparameter $wgAllowUserCss is ingesteld)

Daarnaast biedt de CSS extensie [45] ondersteuning voor CSS stylesheets in de vorm van aparte artikelen. Om de afhankelijkheid met extensies zoveel mogelijk te beperken, wordt in dit document echter verder weinig aandacht besteedt aan deze mogelijkheid. Dit is mede omdat inline CSS stijlen, hoewel het meer verbose syntax oplevert, net als stylesheets, alle benodigde presentatie-attributen ondersteunen.

Inline

Middels het toevoegen van een style attribuut kan de CSS stijl van een element worden aangepast. Dit werkt voor nagenoeg alle MediaWiki elementen, waaronder de eerder besproken table syntaxis (met als kanttekening het stramien van een tabel) en div elementen.

Implementatie

Om in plaats van de nu gebruikte MediaWiki table syntaxis, een structuur met div-elementen te gebruiken, wordt in dit hoofdstuk een mogelijke implementatie voorgesteld.

Structuur

Aangezien div-elementen direct gerelateerd kunnen worden aan de verschillende elementen in het Abstract Box Model, is de structuur ervan triviaal en als volgt:

```
<div class=”ff_resource”> <!-- Resource Box -->
     <div class=”ff_property”> <!-- Property Box -->
          <div class=”ff_label”> <!-- Label Box --> </div>
```

[45] http://www.mediawiki.org/wiki/Extension:CSS

```
            <div class=”ff_value”> <!-- Value Box --> </div>
            … <!-- Andere Value Boxes -->
        </div>
        … <!-- Andere Property Boxes -->
</div>
```

De class-attributen hier zijn niet echt nodig, maar bieden de mogelijkheid om in de toekomst stylesheets te gaan gebruiken om de infoboxes op een algemenere manier vorm te geven.

Standaard stijlen voor de verschillende niveaus

Omdat div-elementen standaard geen presentatie-informatie bevatten en de structuur van div-elementen er toch uit te laten zien als een infobox, dient hiervoor door Fresnel Forms een standaard CSS stijl gedefinieerd te worden voor de verschillende niveaus. De standaard stijlgegevens dienen dan vóór eventuele gebruikers-stijlen te worden toegevoegd voor een resulterende invulling van het betreffende XHTML style-attribuut om herdefinitie van stijlen door gebruikers toe te staan.

Initieel kan voor de standaard stijlen het volgende gekozen worden om een uiterlijk conform standaard infoboxes te krijgen:

- <div class=”ff_resource” style=”border: 1px solid #aaa; border-spacing: 3px; background-color: #f9f9f9; color: black; margin: 0.5em 0 0.5em 1em; padding: 0.2em; float: right; clear:right; text-align: left; font-size: 88%; line-height: 1.5em;”>…
- <div class=”ff_property” style=”float: left; clear:left; padding: 0.2em; width:100%”>…
- <div class=”ff_label” style=”float: left; clear:left; width:35%”>…
- <div class=”ff_value” style=”float: left; padding: 0.2em”>…

Ondersteuning voor niet-gedefinieerde properties

Om div-elementen weg te laten voor properties die voor een bepaalde infobox (individual van een ontologie) niet gevuld zijn dient het div-element voor een Property bovendien omgeven te zijn door de volgende SMW structuur, dus inclusief de onderliggende label en value-elementen:

{{#if:{{{*property*|}}}| … |}}

Toepassing

Omdat de standaard stijlen vóór eventuele gebruikers-stijlen worden geplaatst, heeft de gebruiker middels de verschillende in te stellen Fresnel CSS properties (fresnel:resourceStyle, fresnel:propertyStyle, fresnel:LabelStyle & fresnel:valueStyle) de volledige controle over hoe de gegevens getoond dienen te worden. Zelfs het uiterlijk van de infobox zelf kan volledig naar eigen hand gezet worden. Zie voor uitgebreide informatie over CSS (Bos, Håkon, Lilley, & Jacobs, 2015).

In dit hoofdstuk worden enkele CSS oplossingen geboden voor veel voorkomende stijlwensen. Deze oplossingen worden weergegeven in de volgende tabel waarbij voor een bepaalde wens wordt aangegeven welke Property welke CSS invulling hiervoor kan krijgen om het gewenste resultaat te bereiken:

Wens	Property	Invulling
Het label wordt niet getoond	fresnel:labelValue	display: none;
Het label krijgt de breedte van de infobox	fresnel:labelValue	width: 100%;
Het label wordt gecentreerd	fresnel:labelValue	text-align: center;

Tabel 3: Invulling voor veel voorkomende stijlwensen

De invulling voor verschillende elementen uit deze tabel kunnen gecombineerd worden om de invulling voor de combinatie van wensen te krijgen. Dit, door de verschillende invullingen te concateneren.

Bibliografie

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage III: CSS van Fresnel naar wikicode

Joop van de Heijning

Open universiteit

CSS van Fresnel naar wikicode

Introductie

De ABI Team 30 Fresnel Forms plug-in zal ontologieën inclusief aangepaste stijlinformatie omzetten van Fresnel naar wiki-pagina's, met de nadruk op infoboxes. Daarom is het belangrijk om te weten hoe stijlinformatie vertaald kan worden van Fresnel naar wikicode. Wiki accepteert HTML en CSS opmaak, dit rapport zal zich concentreren op CSS.

Dit verslag gaat alleen in op de laatste stappen van wáár in de wikicode stijlinformatie voor de infoboxes zou kunnen gaan. Het hoofdstuk MediaWiki beschrijft de syntax en is voornamelijk een synthese van de beschikbare informatie op de MediaWiki website[46]. Het hoofdstuk Semantic Forms geeft wat extra syntax-informatie voor formulieren. Het laatste hoofdstuk Hoe te coderen verwijst naar OWL Wikiforms (Rutledge, OWLwikiForms v0.2.1.zip), een tool voor het omzetten van ontologieën in Fresnel naar wiki-pagina's. Enige eerste gedachten over hoe de laatste stappen van de conversie in Fresnel Forms te programmeren worden gegeven. De details van Fresnel worden in dit verslag niet beschreven. Addendum: Paragraaf *Het Fresnel model in relatie tot MediaWiki's infoboxes* toegevoegd.

MediaWiki

CSS

Zo wordt CSS beschreven zoals het van toepassing is op MediaWiki op de website:

Cascading style sheet (CSS) markup sets much of the look and feel of MediaWiki: font size, colors, spacing, the logo and background image, even whether site content is displayed or is hidden.

CSS can be used to change the style of the entire wiki, for example to make the background a different colour, or you can use inline css to style specific pieces of text in your wiki. For example green text can be accomplished by doing `<span style="color:green">green text</span>`. If you want to make all text on the wiki green you can add the code `* {color:green}` to mediawiki:Common.css. (MediaWiki)

XHMTL Attributen

De website van MediaWiki behandelt XHMTL attributen als volgt:

> You can add XHTML attributes to tables. For the authoritative source on these, see the W3C's HTML 4.01 Specification page on tables.

[46] `http://www.mediawiki.org/wiki/MediaWiki`

Attributes on tables

Placing attributes after the table start tag (`{|`) applies attributes to the entire table.

You type	You get		

`{| class="wikitable" style="text-align: center; color: green;"`
`|Orange`
`|Apple`
`|12,333.00`
`|-`
`|Bread`
`|Pie`
`|500.00`
`|-`
`|Butter`
`|Ice cream`
`|1.00`
`|}`
	Orange	Apple	12,333.00

	Bread	Pie	500.00
	Butter	Ice cream	1.00

Attributes on cells

You can put attributes on individual **cells**. For example, numbers may look better aligned right.

You type	You get		

`{| class="wikitable"`
`| Orange`
`| Apple`
`| style="text-align:right;" | 12,333.00`
`|-`
`| Bread`
`| Pie`
`| style="text-align:right;" | 500.00`
`|-`
`| Butter`
`| Ice cream`
`| style="text-align:right;" | 1.00`
`|}`
	Orange	Apple	12,333.00

	Bread	Pie	500.00
	Butter	Ice cream	1.00

[..]

Attributes on rows

You can put attributes on individual **rows**, too.

You type	You get
`{| class="wikitable"`	

```
| Orange
| Apple
| style="text-align:right;"|
12,333.00
|-
| Bread
| Pie
| style="text-align:right;"|
500.00
|- style="font-style: italic;
color: green;"
| Butter
| Ice cream
| style="text-align:right;"| 1.00
|}
```

[..]

Orange	Apple	12,333.00
Bread	Pie	500.00
Butter	*Ice cream*	*1.00*

With HTML attributes and CSS styles

CSS style attributes can be added with or without other HTML attributes.

You type

```
{| class="wikitable"
style="color:green; background-
color:#ffffcc;" cellpadding="10"
|+ style="caption-side:bottom;
color:#e76700;"|''Food
complements''
|-
|Orange
|Apple
|-
|Bread
|Pie
|-
|Butter
|Ice cream
|}
```

[..]

You get

Orange	Apple
Bread	Pie
Butter	Ice cream

Food complements

Table alignment

Table alignment is achieved by using CSS. The table alignment is controlled by margins. A fixed margin on one side will make the table to be aligned to that side, if on the opposite side the margin is defined as *auto*. To have a table center aligned, you should set both margins to *auto*

For example, a right-aligned table:

You type **You get**

```
{| class="wikitable"
style="margin-left: auto;
margin-right: 0px;"
| Orange
| Apple
|-
| Bread
| Pie
|-
| Butter
| Ice cream
|}
```

Orange	Apple
Bread	Pie
Butter	Ice cream

[..]

Table floating around text

If you align a table to the right or the left side of the page, the text that comes after the table starts at the end of it, leaving an empty space around the table. You can make the text to be wrapped around the table by making the table to *float* around the text instead of just aligning it. This can be achieved using the `float` CSS attribute, which can specify where the table floats to the right side or to the left. When using *float*, margins doesn't control table alignment and can be used to specify the margin between the table and the surrounding text.

You type

```
{| class="wikitable"
style="float:right; margin-left:
10px;"
| Orange
| Apple
|-
| Bread
| Pie
|-
| Butter
| Ice cream
|}


Lorem ipsum dolor sit amet,
consectetuer adipiscing
elit, sed diam nonummy nibh
euismod tincidunt ut
laoreet dolore magna aliquam erat
volutpat. [..]
```

You get

Orange	Apple
Bread	Pie
Butter	Ice cream

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. [..]

(MediaWiki)

Semantic Forms

Nieuwe klassen kunnen toegevoegd worden aan MediaWiki's common.css bestand. Deze kunnen worden gebruikt in "tags" gebruikt door Semantic Forms:

'field' tag

Parameter `class=`*class name* - Specifies a CSS class that the input for this field should have (MW. s.d.c).

| {{{field|class="*klasse hier*"|aantal poten}}}

'standard input' tag

Parameters:

- class=*label name* - Specifies the CSS class for this input.
- style=*label name* - Specifies the CSS style for this input.

(MediaWiki)

Hoe te programmeren

Het ABI Team 30's project bouwt voort op het werk met OWF van Professor Rutledge (Rutledge, From Ontology to Wiki, 2013). Dit verslag zal zich dan ook concentreren op de CSS styling voor sjabloon-pagina's en infoboxes op een wiki.

OWF's code genereert wiki-pagina's uit een ontologie in verschillende stappen. Één van de pagina's die wordt gebouwd voor elke aanwezige semantische klasse is een sjabloon-pagina. Een sjabloon-pagina bevat een infobox.

Een sjabloon-pagina wordt gebouwd in twee stappen: Eerst maakt `writeBox()` met de `tplRow()` helper-functie een string $TplRows met alle rijen met de klasse-eigenschappen voor de infobox-tabel. Dan plakt `makeTemplatePage()`een tabel header hierin en vervolgens de tabelrijen.

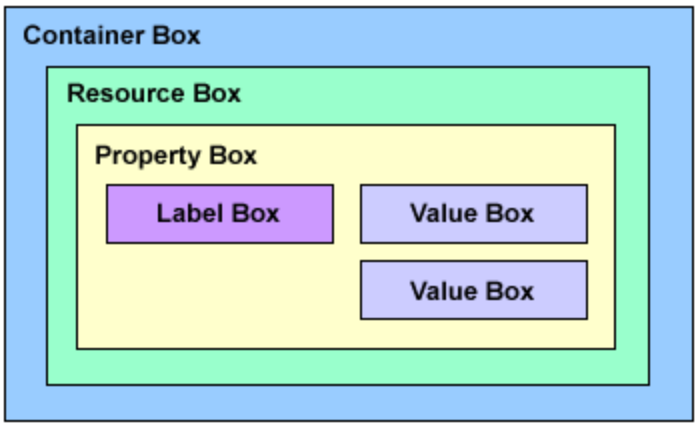
Het lijkt verstandig voor ABI Team 30 om OWF's code te volgen in dit aspect. Het hoofdstuk over MediaWiki legt uit hoe de tabel-header, de rijen en de cellen apart van elkaar worden vormgegeven. Daarom kan de algemene stijl van een infobox worden vormgegeven in een equivalente `makeTemplatePage()` methode en de individuele properties in de equivalente `tplRow()` methode.

Momenteel is de standaard stijlinformatie hard-coded in die methoden. Om functionaliteit toe te voegen in de plug-in voor aangepaste styling van de GUI onder de Protégé Fresnel Forms tab, moeten die methoden toegang hebben tot aangepaste stijlinformatie op één of andere manier. Of dit moet in de vorm van parameters, verwijzingen naar objecten of in een andere vorm is een onderwerp voor verder onderzoek.

Addendum: Het Fresnel model in relatie tot MediaWiki's infoboxes

Fresnel relies on external CSS stylesheets for defining colors, margins, fonts, borders, and other decorative elements. (W3C)

Figuur 1 laat een voorbeeld van het abstract box model van Fresnel zien.



Fresnel	**MediaWiki**
Resource Box	Conforms most naturally to the (infobox) table as a whole: CSS attribute on table.
Property Box	One row (property + value): CSS attribute on row.
Label Box	Property cell: CSS attribute on (left) cell.
Value Box	Value cell: CSS attribute on (right) cell.
Container Box	Not a natural match. Could be used for styling all or some wikipages on the server.

Figuur 1 **Fresnel Abstract Box Model (W3C)**

Interessant voor ons project is hoe het model relateert aan MediaWiki's infoboxes.

Referenties

MediaWiki. (n.d.). Retrieved December 5, 2014, from Manual:CSS: https://www.mediawiki.org/wiki/Manual:CSS

MediaWiki. (n.d.). Retrieved December 5, 2014, from Help:Tables: http://www.mediawiki.org/wiki/Help:Tables

MediaWiki. (n.d.). Retrieved December 5, 2014, from Extension:Semantic_Forms/Defining_forms: https://www.mediawiki.org/wiki/Extension:Semantic_Forms/Defining_forms

Rutledge. (2013). From Ontology to Wiki. *ESWC.* Montpellier.

Rutledge. (n.d.). *OWLwikiForms v0.2.1.zip*. Retrieved December 6, 2014, from OWL Wiki Forms v0.2 installation: http://is.cs.ou.nl/OWF/files/OWLwikiForms_v0.2.1.zip

W3C. (n.d.). *Fresnel - Display Vocabulary for RDF*. Retrieved January 11, 2015, from W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage IV: Fresnel generatie voor het koppelen van formats aan lenzen en properties.

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

Fresnel generatie voor het koppelen van formats aan lenzen en properties. 64
Inleiding 66
Samenvatting 66
De standaard fresnel generatie 66
Frensel generatie na aanpassen van de lens. 66
Alternatief voorstel Fresnel generatie. 67
Conclusie 68
Bijlage A. Definities van begrippen. 68
Lenzen definieren wat er getoond moet worden. 68
Formats definieren hoe de lens attributen getoond moet worden. 69
Groep definities 69
References 69

Inleiding

Dit document analyseert de gegenereerde Fresnel code en doet een alternatief voorstel. We gebruiken Jfresnel als ‘gouden standaard’ en laten zien dat het alternatief geparset kan worden door Jfresnel.
De ‘Shakespeare’ ontologie is als voorbeeld genomen.

Samenvatting

De standard Fresnel generatie bevat een aantal omissies. Er wordt geen standaard stijl gedefinieerd. De stijldefinities worden niet goed toegekend. Benodigde groepen worden niet goed gedefinieerd.
Voor specifieke stijling van properties worden blank nodes gebruikt.

In het alternatieve voorstel worden de problemen opgelost door per lens een groep te definiëren voor de standaard stijling. De specifieke stijling voor een property wordt gekoppeld aan de lens groep en het attribuut.

De blank nodes blijken geen probleem te vormen en zijn gehandhaafd.

De standaard fresnel generatie.

Het blijkt dat alle lenzen worden toegevoegd aan een standaard groep <naamontology>Group.

```
fresnel:group                 :shakespeareGroup
```

Voor elke property die getoond wordt, wordt een format aangemaakt.
Via fresnel:propertyformatdomain wordt het format gekoppeld aan de property.
Ook wordt elke format gekoppeld aan de standaard groep <naamontology>Group.

```
:marriedFormat  a                     fresnel:Format ;
        fresnel:group                 :shakespeareGroup ;
        fresnel:label                 "married" ;
        fresnel:propertyFormatDomain  shakespeare:married .
```

Er wordt geen groep definitie vastgelegd voor een lens en daarmee ook geen standard formattering.

Stijldefinities worden aan lenzen toegekend en niet aan formats of groepen, deze stijldefinities worden dan ook niet door Jfresnel herkent.

Frensel generatie na aanpassen van de lens.

De personlens is aangepast, op alle properties een rename uitgevoerd (Married, Lived in, Has child) en de property ‘married’ heeft een speudo class als css style gekregen (ps, ls, vs, cs).
Deze rename actie heeft als resultaat propertie formats die de labels definiëren.

```
:marriedPersonLensFormat
        a                             fresnel:Format ;
        fresnel:label                 "Married" ;
        fresnel:propertyFormatDomain  shakespeare:married .
```

De css stijldefinities worden in aparte blank nodes (_:b0, _:b1, _:b2) gedefinieerd

```
_:b2    a                 fresnel:propertyDescription ;
        fresnel:property  shakespeare:married ;
```

```
        fresnel:use       :marriedPersonLensFormat .
```

De lens ‘PersonLens’refereerd nu aan deze blanc nodes:

```
fresnel:showProperties   ( _:b2 _:b0 _:b1 ) .
```

Deze contructie is opgezet om de fresnel:use tag te kunnen gebruiken. Hiermee wordt het gedefinieerde formaat enkel gekoppeld aan de specifieke lens.

Jfresnel laat de lenzen en de gedefinieerde formats zien maar de blank nodes worden niet getoond in het showproperties statement. Voor de properties van de lens wordt ‘MP’ getoond.

Alternatief voorstel Fresnel generatie.

Elke lens krijgt een groep, in die groep wordt de stijl definitie van die lens vastgelegd.

```
:PersonLensGroup   a                  fresnel:Group;
       fresnel:containerStyle   "ccs"^^fresnel:styleClass ;
       fresnel:labelStyle       "cls"^^fresnel:styleClass ;
       fresnel:propertyStyle    "cps"^^fresnel:styleClass ;
       fresnel:resourceStyle    "crs"^^fresnel:styleClass ;
       fresnel:valueStyle       "cvs"^^fresnel:styleClass .
```

De properties van de lens die gewijzigd zijn krijgen elk een eigen format, zoals nu ook al het geval is, waarin het label en de css stijl definities zijn opgeslagen. Dit format wordt nu gekoppeld aan de lensgroep i.p.v. de ontology groep. Omdat het formaat ook aan de betreffende property is gekoppeld is het format specifiek voor de property van de betreffende lens.

```
:marriedPersonLensFormat   a               fresnel:Format ;
              fresnel:group                :PersonLensGroup;
              fresnel:propertyFormatDomain  shakespeare:married ;
```

Jfresnel kan deze constructie interpreteren en het resultaat is:

```
Lens: file:///D:/My Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#PersonLens
Label: PersonLens
Comment: null
Purpose: DEFAULT
Lens Domains:
BasicClassDomains: http://www.semanticweb.org/owl/owlapi/turtle#Person
Properties to show:
BASIC http://www.semanticweb.org/owl/owlapi/turtle#married
BASIC http://www.semanticweb.org/owl/owlapi/turtle#livedIn
BASIC http://www.semanticweb.org/owl/owlapi/turtle#hasChild
VISIBILITY, allProperties at -1
Associated Groups:
file:///D:/My Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#PersonLensGroup
Associated Formats:
file:///D:/My
Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#marriedPersonLensFormat
```

```
* 1 group(s)
--------------------------------
Group: file:///D:/My
Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#PersonLensGroup
Label: PersonLensGroup
Associated Lenses:
file:///D:/My Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#PersonLens
Associated Formats:
file:///D:/My
Documents/Protege/shakespeare_fresnel_customPersonLens_v5.n3#marriedPersonLensFormat
Resource Style: crs
Label Style: cls
Value Style: cvs
Property Style: cps
```

We zien hier dat de lens gekoppeld is aan een groep, in die groep kunnen stijldefinities staan. Deze stijldefinities behoren aan de lens toe.
Aan de lens zijn algemene formats gekoppeld voor elke property.
Aan de groep zijn de formats gekoppeld die specifiek zijn voor de properties van deze lens.

Deze constructie voorkomt dus dat er blank nodes ontstaan.
De fresnel:use tag werkt zover ik kan nagaan alleen met blank nodes.
Volgens mij is de Fresnel structuur ook makkelijker te parsen om wiki code mee te produceren.

Conclusie

Besluit Fresnel generatie:

Omdat het geen problemen oplevert maken we nog steeds gebruik van de blank nodes voor de property definities en de expliciete koppeling naar een formaat met de fresnel:use tag.

Voor elke lens wordt een groep gedefinieerd met daarin de standaard css formattering.

Definities van begrippen.

Lenzen definiëren wat er getoond moet worden.

Via de property fresnel:instanceLensDomain wordt vastgelegd voor welke instanties de lens geld. Met de property fresnel:classLensDomain wordt vastgelegd voor welke classe de lens geld.

De lens definieerd welke attributen getoond worden en in welke volgorde.
`fresnel:showProperties, fresnel:hideProperties`

Er kan nog op waardes van attributes geselecteerd worden d.m.v. sublensen.

```
[ fresnel:property ex:activity ;
fresnel:sublens :projectDefaultLens ;
fresnel:sublens :hobbyDefaultLens ]
```

Lensen kunnen nog onderscheiden worden via de tag fresnel:purpose,
bv. Infobox, forms

Formats definiëren hoe de lens attributen getoond moet worden.

Formats kunnen gekoppeld worden aan properties, classes, instances, groepen en lenzen:

1. Fresnel:propertyFormatDomain
   definieert de property waarvoor dit format geld.
2. Fresnel:classFormatDomain
   definieert het format voor classes.
3. Fresnel:instanceFormatDomain
   definieert het format voor instances.
4. Fresnel:use
   Lenzen kunnen direct aan een format gekoppeld worden door in de lensdefinitie een '**fresnel:use'** tag op te nemen
5. Fresnel:group
   definieert de groep waarop het format van toepassing is.

Groep definities

Format statements kunnen opgenomen worden bij groep definities en gelden dan voor alle lenzen behorende bij die groep. Dit geeft een standaard formattering voor alle lenzen behorende bij die groep.

```
:foafGroup rdf:type fresnel:Group ;
           fresnel:resourceStyle "globalResource"^^fresnel:stlyeClass ;
           fresnel:propertyStyle "globalProperty"^^fresnel:stlyeClass ;
           fresnel:labelFormat [ :contentAfter ":"  ] .
```

Formats kunnen gekoppeld worden aan een groep met het statement

```
:depictFormat rdf:type fresnel:Format ;
              fresnel:group :foafGroup .
```

References

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Retrieved from W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Retrieved from W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (n.d.). *Help:Import vocabulary*. Retrieved May 3, 2015, from Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (n.d.). *Help:RDF Export*. Retrieved May 2, 2015, from Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Retrieved from https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Retrieved January 11, 2015, from W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage V: Fresnel uitbreiding met OWF_Style properties

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

Fresnel uitbreiding met OWF_Style properties 72
Inleiding 74
Probleem 74
Uitbreiden van de fresnel ontologie 74
De OWF Style definties 75
De range property 75
Individuals voor de Fresnel.purpose property 75
Referenties 76

Inleiding

Dit document is geschreven om te onderzoeken hoe de Fresnel ontologie uitgebreid kan worden. Daartoe wordt in kaart gebracht welke informatie de wiki moet tonen. Daarna of deze informatie al in de Fresnel ontologie bestaat en hoe deze dan gebruikt kan worden. Bestaat de informatie niet dan wordt bekeken hoe de Fresnel ontologie uitgebreid kan worden zodat de Fresnel to wiki conversie alsnog de informatie kan tonen op de wiki.

Probleem

Het OWLWikiForms PHP script query'd zowel de Fresnel ontologie als de bron ontologie om de vertaling van Fresnel naar Semantic Wiki Forms te maken. De Fresnel Forms plug-in dient hiervoor alleen de Fresnel ontologie te gebruiken. Om de definitie van de forms user interface onafhankelijk te maken van de de bron ontologie dient de Fresnel ontologie uitgebreid te worden.

Concreet voorbeeld is de range van object properties deze wordt uit de bronontologie gehaald. De Fresnel ontologie zou uitgebreid moeten worden met een owf:range predicaat die de rdfs:range vervangt.

Uitbreiden van de Fresnel ontologie

De uitbreidingen op de Fresnel ontologie worden opgenomen in de OWF_Style ontologie. Deze ontologie wordt gehost op adres http://is.cs.ou.nl/OWF/OWF_Style.owl
Om te bepalen hoe deze ontologieën gekoppeld moeten worden is als voorbeeld de uitbreiding van Fresnel zelf genomen. Deze uitbreiding staat in het bestand extended.owl, hierna genoemd Extended Fresnel. Hier wordt aan de bestaande klasse van Fresnel ontologie definities toegevoegd.
In de Extended Fresnel ontologie wordt via owl.imports de Fresnel ontologie geimporteerd.

```
owl:imports      <http://www.w3.org/2004/09/fresnel> ;
```

De default prefix/namespace in de Extended Fresnel ontologie is:

```
@prefix  :           <http://www.w3.org/2004/09/fresnel#>  .
```

De default namespace van het document is dus dezelfde als die in de core Fresnel ontologie en daarmee breid het deze vocabulaire uit.

In bestand 'OWF_style.owl' is de import methode gebruikt om de owf_style ontologie te koppelen aan de Fresnel ontologie. Omdat we voor de uitbreidingen een aparte namespace willen gebruiken verwijst de standaard prefix niet naar de Fresnel namespace maar naar de OWF namespace (http://is.cs.ou.nl/OWF/). De import methode verwijst naar de extended Fresnel ontologie, de extended Fresnel ontologie importeert weer naar de core Fresnel ontologie. Hierdoor zijn alle Fresnel constructies beschikbaar als er naar de OWF_Style ontologie verwezen wordt.

De OWF style ontologie bevat hiervoor de volgende statements:

```
@prefix : <http://is.cs.ou.nl/OWF/OWF_style#> .

@prefix fresnel: <http://www.w3.org/2004/09/fresnel-extended#> .

owl:imports <http://www.w3.org/2004/09/fresnel-extended> .
```

Omdat de gehele Fresnel ontologie uit meerdere bestanden bestaat is voor elke bestaande Fresnel term die gebruikt wordt is een 'isDefinedBy' triple opgenomen zoals in onderstaand voorbeeld:

```
:fslSelector  rdfs:isDefinedBy <http://www.w3.org/2004/09/fresnel> ;
```

De OWF Style definties

De range property

De fresnel2wiki conversie vraagt de rdfs:range property op bij het bepalen uit welke klasse de individuals worden getoond van het wiki AutoCompleteFromClass statement.

De rdfs:range informatie van een property wordt in de OWF_Style ontologie gedefinieerd met: autocompleteFromClass. Dit objectproperty koppeld een formatdefinitie aan de klasse waarvan de individuals getoond moeten worden.

De owf:autocompleteFromClass property is gedefinieerd als

```
:autocompleteFromClass rdf:type owl:ObjectProperty ;
          rdfs:isDefinedBy "<http://is.cs.ou.nl/OWF/OWF_style>"^^xsd:string ;
          rdfs:comment "The autocompleteOnClass defines the class and
                        thereby the Semantic Wiki Forms category to show in
                        the autocompletion field."

          rdfs:range rdfs:Class ;
          rdfs:domain fresnel:Format .
```

Een voorbeeld uit de Tim Berners Lee ontologie

```
:parentPersonLensFormat a fresnel:Format ;
       owf:autocompleteFromClass  dbpedia_tim_berners_lee_extract:Person ;
```

Een property kan meerdere range definities hebben, in dat geval zouden ze gecombineerd kunnen worden. De PHP code van OWL wiki forms beperkt zich tot 1 range definitie van een property. In de plug-in Fresnel Forms is dat ook zo gedefinieerd.

Het object van de fresnel.range tripel is gedefinieerd als klasse er kunnen dus geen literals opgenomen worden.

De range bepaalt van welke klasse de individuals getoond moet worden maar er is geen mogelijkheid om een property van een klasse te tonen.

Individuals voor de fresnel.purpose property

De fresnel.purpose property wordt gebruikt om onderscheid tussen de verschillende klasse en format definities te maken.

De Purpose klasse definieert de termen voor de purposes die onderscheiden kunnen worden.
In de owf_style ontologie zijn 2 uitbreidingen gedefinieerd om onderscheid te maken tussen formats die gebruikt worden voor de infoboxen en wikiforms.

```
:InfoBox rdf:type owl:NamedIndividual ,
           fresnel:Purpose ;
:WikiForms rdf:type owl:NamedIndividual ,
             fresnel:Purpose> ;
```

Een voorbeeld van het gebruik:

```
:isEmployedByFormat rdf:type fresnel:Format ;
        fresnel:propertyFormatDomain  TheFirm:isEmployedBy ;
        fresnel:group :TheFirmGroup ;
        fresnel:purpose :InfoBox;
        fresnel:label "isEmployedBy" .
```

Referenties

Ryan Lee. (2008). Fresnel - Display Vocabulary for RDF. Opgehaald op 19-11-2014. http://www.w3.org/2005/04/fresnel-info/

Bijlage VI: Overzicht van de fresnel2wiki functionaliteit.

Open Universiteit
Faculteit Informatica

18-9-2017

versie 1.0

Joop van de Heijning
J. N. Theunissen
838573218

Inhoud

Overzicht van de fresnel2wiki functionaliteit. 77
Inleiding 79
Overzichten 79
High level overzicht functionaliteiten 79
OWF ontologie definities 79
Referenties 81

Inleiding

Dit document geeft een highlevel overzicht van de fresnel2wiki functionaliteiten van de Fresnel Forms plugin. Daarnaast wordt een overzicht gepresenteerd van de uitbreidingen op de Fresnel ontologie.

Overzichten

Onderstaande lijst geeft een overzicht van de functionaliteiten die nodig zijn voor de wiki export.

High level overzicht functionaliteiten

- Elke lens krijgt een category pagina en bijbehorende form, template en property pagina's. Deze zijn voorzien van de nodige tags om te functioneren in een Semantic MediaWiki met Semantic Forms omgeving.
- Deze worden in een XML bestand gezet, gereed om te importeren met special:import in een MediaWiki server met Semantic MediaWiki en Semantic Forms extensie.
- Fresnel resourceStyle, propertyStyle, labelStyle en valueStyle properties worden uit de lens en property formats gehaald op basis van prioriteit [1]. Deze worden met CSS in de informboxes in de templates verwerkt[2].
- Het Fresnel property label wordt, gewijzigd of niet door de gebruiker in de gui, opgehaald en getoond in de informbox. Tenzij de gebruiker in de gui heeft aangegeven dat het niet getoond moet worden (dan heeft "fresnel:label" de waarde "fresnel:none").
- Indien de gebruiker aangegeven heeft dat een property als type een "Image" is, krijgt de property in Semantic MediaWiki type "URL" en de template wordt zo geformatteerd dat de informbox een gecentreerd plaatje verwacht.
- Per property wordt de waarde van owf:autocompleteFromClass uit Fresnel gehaald, indien de property type "page" heeft.
- Het type van de range van een property van een lens wordt opgehaald en aan het formulier behorende bij de lens meegegeven[2].
- Kardinaliteit van een property wordt opgehaald (na aangeven van gebruiker in gui) en aan het formulier meegegeven.
- In samenhang met bovenstaande: Meerdere waarden of niet voor een property in een formulier?
- Per property wordt de waarde van owf:delimiter uit Fresnel gehaald. Deze waarde is aangegeven door de gebruiker in de gui. Wordt verwerkt in de informbox op de template en geeft aan wat het tussenvoegsel bij de weergave van meerdere waarden voor één property wordt. Kan ook gebruikt worden om waarden onder elkaar te laten weergeven door als delimiter "\n" op te geven in de gui.

OWF ontologie definities

De tabel bestaat uit 3 kolommen, de 1e toont de rdf patronen in de bronontologie die door de OWF uitbreiding beschreven worden, de 2e kolom toont het bijbehorende OWF statement, de 3e kolom laat de Semantic Forms (SF) code zien.

RDF patroon in bronontologie	OWF statement	SF code
Een object property met een range definitie.	Koppelt een classe aan een format definitie voor een property.	De SF code die een veld definieert met een auto compleet functie op de bij de klasse behorende category.
bio:livedIn rdf:type owl:ObjectProperty ; rdfs:domain lit:Person ; rdfs:range lit:Place .	owf:autocompleteFromClass	| {{{field|livedIn|autocomplete on category=Place|list }}}
n.v.b.	Definieert de x positie van een lens in de GUI van de Fresnel Forms plug-in.	n.v.b voor SF code generatie
	owf:xpos	
	Definieert de y positie van een lens in de GUI van de Fresnel Forms plug-in.	n.v.b voor SF code generatie
	owf:ypos	
Een property met een literal en een datatype specificatie. Zie bijlage A voor mapping xsd datatype naar OWF datatype	Specificeert het datatype van een property in de infobox weergave.	De SMW tag
Jane eg:age "15.0"^^xsd:decimal	owf:datatype :Number	[[Datatype::Number]]
Een fresnel format definitie met property owf:delimiter	Definieert het lijst scheidingsteken	
owf:delimiter "\\n" ;	owf:delimiter "\n"	{{{award|}}}|,|xxx|[[award::xxx]]|\n\n}
De equivalentie klasse die aangeeft dat het veld naam een enkelvoudig veld is	Definieert een enkelvoudige property.	Het keyword 'List' wordt niet vermeld in de SF veld definitie.
owl:equivalentClass [rdf:type owl:Restriction ; owl:onProperty foaf:name ; owl:maxQualifiedCardinality "1"^^xsd:nonNegativeInteger ; owl:onDataRange xsd:string] ,	owf:isList false;	{{{field|actors_and_movies-Actor-foaf-name }}}
De equivalentie klasse die aangeeft dat het veld naam verplicht is.	Definieert een verplichte property.	Het keyword mandatory geeft aan dat het veld verplicht is.
owl:equivalentClass [rdf:type owl:Restriction ; owl:onProperty foaf:name ; owl:minQualifiedCardinality "1"^^xsd:nonNegativeInteger ; owl:onDataRange xsd:string] ,	owf:isMandatory true ;	{{{field|directedBy|autocomplete on Director|mandatory }}}

Elk OWF statement is gedefinieerd in de OWF namespace met de standaard prefix owf.
n.v.b.: niet van belang.

Referenties

1. ABI-team 30. Prioritering van Property Formats, 2015.
2. ABI-team 30. CSS van Fresnel in wiki code_v2, 2015
3. ABI-team 30. Issue29_range in fresnel definieren, 2015

Bijlage VII: Prioritering van Property Formats

Ontwerp van een algoritme voor het prioriteren van Fresnel Property Formats .

Alex Mekkering

Revisiehistorie

Datum	**Beschrijving**
2015-01-25	Initiële versie
2015-02-01	CSS concatenatie + owf:delimiter verwerkt
2015-02-20	Revisiehistorie bijgewerkt
2015-03-31	Overige OWF properties toegevoegd
2015-05-12	Uiteindelijke versie

Doel

Het doel van dit document is om een algoritme te ontwerpen voor het prioriteren van Fresnel Property Formats.

Inleiding

Bij het toepassen van Fresnel Formats op Properties welke geselecteerd zijn door Fresnel Lenses kan het zijn dat er meerdere Fresnel Formats geldig zijn voor een betreffende Property voor die Lens. Deze Fresnel Formats kunnen echter tegenstrijdige presentatie-informatie bevatten waardoor er een noodzaak is tot het eenduidig selecteren van presentatie-informatie.

Om de presentatie-informatie eenduidig te kunnen bepalen uit een collectie van geldige formaten, wordt hiervoor in dit document een algoritme ontworpen.

In dit document wordt, na een korte uitleg over verschillende onderdelen van de Fresnel ontologie, een algoritme bepaald welke uit een collectie van geldige formaten, de presentatie-informatie zo goed als mogelijk eenduidig bepaalt.

Achtergrond

Properties op een Lens

De selectie van Properties door een Lens wordt bepaald door de fresnel:showProperties en fresnel:hideProperties predicaten. Het fresnel:showProperties predicaat bepaalt welke properties geselecteerd worden en het fresnel:hideProperties predicaat bepaalt welke properties ge-deselecteerd worden. Hierbij geldt dat fresnel:hideProperties voorrang heeft op fresnel:showProperties; properties welke zowel in fresnel:showProperties als in fresnel:hideProperties vermeld zijn, zijn niet geselecteerd voor die lens.

Aanduiding van properties

Het aanduiden van properties bij fresnel:showProperties en fresnel:hideProperties kan op twee manieren gebeuren:

1. Een property wordt, middels zijn IRI, rechtstreeks vermeld in één van de lijsten.
2. Er wordt een blank node toegevoegd welke de property beschrijft. Deze blank node bevat minstens de statements ‘rdf:type fresnel:propertyDescription’ en ‘fresnel:property <property IRI>’, waarbij <property IRI> de IRI van de betreffende property is. Daarnaast wordt expliciet een Format gekoppeld aan de betreffende property door een ‘fresnel:use <Format IRI>’ statement, waarbij <Format IRI> de IRI van de betreffende Format is.

Presentatie-informatie

De presentatie-informatie voor een property, zoals die nu gebruikt wordt binnen de Fresnel Forms Protégé Plugin, kan bestaan uit statements met de volgende predicaten:

- fresnel:label – het label van een property. Kan tekst zijn welke het label definieert, fresnel:none om het label niet te tonen of fresnel:show om het label wel te tonen (default).

- fresnel:value – manier om aan te geven hoe alle waarden van een property getoond moeten worden. Kan één van de volgende waarden hebben:
    - fresnel:image – waarde moet getoond worden als plaatje.
    - fresnel.externalLink – waarde moet getoond worden als klikbare link.
    - fresnel.uri – waarde moet getoond worden als URI.
- fresnel:propertyStyle – de (CSS) stijl van het gebied waarin de property wordt weergegeven.
- fresnel:labelStyle – de (CSS) stijl van het gebied waarin het label van de property wordt weergegeven.
- fresnel:valueStyle – de (CSS) stijl van het gebied waarin de waarde van de property wordt weergegeven.
- owf:delimiter – het scheidingsteken welke gebruikt moet worden om verschillende waarden voor die property in een lijst van elkaar te scheiden.
- owf: autocompleteFromClass – welke klasse(n) gebruikt moet(en) worden voor een object property door een autocompletion functie.
- owf:datatype – welk datatype voor een datatype property geldt.
- owf:isList – of een property meerdere waarden (objecten) kan bevatten voor hetzelfde subject, of niet.
- owf:isMandatory – of de invulling van een property voor een subject verplicht is, of niet.

Een lens kan, naast de CSS stijlen hierboven welke dan als default gelden voor alle properties op die lens, ook nog een fresnel:resourceProperty statement hebben welke de (CSS) stijl bepaalt van het gebied waar de lens in weergegeven wordt. Alle CSS stijlen voor een Lens bevinden zich in een fresnel:Group.

Gelaagdheid

De presentatie-informatie van een property op een lens kan op verschillende niveaus gedefinieerd zijn. Opgesomd in toenemende mate van specificiteit zijn deze niveaus:

1. Er kunnen generieke Formats gedefinieerd zijn met als fresnel:propertyFormatDomain de betreffende property. Dit generieke Format heeft dan geen fresnel:Group gedefinieerd.
2. Wanneer de betreffende Lens onderdeel is van een fresnel:Group, kan voor die Group bepaald zijn wat de presentatie-eigenschappen zijn voor alle Resources en properties welke door de Lens geselecteerd worden. Een fresnel:Group kan echter slechts stijlinformatie voor properties definiëren, dus geen fresnel:label en fresnel:value.
3. Er kunnen Formats gedefinieerd zijn met als fresnel:propertyFormatDomain de betreffende property én welke onderdeel zijn van één of meer van de groepen waar ook de Lens onderdeel van is. Deze laag kan bestaan uit meerdere sub-lagen: hoe meer fresnel:Group objecten overeenkomen tussen het betreffende Format en de betreffende Lens, hoe specifieker het Format is voor die property en die Lens.
4. Een Format kan expliciet gekoppeld zijn aan de selectie van een property voor een Lens. Dit vindt dan altijd plaats door een fresnel:propertyDescription aanduiding, dus met een Blank Node.

Voor het bepalen van de uiteindelijk presentatie-eigenschappen van een Property voor een Lens is het belangrijk om alle lagen van minst specifiek naar meest specifiek te

inventariseren. In de minst specifieke laag kan immers bepaald zijn dat de property een fresnel:propertyStyle en fresnel:valueStyle definitie heeft, terwijl op een specifieker niveau de fresnel:propertyStyle nogmaals gedefinieerd is. De uiteindelijke presentatie-informatie die dan geldt, is de fresnel:valueStyle uit het minst specifieke Format en de fresnel:propertyStyle uit het specifiekere Format.

Ambiguïteit

Binnen een laag zoals hierboven opgesomd kunnen Formats gedefinieerd zijn die tegenstrijdig zijn. Er kunnen immers best twee Formats zijn die een property op een generieke manier beschrijven. Dit levert echter een niet-consistente Fresnel ontologie op. Volgens (Bizer, Lee, & Pietriga, 2015) dient dan één van de formaten gekozen te worden.

Meerdere definities binnen hetzelfde Format

Voor een Format kunnen ook meerdere statements voor dezelfde property gedefinieerd zijn.

Als voorbeeld kan de property owf:autocompleteFromClass meerdere keren gedefinieerd zijn voor een Format. Dit houdt dan in dat de autocompletion functionaliteit een keuze moet bieden uit de vereniging van alle instanties van de opgegeven klassen.

Het algoritme kan hier generiek ondersteuning voor bieden door voor elke laag, per property, een lijst van mogelijke waarden terug te geven.

CSS

De waarden van de CSS stijl properties (fresnel:*Style) bestaan uit CSS welke meerdere CSS attributen kan bevatten. Zo kan in bijvoorbeeld voor fresnel:propertyStyle de waarde “color:purple; font-weight:bold;” gedefinieerd zijn, wat inhoud dat de tekst binnen een property de kleur paars moet krijgen en vet gedrukt moet zijn.

Dezelfde styling property, in verschillende lagen gedefinieerd, dus met verschillende specificiteit, kan dus CSS attributen bevatten welke elkaar niet of slechts gedeeltelijk overlappen. Om hiervoor ondersteuning te kunnen bieden dienen dezelfde styling properties van verschillende specificiteit zodanig samengevoegd te worden dat alle informatie erin behouden blijft, evenals de specificiteit ervan.

CSS (Bos, Håkon, Lilley, & Jacobs, 2015) biedt hier ondersteuning voor doordat attributen die later zijn gedefinieerd (achteraan in de string staan) per definitie meer specifiek zijn dan attributen die eerder zijn gedefinieerd. Zo geldt bij de CSS waarde “color:purple; color:yellow;” dat de kleur van het element geel moet worden.

Oplossing

Om de meest specifieke presentatie-informatie voor een Property voor een Lens te bepalen is het volgende bedacht:

- De gelaagdheid, zoals hierboven opgesomd, wordt geïmplementeerd door Formats en Groups prioriteiten toe te kennen waarbij 0 het minst specifiek is en een hoger nummer een steeds specifieker Format of Group betreft,
- Het aantal prioriteiten is gelijk aan 3 (de 1^e, 2^e en 4^e laag) + het aantal fresnel:Group statements voor de betreffende Lens. Omdat de doorsnede van twee verzamelingen niet meer elementen kan hebben dan het minimum van het aantal elementen van de

verzamelingen kan een Format niet meer overeenkomstige fresnel:Group statements hebben met de Lens als dat de Lens zelf fresnel:Group statements heeft.

- Voor elke prioriteit wordt een lijst voor Formats en Groups gedefinieerd. Dit om de eigenschappen van Formats die in principe tegenstrijdig zijn, maar andere presentatie eigenschappen beschrijven (dus eigenlijk toch niet tegenstrijdig zijn), toch correct mee te kunnen nemen. Formats en Groups worden aan één van deze lijsten toegevoegd, op basis van hun prioriteit.

De prioriteit van Formats en Groups wordt als volgt bepaald:

- Een generiek Format met als fresnel:propertyFormatDomain <property IRI> en geen fresnel:Group krijgt prioriteit 0.
- Een fresnel:Group waarvan de Lens onderdeel is, krijgt prioriteit 1.
- Een Format welke een fresnel:propertyFormatDomain <property IRI> heeft en fresnel:Group statements heeft die overeenkomen met die van de betreffende Lens, krijgt als prioriteit 1 + (het aantal fresnel:Group statements welke overeenkomen met die van de Lens).
- Een Format welke expliciet gekoppeld is aan een Property (middels fresnel:propertyDescription) krijgt de hoogste prioriteit (3 + (het aantal fresnel:Group statements welke overeenkomen met die van de Lens) -1).

Per property voor een Lens worden de presentatiegegevens als volgt bepaald:

1. Het algoritme wordt geïnitialiseerd:
    a. Er wordt gekeken of er voor de betreffende property een propertybeschrijving aanwezig is voor de Lens (blank node met fresnel:propertyDescription).
    b. Het aantal prioriteiten wordt bepaald op basis van het aantal fresnel:Groups waarvan de Lens onderdeel is.
    c. Er wordt een geprioriteerde Lijst van lege verzamelingen opgebouwd waarbij de verzameling met de laagste prioriteit (0, minst specifiek) als eerst gedefinieerd is.
2. Alle fresnel:Groups worden in de verzameling met prioriteit 1 geplaatst.
3. Voor alle Formats wordt nu het volgende gedaan:
    a. De prioriteit wordt bepaald.
    b. Indien de prioriteit bepaald kon worden (Het Format was van toepassing op de betreffende Property op de Lens), wordt het Format toegevoegd aan de verzameling met de juiste prioriteit.
4. Vervolgens wordt de geprioriteerde lijst van verzamelingen in toenemende prioriteit doorlopen om de hierin verzamelde meest-specifieke presentatie-gegevens voor de Property op de Lens te bepalen.
    a. Voor styling properties worden de waarden (CSS) achteraan toegevoegd aan eventueel eerder gevonden waarden
    b. Voor overige properties geldt dat gegevens welke in een eerdere verzameling (met lagere prioriteit) aanwezig waren worden overschreven. Hierbij worden de complete lijsten van property waarden overschreven met de property waarden welke voor het betreffende Format gelden.
    c. Voor de verzameling met Groups (prioriteit 1) worden de niet-stijl properties niet meegenomen.

Alle meest specifiek presentatiegegevens worden verzameld in een instantie van een klasse welke is afgeleid van een HashMap welke als sleutel een property heeft en als waarde een

lijst van propertywaarden. Er wordt gebruik gemaakt van een afgeleide klasse van HashMap om een ‘pick’ methode te kunnen definiëren welke van de lijst van waarden één arbitraire waarde teruggeeft. Naast de standaard interface van een HashMap, wordt hiermee een API geboden waarmee voor eenvoudige properties, waarvan verwacht wordt dat er niet meer dan één waarde gedefinieerd is, de waarde eenvoudiger bepaald kan worden.

Bibliografie

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor’s group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS’14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage VIII: Integratie van namespaces

Ontwerp voor het integreren van namespaces in Fresnel Forms.

Alex Mekkering

Revisiehistorie

Datum	**Beschrijving**
2015-01-11	Initiële versie
2015-01-17	Ontwerp bijgewerkt
2015-02-20	Revisiehistorie bijgewerkt
2015-05-13	Uiteindelijke versie

Doel

Het doel van dit document is om te bepalen hoe we namespaces van elementen uit de bronontologie kunnen integreren in Fresnel Forms. Het is hierbij uitdrukkelijk niet de bedoeling om het huidige ontwerp in zijn geheel te herzien; slechts de delen die belang zijn voor het correct integreren van URI namespaces worden aangepakt, mede om de tijdsbesteding voor deze taak niet uit de hand te laten lopen.

Inleiding

Op dit moment wordt vanuit de bronontologie bepaald voor welke OWL Classes en Properties lenzen dienen te worden gecreëerd op de gebruikersinterface. Hierbij wordt echter bij de benaming van zowel de lenzen als de properties geen rekening gehouden met de namespace van de betreffende Class of Property.

Wanneer verschillende properties dezelfde naam hebben, maar een andere namespace, kunnen deze hierdoor niet van elkaar onderscheiden worden op de GUI. Door rekening te houden met de namespace van een Class of Property, kan dit onderscheid wel gemaakt worden aangezien de namespace en de naam (fragment identifier) van een resource de bijbehorende URI vormen en de URI van een RDF resource altijd uniek is. (The Internet Engineering Task Force, 2015)

In dit document wordt, na een korte uitleg over het toepassen van prefixes, eerst een analyse uitgevoerd van de huidige implementatie waarin de problemen erin worden benoemd. Daarna wordt een oplossing voor deze problemen geboden in de vorm van een aangepast ontwerp en uitleg over het dynamisch bepalen van namen op basis van URI’s.

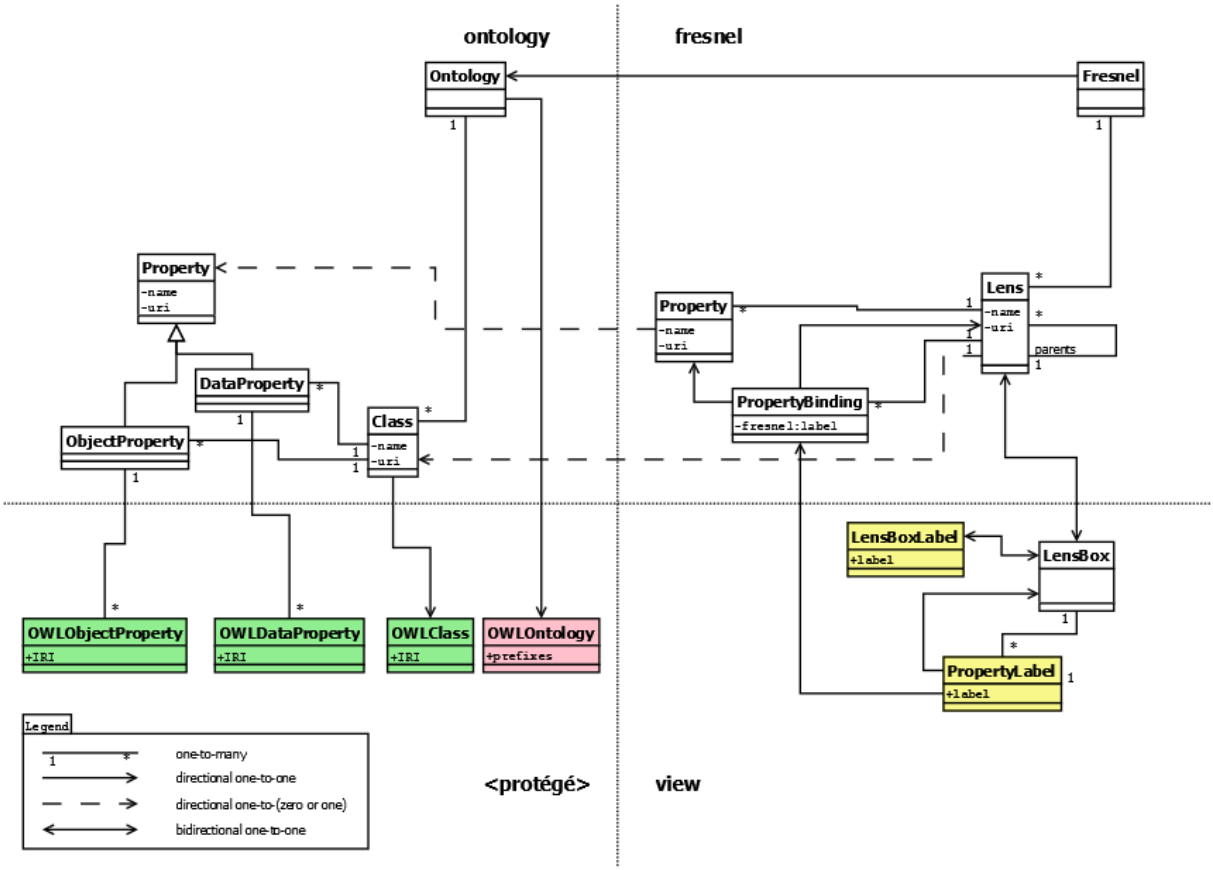
Het gebruik van prefixes

Volledige URI’s hebben meestal een lengte die lastig weer te geven is in een gebruikersinterface. Binnen de meeste ontologiëen zijn echter ook prefixes aanwezig welke een verkorte weergave definiëren voor de namespace van een URI. Door, in plaats van de volledige namespace, deze prefixes te gebruiken, blijven namen op de gebruikersinterface kort en overzichtelijk.

Vervolgens kan eventueel voor resources die binnen de default namespace (:) vallen, gekozen worden om de dubbele punt niet te tonen. Hierdoor blijft de gebruikersinterface voor elementen die binnen de default namespace van de bronontologie bestaan, gelijk aan de huidige. Slechts elementen die buiten de default namespace vallen, krijgen dan een expliciete namespace (prefix).

Analyse van de huidige implementatie

Om een idee te krijgen van hoe de namespaces, en prefixes daarvan, binnen de huidige implementatie het best bepaald kunnen worden is een analyse gemaakt van de huidige implementatie. De huidige implementatie is gebouwd op het volgende model:



Figuur 28: Huidige implementatie

Hierbij geldt het volgende:

- De plug-in is gebouwd rond de packages ontology, fresnel en view (en wat hulp-packages die hier niet benoemd worden).
- Klassen uit de package ontology raadplegen de Protégé OWLAPI interface.
- De gele klassen leveren de labels aan de gebruikersinterface. Deze worden afgeleid uit de gedefinieerde naam van de betreffende Lens of Property (met optioneel het fresnel:label van de PropertyBinding).
- De groene klassen leveren de URI's welke de namespaces bevatten.
- De rode klasse levert de eventueel aanwezige prefixes.

Uit dit overzicht valt bovendien het volgende af te leiden:

- Klassen in de package 'ontology' representeren overeenkomstige klassen uit de OWL API interface van de actieve ontologie in Protégé.
- Klassen in de package 'fresnel' definiëren het model van de Fresnel lenzen zoals ze gebruikt worden binnen de plug-in. Elke lens is hierbij optioneel gekoppeld aan een klasse uit de ontology package en elke Property is optioneel gekoppeld aan een generieke Property uit de ontology package. Een PropertyBinding legt vervolgens de relatie tussen een Lens en een bijbehorende Property vast.
- Klassen in de package 'view' worden gebruikt om gegevens voor de GUI vast te leggen, waaronder de labels welke te zien zijn op het scherm. Elke Lensbox heeft een koppeling met een Lens uit de fresnel package en elk PropertyLabel heeft een koppeling met een PropertyBinding uit de fresnel package.
- Er zijn twee kapstok klassen:

    - In fresnel.Fresnel worden alle lenzen geregistreerd en een koppeling gelegd met de (representatie van) de bronontology ontology.Ontology.
    - In ontology.Ontology worden de (representaties van) de klassen uit de bronontologie geregistreerd en bovendien een koppeling gelegd met de OWL API interface naar de bronontologie zelf, zoals die vanuit Protégé aangeboden wordt.
- Ieder data object (ontology.Class, ontology.Property, fresnel.Property & fresnel.Lens) heeft naast een URI ook een naam.
- De naam van een object uit de fresnel klasse is bepalend voor het betreffende label op de GUI.
- De koppeling van Property en Lens klassen in de fresnel package naar klassen in de ontology package zijn optioneel. Dit is waarschijnlijk gedaan om zelf (op de GUI) Lenzen en Properties te kunnen creëren waarvoor geen klasse of property in de bronontologie bestaat.

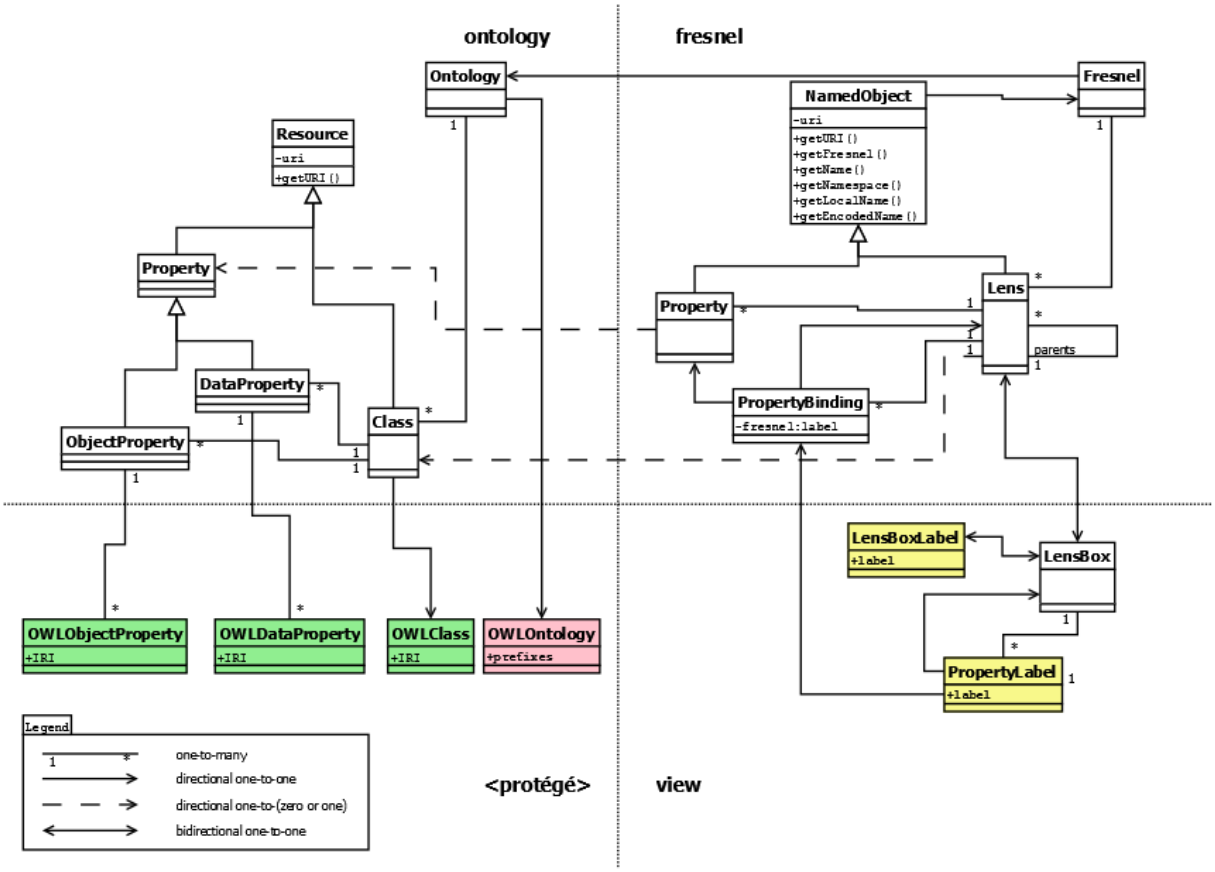
Problemen met de huidige implementatie

De grootste problemen met de huidige implementatie zijn de volgende:

- De namen van data-objecten worden initieel bepaald als zijnde de fragment identifier van een URI. Hierbij is een object op basis van zijn naam niet per definitie meer uniek te benaderen; Voor verschillende namespaces kan immers dezelfde fragment identifier bestaan.
- Vanuit de klassen in de view package kunnen op dit moment de prefixes, welke aanwezig zijn in de OWLOntology interface, niet gebruikt worden om een verkorte weergave van een URI weer te geven. Voor de bijbehorende Lenzen en Properties is namelijk niet bekend op welke ontologie ze betrekking hebben; Vanuit klassen in zowel de fresnel als de ontology klasse is de kapstokinstantie op dit moment niet te herleiden.
- Doordat de namen van de verschillende objecten gebruikt worden om de betreffende Lens- , Group- en Format-instanties in de fresnel doelontologie te bepalen is op dit moment niet gegarandeerd dat de benoemde elementen erin altijd een unieke Lens, Property, Format of Group binnen de namespace van de doelontologie beschrijven.

Oplossing

De oplossing voor de problemen, zoals hierboven beschreven, bestaat uit een implementatie van het volgende gewijzigde ontwerp:



Figuur 29: Aangepast ontwerp

Hierbij zijn de volgende wijzigingen aangebracht:

- De statische namen van alle objecten zijn verwijderd.
- De URI's van objecten zijn eenduidig bepaald.
    - Klassen in de package ontology zijn nu afgeleid van één basisklasse Resource zoals gedefinieerd aan de hand van 'Figure 4.1 Subclass relationships between OWL and RDF/RDFS' uit (Antoniou & van Harmelen, 2008).
    - Klassen in de package fresnel zijn nu afgeleid van de klasse NamedObject.
- Waar namen nodig zijn (binnen de fresnel package), worden ze in de klasse NamedObject uniek bepaald uit de URI van een object waarbij waar mogelijk gebruik gemaakt wordt van de beschikbare prefixes om de naam kort en bondig te houden.
- Om prefixes uit de bronontologie te kunnen gebruiken voor een verkorte weergave, heeft elke NamedObject instantie nu kennis van het Fresnel model waarin het bepaald is. Dit Fresnel model weet op zijn beurt op welke bronontologie het van toepassing is.
- Voor het definiëren van unieke elementen in de fresnel doelontologie is een methode getEncodedName in klasse NamedObject opgenomen die per Lens, Group en Format een unieke naam binnen de namespace van de doelontologie moet bepalen.

Het bepalen van namen

De namen van objecten binnen de plug-in worden nu als volgt bepaald op basis van de URI van het betreffende object:

1. Als voor een URI geen prefix gevonden kan worden in de bronontologie, dan wordt de volledige URI gebruikt als naam.
2. Als de namespace van een URI gelijk is aan die van de default namespace van de bronontologie, dan wordt de fragment identifier gebruikt als naam (deze is immers uniek binnen de default namespace.
3. Als voor een URI een geprefixte versie bepaald kan worden (buiten de default namespace), dan wordt deze gebruikt als naam.

Het bepalen van namen voor Lenses, Groups en Formats in de doelontologie

Omdat de elementen in de fresnel doelontologie zich allen binnen de default namespace van de doelontologie bevinden, is het voor het bepalen van de namen hiervan van groot belang om ook hier te zorgen dat deze voor elk element uniek is. Om dit goed te kunnen doen, dienen ook deze namen, welke fragment identifiers worden in de default namespace van de doelontologie, gebaseerd te zijn op de URI's van de objecten uit de bronontologie waarop ze betrekking hebben.

Omdat in fragment identifiers van URI's echter geen :-tekens mogen voorkomen, worden deze namen als volgt samengesteld:

1. Waar mogelijk wordt voor objecten waarop de Lens, Group of Format van toepassing is, de geprefixte versie van de URI ervan genomen. Dit is gedaan om, net als in de GUI, de namen waar mogelijk kort en bondig te houden.
2. In de bepaalde naam worden alle liggende streepjes (-) vervangen door twee liggende streepjes (--). Dit is gedaan om na de wijziging in de volgende stap toch altijd een unieke naam als resultaat te hebben.
3. In de naam met eventueel een even aantal liggende streepjes, wordt elk : element vervangen door een liggend streepje. Aangezien het :-teken een gereserveerd URI teken is kan het niet twee maal achter elkaar voorkomen, het wordt slechts als scheidingteken gebruikt na het scheme-part of als scheidingsteken na een prefix. (The Internet Engineering Task Force, 2015)

Bibliografie

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage IX: Cardinaliteit in Fresnel Forms.

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

Cardinaliteit in Fresnel Forms. 98
Inleiding. 100
Analyse 100
Wiki Forms 100
Vertaling 100
GUI 101
Implementatie 102
Fresnel Export 103

Inleiding.

De cardinaliteit restricties in de bronontologie dienen geimplementeerd te worden in Fresnel Forms.
De classe restricties min- en maxcardinaliteit dienen vertaald te worden naar de Semantic Wiki Forms eigenschappen list en mandatory.

Analyse

OWL kent 3 type termen om de cardinaliteit van een property te definiëren:

1. Owl:cardinality : Het exacte aantal voorkomens van een property.
2. Owl:minCardinality: Het minimale aantal voorkomens van een property.
3. Owl:maxCardinality: Het maximale aantal voorkomens van een property.

In tabel 1 worden de Limiet waardes van min- en maxcardinaliteit getoond.

Tabel 4: Restrictie waarden

Owl:minCardinality=1	Verplicht
Owl:minCardinality=0	Optioneel
Owl:maxCardinality=1	Unique
Owl:maxCardinality=0	Geen waarde toegestaan

Wiki Forms

De Wiki Forms tag ‘field’ kent de volgende properties die relevant zijn:

1. List: geeft aan of het veld een lijst is, dit is standaard het geval.
2. Mandatory: geeft aan of het veld verplicht is.
3. Max values: geeft het maximale aantal waarden aan dat voor een veld ingevuld mag worden.

Vertaling

In tabel 2 worden de voorwaarde genoemd waaronder de List en Mandatory tag toegevoegd kunnen worden aan de Fresnel ontology. De ‘A’ staat voor aanwezig in de bron ontology en ‘NA’ staat voor niet aanwezig in de bron-ontology.

Tabel 5: Vertaling bron ontology restrictions naar OWF tags.

Tag	**Waarde**	**Voorwaarde**
List	A	Owl:cardinalty > 0 of NP Owl:maxcardinality <> 1 of NP
	NA	Owl:maxcardinality = 1 Of Owl:cardinalty = 1
Mandatory	A	Owl:minCardinality=1
	NA	Owl:minCardinality=0 Of Owl:cardinalty > 0
Max values	<nr>	Owl:cardinalty = <nr>

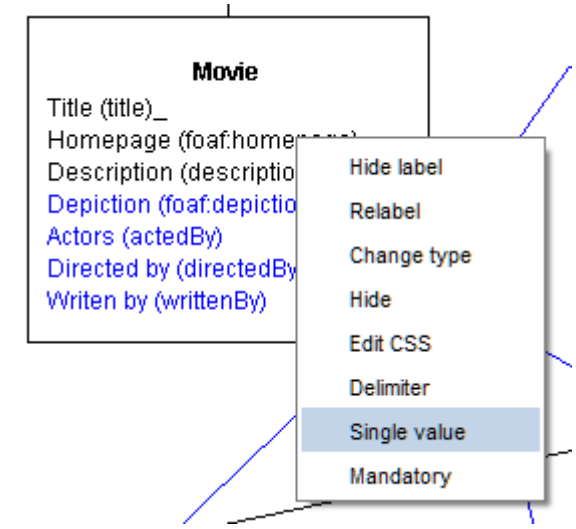
GUI

In de Fresnel Forms user interface kan per property via het rechtermenu de volgende eigenschappen ingesteld worden, de ondersteepte opties zijn de standaard waarden voor een property.
Tabel 3 toont de eigenschappen waarbij de onderstreepte opties de standaard waarden voor een property zijn. Hier staat 'NA' voor niet aanwezig.

Tabel 6: Userinterface property definitie

Lijst/Enkelvoudig	NA	Owl:maxcardinality = 1
Verplicht/Optioneel	Owl:minCardinality=1	Owl:minCardinality=0

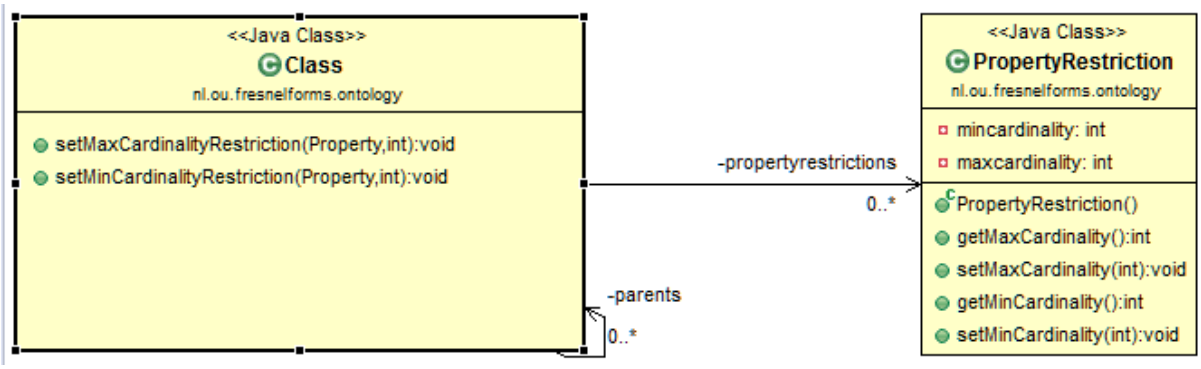
De property 'homepage' heeft de standaard waarde 'list' en 'optional'. Het rechtermuis menu toont de alternatieve opties respectievelijk 'Single value' en 'Mandatory'. Zie figuur 1.



Figuur 1: Rechtermuis menu van de propert Homepage.

Implementatie

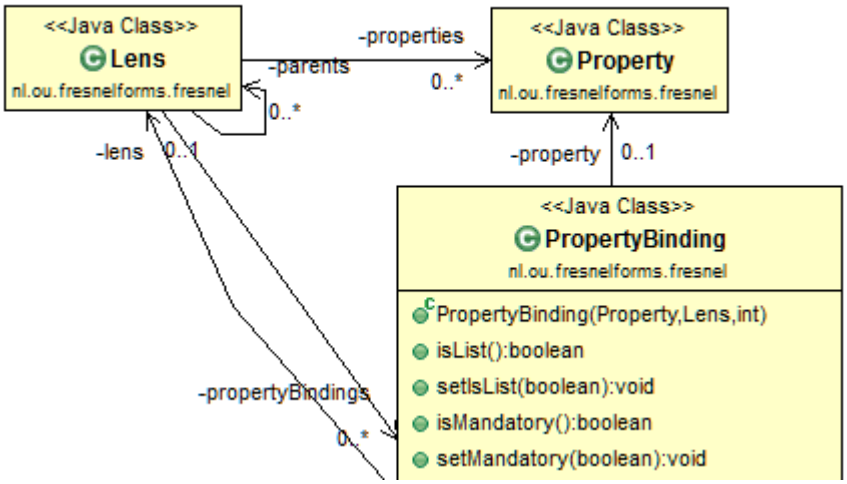
Het model voor de bron ontologie is uitgebreid met een klasse PropertyRestriction om de cardinaliteitrestricties te modeleren.
De methode `initializeOntology` van de klasse OWLImport is aangepast zodat de restricties worden geïmporteerd. De klasse Class bevat een hashmap die een property mapt op een propertyrestriction object. Dit PropertyRestriction object bevat de restricties voor die property voor de betreffende klasse. Op dit moment is alleen nog de min en max cardinaliteit restrictie geimplementeerd maar dat zou uitgebreid kunnen worden.



Figuur 31: Het klasse diagram van de PropertyRestriction klasse en de Class klasse.

Het klasse diagram laat zien dat de klasse Class methodes heeft voor de restricties die ondersteund worden.

In het Fresnel object model worden de restricties min- en maxcardinaliteit vertaald naar eigenschappen isList en IsMandatory van de klasse PropertyBinding. Een PropertyBinding object legt de eigenschappen vast die een property heeft voor een lens, zoals een label en autocomplete.
De klasse Lens heeft een lijst van propertybinding objecten, De propertybinding klasse heeft methodes voor elke eigenschap die deze definieerd. Zoals figuur 3 laat zien heeft de PropertyBinding klasse de methode isList(), setIsList(), isMandatory() en setMandatory().



Figuur 32:Klassediagram Lens, Property en PropertyBinding klasse.

De constructor van het PropertyBinding klasse bepaald de waarde van deze eigenschappen met het volgende stukje code. De bron klasse wordt bepaald via de lens methode

getClassLensDomain. Als de bron klasse restrictiedefinities heeft dan wordt het propertyrestriction object opgevraagd. Via het propertyrestriction object wordt de cardinaliteit opgevraagd.

```
//init restrictions if there are any
nl.ou.fresnelforms.ontology.Property dtp = property.getProperty();
if (lens.getClassLensDomain()!=null){
                        nl.ou.fresnelforms.ontology.Class cls = lens.getClassLensDomain();
                        if (cls.hasRestrictions()){
                                PropertyRestriction propres = cls.getPropertyrestrictions().get(dtp);
                                if (propres != null){
                                        this.isList = !( propres.getMaxCardinality() == 1);
                                        this.mandatory = (propres.getMinCardinality() == 1);
                                }
                        }
```

Vanuit het Fresnel model kan nu de Semantic Wiki Forms code bepaald worden.

Fresnel Export

Om de uitbreiding op het Frensel object model te kunnen exporteren naar een Fresnel ontology bestand dient de Fresnel vocabulaire uitgebreid te worden met de isList en isMandatory properties.
De uitbreidingen op de Fresnel ontologie worden gedefinieerd in de OWF_Style.owl ontologie zoals hieronder weergegeven.

```
### http://is.cs.ou.nl/OWF/OWF_style#isList
:isList rdf:type owl:DatatypeProperty ;
      rdfs:domain fresnel:Format ;
      rdfs:range xsd:boolean .

### http://is.cs.ou.nl/OWF/OWF_style#isMandatory
:isMandatory rdf:type owl:DatatypeProperty ;
          rdfs:domain fresnel:Format ;
          rdfs:range xsd:boolean .
```

Deze OWF_Style properties zijn in het FresnelForms project gedefinieerd in de package ‘nl.ou.fresnelforms.vocabulary’ en java klasse OWF.java zoals hieronder weergegeven.

```
/**
 * A OWF's Format's isList property.
 */
public static final Property ISLIST = property("isList");

/**
 * A OWF's Format's isMandatory property.
 */
public static final Property ISMANDATORY = property("isMandatory");
```

De protected methode ‘property’ geeft de complete URI van de property door deze samen te stellen met de namespace uri van de owf_style ontology.

```
/**
 * The Presentation's URI.
 */
protected static final String URI = "http://is.cs.ou.nl/OWF/#";

/**
 * Returns a RDF Property for this vocabulary.
 *
 * @param local the local name of the property
 * @return the RDF Property
 */
protected static final Property property(String local) {
        return ResourceFactory.createProperty(URI, local);
}
```

Bijlage X: DBpedia in Fresnel Forms plugin.

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

DBpedia in Fresnel Forms plugin. 105
Inleiding 107
Analyse 107
Situatie 107
Knelpunten 107
Oplossing 108
User interface aanpassingen 108
Conclusie 109

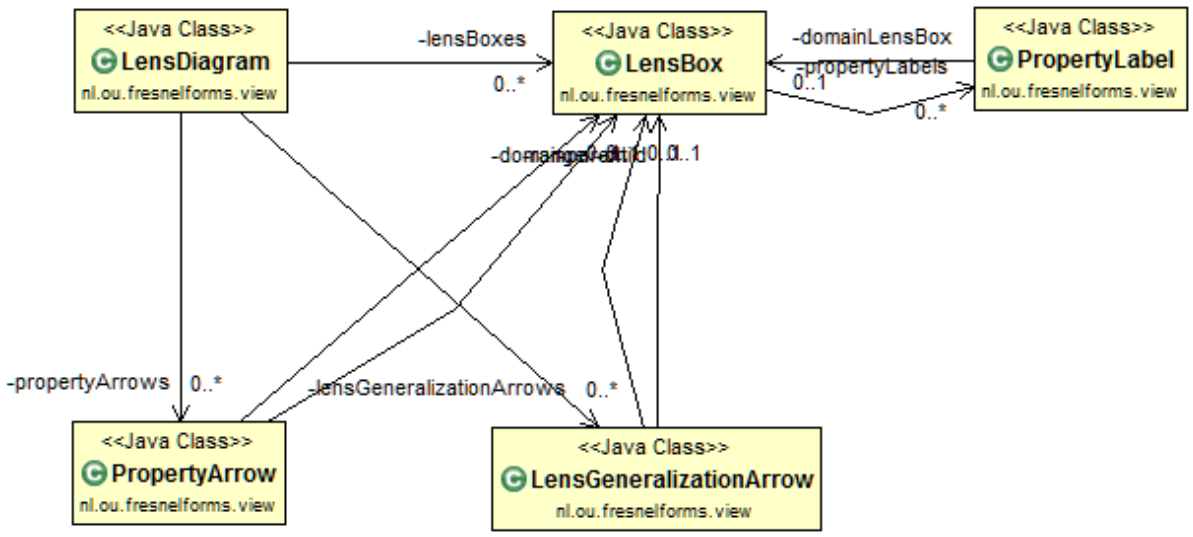
Inleiding

Om de vraag van de opdrachtgever te kunnen beantwoorden of DBpedia ingelezen kon worden in de Fresnel Forms plug-in hebben we dit getest. Het bleek dat de DBpedia ontologie wel ingelezen kon worden maar ook dat de user interface onwerkbaar traag werd. In dit document wordt de analyse en de oplossing voor dit probleem beschreven. Nadat de oplossing de user interface weer werkbaar maakte zijn nog enkele kleine aanpassingen doorgevoerd die het werken met zeer grote ontologieën vergemakkelijken.

Analyse

Situatie

Om de oorzaak van de traagheid van de user interface te analyseren is de flow van de draw events van de verschillende klassen geanalyseerd. In fig. 1 staan de klasse die hierbij een rol speelde.



Figuur 33: Klassendiagram Fresnel Forms user interface.

We zien dat de klasse LensDiagram de klasse LensBox, PropertyArrow en LensGeneralizationArrow beheert. De beide klassen PropertyArrow en LensGeneralizationArrow zijn gerelateerd aan de twee lensboxes waar tussen de pijl getrokken moet gaan worden. De Lensbox beheert de property labels van de lens.

Een aanroep van het draw event van de klasse LensDiagram tekent het gehele Fresnel diagram bestaande uit de Lensen, de propertylabels, de pijlen naar de lenzen voor de object properties van de actieve lens en de hiërarchie pijlen tussen de lenzen.

Knelpunten

Uit de analyse van de draw events zijn de volgende knelpunten naar voren gekomen.

1. De LensDiagram.draw methode initialiseert de PropertyArrow en LensGeneralisationArrow structuur elke keer als er een draw event plaats vind.

2. De lijnen worden bij elk draw event opnieuw berekend terwijl dit alleen maar hoeft als er een lensbox of propertlabel verplaatst is.
3. De propertylabels worden bij elk draw even van de Lensbox opnieuw gesorteerd door de Lensbox methode getPropertyLabels().

Oplossing

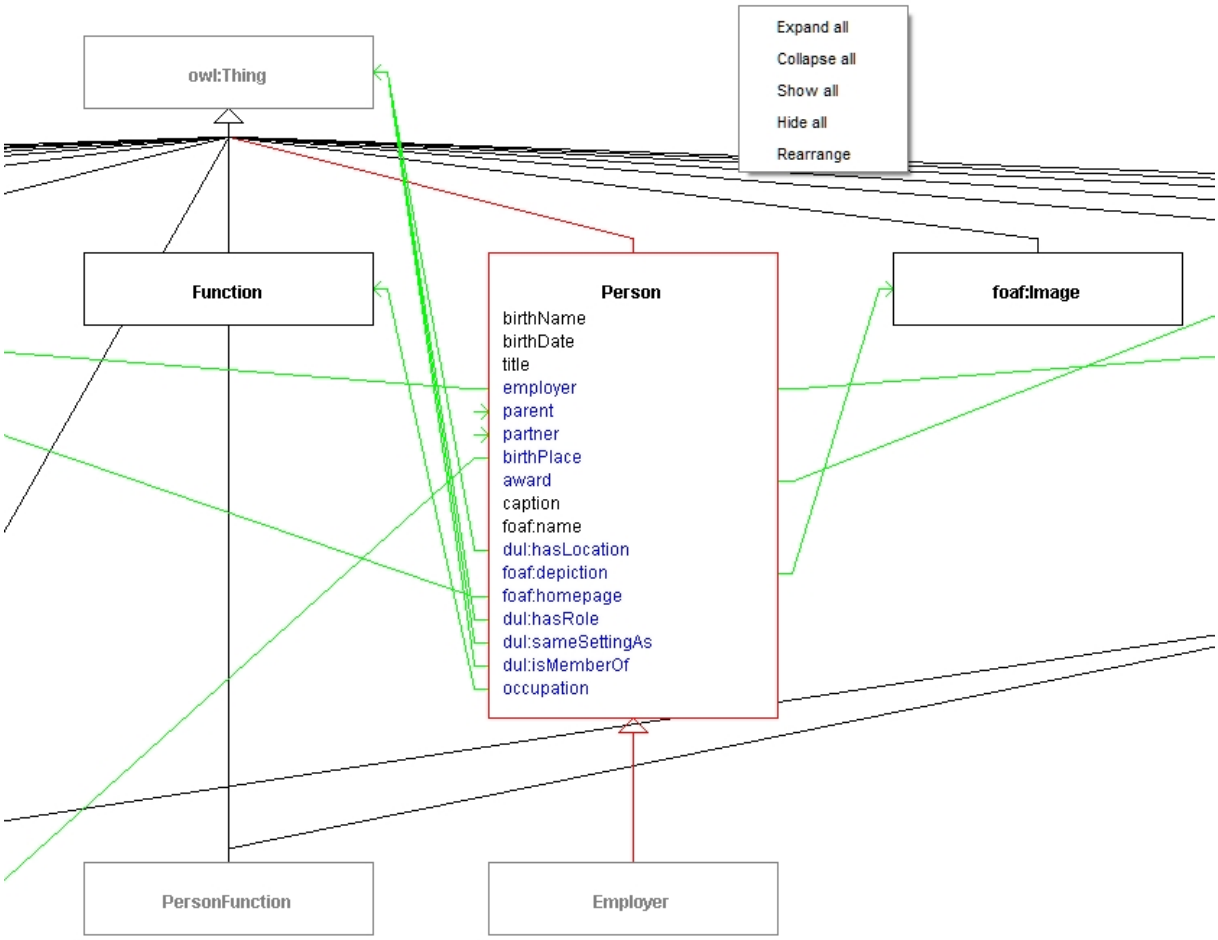
Voor bovengenoemde knelpunten zijn de volgende oplossingen mogelijk.

1. De PropertyArrow en LensGeneralisationArrow structuur worden één keer opgezet tijdens de initialisatiefase die aangegeven wordt door de boolean object variabele '`init`' in het draw() event van de klasse LensDiagram.
2. De lensboxes worden voorzien van de boolean object variabele '`lensdirty`'. Deze variabele wordt geset als een lens verplaatst wordt of als de property labels opnieuw gesorteerd worden. De PropertyArrow en LensGeneralisationArrow klassen zijn voorzien van een refresh() methode die controleert of één van de twee lensboxes dirty is, alleen indien dat het geval is wordt de pijl opnieuw berekend. Het draw event van de LensDiagram reset de object variabele '`lensdirty`' van de getekende lens.
3. De methode getPropertyLabels() retourneerd de propertylabel listarray zonder deze te sorteren. Er zijn 2 menuitems toegevoegd aan het rechtermuismenu van de lensbox waarmee de gebruiker de properties kan sorteren. De listarray wordt dan opnieuw opgebouwd met propertylabel objecten in de juiste volgorde.

User interface aanpassingen

Het tekenen van alle propertylabels en de lijnen tussen de properties en de lensen vormen de grootste belasting. Daarom is besloten om alle lenzen gesloten te tekenen dat wil zeggen zonder de propertylabels. Een gebruiker kan met het rechtermuis menu een lens openen en sluiten.

Het algemene rechtermuis menu biedt de optie om alle lenzen te openen of te sluiten. Voor een uitgebreidere instructie van de user interface-wijzigingen zie de bijlage voor het document 'User interface instructie.docx' **Invalid source specified.**.



Figuur 34: Vernieuwde Fresnel Forms user interface.

Conclusie

Bijna de gehele objecten structuur voor het LensDiagram werd bij elk draw event opnieuw opgebouwd waardoor het enorm traag werd bij grotere ontologieën. Door deze opbouw te beperken tot de situaties waarbij er iets veranderd is wordt de uitvoering van het draw event enorm versneld.
De DBpedia ontologie kan nu in de FrenselForms plugin worden ingelezen.

Het blijkt dat een objectstructuur opbouwen een zware actie is terwijl het uitlezen van de object-structuur heel snel gaat.

References

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (25 de 01 de 2015). *Fresnel - Display Vocabulary for RDF*. Obtenido de W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (01 de 02 de 2015). *Cascading Style Sheets, level 2 CSS2 Specification*. Obtenido de W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (pág. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (12 de 13 de 2014). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (s.f.). *Help:Import vocabulary*. Recuperado el 3 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (s.f.). *Help:RDF Export*. Recuperado el 2 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (págs. 1-27). Springer.

The Internet Engineering Task Force. (17 de 01 de 2015). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Obtenido de https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (págs. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (págs. 1-10). Bucharest: Matrix ROM.

W3C. (01 de 05 de 2015). *Fresnel - Display Vocabulary for RDF*. Recuperado el 11 de January de 2015, de W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage XI: RDF export Fresnel2Wiki

Joop van de Heijning
Open universiteit

Introductie

Een belangrijke requirement voor Fresnel Forms is dat het mogelijk is om van de naar een wiki geüploade ontologie de RDF triples te exporteren met de Semantic MediaWiki (SMW) special page special:ExportRDF voor RDF export (Semantic MediaWiki). Dit verslag geeft een beknopte uitleg van het proces.

De Semantic MediaWiki-kant

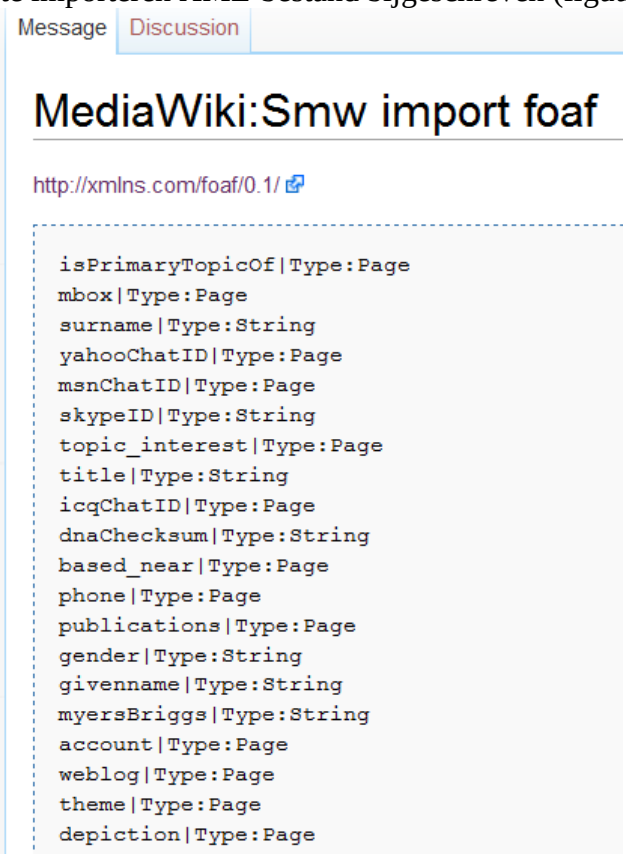
RDF export kan de SMW-attributen en relaties exporteren als properties in de externe vocabulaire (zoals foaf:knows). Daarvoor moet die externe vocabulaire dan wel eerst in de semantische wiki geïmporteerd zijn (Semantic MediaWiki). Dit kan gedaan worden door er eerst voor te zorgen dat elke te importeren property pagina op de wiki de tag (met als voorbeeld foaf:knows) `[[imported from::foaf:knows]]` bevat. Daarnaast moet er een pagina in de MediaWiki namespace met prefix `smw_import_` bestaan (Semantic MediaWiki). Deze pagina bevat de URI van de externe vocabulaire en het datatype per property, zie figuur 1 uit onze testwiki.

De Fresnel2Wiki-kant

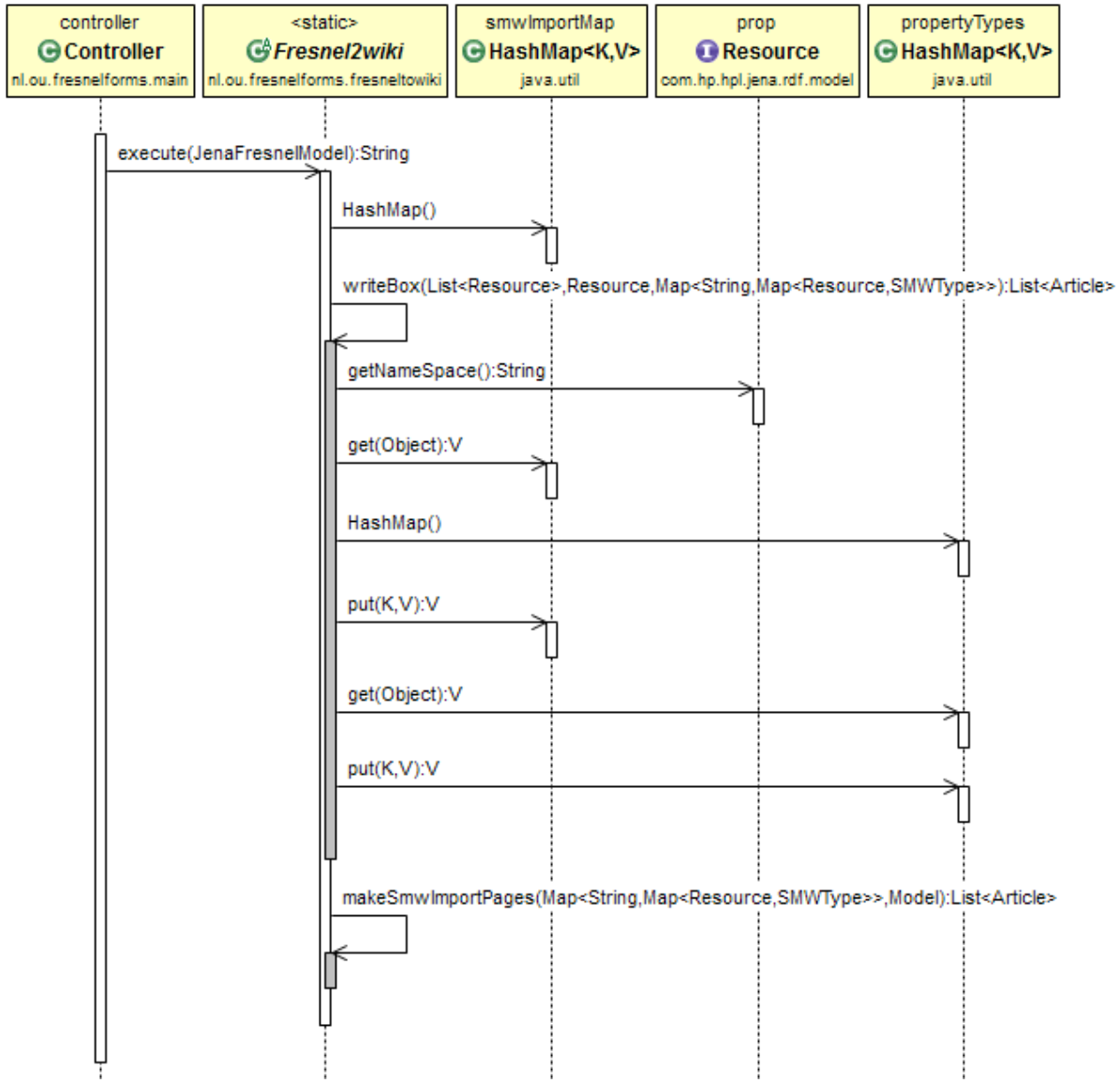
Methode `execute()` in de Fresnel2Wiki klasse maakt eerst een hashMap `smwImportMap` aan en geeft die mee aan methode `writeBox()`. In `writeBox()` worden per lens alle properties doorlopen. Per property wordt bekeken of voor de namespace (bijvoorbeeld foaf) al een map `propertyTypes` (submap van `smwImportMap`) bestaat door deze bij `smwImportMap` op te vragen met `get(namespace)`. Zo niet, dan wordt deze aangemaakt. Vervolgens wordt bekeken of `propertyTypes` al de desbetreffende property bevat, zo niet, dan wordt deze toegevoegd (deze test wordt ook gebruikt om te kijken of deze property al een hoofd-pagina heeft).

Op deze manier worden per namespace alle properties toegevoegd in `smwImportMap`. Aan het einde van methode `execute()` worden dan alle smw_import_ pagina's in het in de wiki

te importeren XML-bestand bijgeschreven (figuur 2).



Figuur 35 Voorbeeld smw_import pagina



Figuur 36 Fresnel2Wiki sequence diagram vereenvoudigd

Uitzonderingen

Special:ExportRDF gebruikt semantische links zoals `[[foaf::knows]]` in de wikicode op een sjabloon-pagina op de wiki. Voor properties met type image hebben wij deze link moeten vervangen voor een HTML img tag: `<img src='xxx'/>`. Vandaar dat het nodig was om in deze gevallen een “dummy” div toe te voegen aan de property div op een sjabloon-pagina met semantische link voor de export. In deze dummy div was het ook belangrijk om de arraymap functie van Semantic Forms **Invalid source specified.** te gebruiken voor lijsten, en juist niet voor properties met enkelvoudige waarden. Zo kan de RDF export functie onderscheid maken tussen enkele of meerdere waarden.

Referenties

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage XII: Heuristic based ranking of properties.

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

Heuristic based ranking of properties. 118
Inleiding. 120
Infobox heuristic sort methode 120
In bucket ordening 123
Term frequentie ordening 123
Range rank calculation 123
Opzet inplementatie van het algorithme 124
Inplementatie in Java 124
Alfabetische sortering implementeren. 125
Heuristic sortering implementeren. 125
Klasse opzet 125
Informatie benodigdheden voor de heuristic 125
Range rank berekening 126
Inventarisatie van de gegevens. 127
Implementatiewijzigingen 127
References 128

Inleiding.

Dit document beschrijft hoe de heuristic sort methode zoals het wordt gepresenteerd in de master scriptie van Paul Falco **Invalid source specified.** geïmplementeerd is in de Fesnel Forms plug-in.

De master scriptie van Paul Falco evalueert van verschillende sorteerd algoritmes welke het beste lijkt op de sorteer volgorde van de infoboxes van wikipedia. De heuristic methode komt het dichtst hierbij in de buurt.

Infobox heuristic sort methode

Deze heuristic methode bestaat uit een aantal algemeen geldende vuistregels. Deze regels worden hieronder beschreven:

In tabel 1 staan de regels en de volgorde waarin de regels uitgevoerd moeten worden.

Tabel 7: De sorteer regels.

Volgorde	Bucket	Omschrijving
1	A	Any property which contains one of the following strings in its label is promoted to bucket *A*: ☐ 'name' ☐ 'code' ☐ 'identifying' ☐ 'identifier' ☐ 'label'
2	E	Any properties from the following ontologies are promoted to bucket *E*: ☐ foaf:name ☐ foaf:nick ☐ foaf:givenName ☐ foaf:givenname ☐ foaf:familyName ☐ foaf:family_name ☐ foaf:firstName ☐ foaf:lastName ☐ foaf:surname ☐ foaf:topic ☐ http://schema.org/alternateName ☐ http://schema.org/name ☐ dc:title ☐ dc:subject

		☐ dc:identifier
3	F	Properties from the following ontologies will be promoted to bucket *F*: ☐ foaf:homepage ☐ foaf:isPrimaryTopicOf ☐ foaf:primaryTopic ☐ foaf:knows ☐ foaf:made ☐ foaf:maker ☐ foaf:page ☐ foaf:gender ☐ http://schema.org/description ☐ http://schema.org/sameas ☐ http://schema.org/url ☐ http://schema.org/about ☐ http://schema.org/category ☐ http://schema.org/object ☐ dc:creator ☐ dc:description ☐ dc:type ☐ dct:alternative
4	B	If the property is any of the following XSD date data types then bucket B is assigned: ☐ XSDdate ☐ XSDdateTime ☐ XSDtime ☐ XSDgYearMonth ☐ XSDgYear ☐ XSDgMonthDay ☐ XSDgMonth ☐ XSDgDay
5	C	We then look at the range of the property and compute a metric that we call the range rank. If a range rank has been assigned than the property is promoted to bucket *C*. The actual range rank becomes the immediate next level of ordering within bucket *C* (popular ranges are ranked higher).

		See below for a calculation of the range rank.
6	D	Properties that are either functional or inverse functional are assigned to bucket *D*.
7	Z	Any other properties are assigned to bucket *Z* (these will be the lowest ranking properties).

In bucket ordening

De in bucket ordering vindt plaats op basis van de property label.

Tabel 8: De in-bucket sorteer regels.

Volgorde	Omschrijving
a	The highest priority (*a*) is assigned to properties that have labels containing any of the following words: “start”, “begin”, “birth”, “former”, “previous”, “opening” or “predecessor”.
b	The second priority (*b*) is set for those properties where the label contains either “stop”, “end”, death”, “future”, “subsequent”, “closing” or “successor”.
c	The lowest priority (*c*) is designated to all properties that do not fall in one of the first two categories.

Term frequentie ordening

Binnen de bucket worden gelijke geordende termen nog geordend op frequentie. Frequenter gebruikte termen komen eerst. Bij gelijke frequentie wordt een alfabetische volgorde toegepast.

Range rank calculation

Nadat een property in een bucket op de juiste volgorde is geplaatst wordt de rank berekend. Voor dataproperties volgt een ranking van 0 voor de onderstaande XSD datatypes.

☐ `XSDbyte`

☐ `XSDdouble`

☐ `XSDfloat`

☐ `XSDdecimal`

☐ `XSDboolean`

☐ `XSDint`

☐ `XSDshort`

☐ `XSDstring`

☐ `XSDunsignedByte`

☐ `XSDunsignedInt`

☐ `XSDunsignedLong`

☐ `XSDunsignedShort`

☐ `XSDnonNegativeInteger`

☐ `XSDnegativeInteger`

☐ `XSDnonPositiveInteger`

☐ `XSDpositiveInteger`

☐ `XSDduration`

Voor object ranges wordt wordt de rank in 2 stappen berekend.

1. Er wordt een geordende lijst aangemaakt voor alle klasse in de ontologie.
   Voor elke van de klasse wordt bepaald hoevaak ze in een range gebruikt worden. De ordening in de lijst loopt van klasse die het meest naar de klasse die het minst gebruikt worden in een range.
2. De range rank van een property wordt berekend door van de range object een lijst te maken van de klasse in de overervings hiërarchie. De range rank is nu de klasse uit de overervings hiërarchie met de hoogste populariteit. De index uit de rangeclasses list is de rangerank. Op deze manier zou een lijst samengesteld kunnen worden van de properties met de range rank.

Opzet implementatie van het algoritme

Het rechtermuismenu van de lens krijgt twee sorteer opties: alfabetisch sorteren en heuristic sorteren. Dit om het verschil te zien tussen beide methodes.

Voor de sorteer actie wordt een nieuwe comparator opgezet die twee property objecten kan vergelijken. De twee property objecten worden voorzien van de bucket waarin ze vallen. Zijn de buckets ongelijk, dan bepaalt het ordinal getal van de buckets de uitkomst van de comparator. Zijn de buckets gelijk dan wordt de ordening binnen de buckets bepaalt en die bepaald dan de uitkomst van de comparator.

Implementatie in Java

De properties van een lens worden weergegeven door de LensBox klasse.
Elke lensbox heeft een arrayList ‘propertylabels’.

De sortering wordt uitgevoerd door de statische methode Collections.sort().
Deze methode verwacht een vergelijker (Comparator). Deze vergelijker bepaalt de sorteervolgorde.

De sort methode wordt in de LensBox klasse in de volgende methoden gebruikt:

- `changeIndexOfPropertyLabel(PropertyLabel propertyLabel)`
- `getPropertyLabels()`

De `PropertyLabelIndexComparator()` wordt gebruikt om de propertylabel array op de index waarde te sorteren.

De index kan gewijzigd worden door :

- verslepen van de property.
- het toevoegen van een property.

Welke methode gebruiken de propertyindex variabele:

- `positionPropertyLabels()`
- `changeIndexOfPropertyLabel(PropertyLabel propertyLabel)`

De propertylabel listarray wordt gevuld in de LensBox klasse met de methode addPropertyLabel(). Deze methode roept de methode positionPropertyLabels aan die de propertylabels in de array op het scherm zet op basis van de index waarde. De index waarde bepaald de hoogte waarop de label geplaatst wordt. De index is opgeslagen in het propertybinding object.

Alfabetische sortering implementeren.

Op basis van de label, default krijgt de label van een property als waarde de naam zonder prefix.
Deze sortering kan geimplementeerd worden door een Comparator te definieren voor de alfabetische volgorde. Deze volgorde zit in Java String klasse ingebouwd.

Heuristic sortering implementeren.

Er dient een klasse gedefinieerd te worden die de bucket structuur opbouwt. De LensBox is de eigenaar van deze structuur. Er dient een comparator gedefinieerd te worden die op basis van deze structuur de relatie van twee labels of twee namen van de resources bepaalt.

De bucket structuur bestaat uit een array met 7 elementen die de buckets voorstellen. Elk element bestaat ook weer uit een array die de inbucket volgorde voorstelt. De waarde van het element van de tweede array is de naam of de label van de resource. Het is dus een string array.

De lengte van de inbucket array is niet van te voren bekend en hangt af van de klasse hiërarchie waarin de lens zich bevindt.

Uit deze bucket structuur dient een mapping gedestilleerd te worden die het mogelijk maakt om snel de naam of label van de resource te koppelen aan de volgorde.

De klasse heeft dan een methode getOrder(propertyname) die de ordening als getal terug geeft.
Voor dat getal kiezen we een float type. Het getal voor de komma geeft de bucket aan, de getallen achter de komma geven de inbucket ordering aan. Alle buckets hebben min. 3 niveaus behalve bucket C. Het aantal niveaus hangt daarbij af van de klassehiërarchie.
bv:

1.1 bucket A level 1

1.2 bucket A level 2

1.3 bucket A level 3

5.1 bucket E level 1

3.05 bucket C level 5 van de 100

3.75 bucket C level 75 van de 100

Klasse opzet

De klasse LensBox krijgt een methode ‘sortPropertyLabelHeuristic()’.
Een nieuwe klasse ‘PropertyLabelHeuristicComparator’, deze klasse wordt in de klasse LensDiagram geinstantieerd en geinitialiseerd voor een LensBox. Deze instantie wordt dan meegegeven aan de LensBox instantie, door een constructor of door een setter.

De klasse ‘PropertyLabelHeuristicComparator()’ bevat de bucketstructuur zoals gedefinieerd door Falco Paul, de klasse ‘FalcoPaulHeuristicSortBucket’

De klasse ‘PropertyLabelHeuristicComparator()’ implementeert de methode compare(). Deze methode maakt gebruik van de bucketstructuur klasse methode getOrder(property) om een floating getal te krijgen die de ordening aangeeft.

De klasse LensDiagram krijgt een private methode om de FPHSBucket te initialiseren voor de properties van een LensBox.

Informatie benodigdheden voor de heuristic

1. Label van de property.
2. Prefix van de propertynaam.
3. Uri van de property.
4. Het XSD datatype van een dataproperty.
5. De range rank van een property.
6. Is een property functional of inverse functional.
7. Label/property frequentie.

Range rank berekening

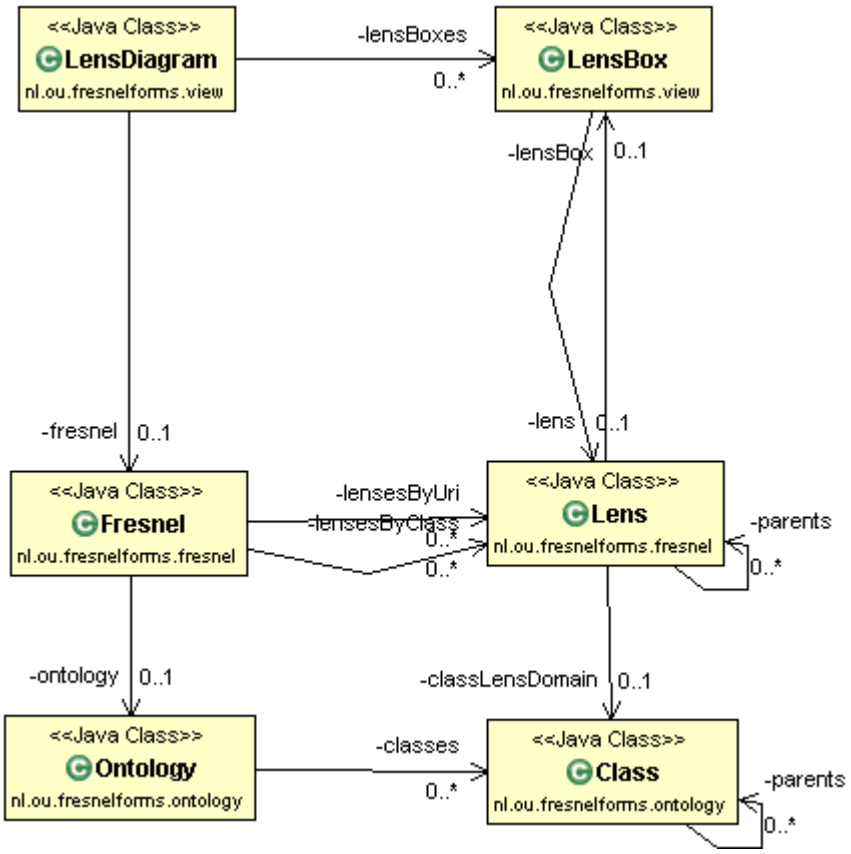
Wordt berekend op basis van de range van een property.
De dataproperties met xsd type krijgen een rank van 0.
Van de object properties wordt het gebruik binnen de ontologie bepaalt.

Er wordt een rangeclass frequentie lijst opgesteld door te bepalen hoe vaak een klasse voorkomt in een range definitie van een property. Deze bepaling wordt gedaan voor alle properties in de ontologie.
Deze lijst wordt aflopend gesorteerd dus populaire klassen eerst.

Per property wordt een hierarchielijst samengesteld van de klasse waarnaar verwezen wordt. Deze hierarchielijst bevat dus alle superklassen van de range klasse. (niet duidelijk is of de range klasse zelf ook meegenomen wordt in deze lijst.)

Om de rank range vast te stellen wordt de rangeclass frequentielijst afgelopen van meest populair naar minder populair. Voor elke klasse wordt gecontroleerd of deze voorkomt in de hiërarchie lijst als dat zo is is de range rank gelijk aan de index in de rangefrequentielijst.

Classe diagram ontology_hierarchie.gif



Figuur 37: Klasse hiërachie Ontology, Fresnel en LensDiagram

Bij initialisatie van de lensboxen in het lensdiagram wordt de heuristics informatie doorgegeven aan de constructor van de LensBox. Zodat de lensbox de sortering kan uitvoeren.

Inventarisatie van de gegevens.

Overzicht van de aanwezigheid van de gegevens die nodig zijn voor de heuristic sortering.

1. De labels worden niet uit de bronontologie uitgelezen maar worden geïnitialiseerd op de naam van de property.
2. De prefix van de property URI is niet bekend. De URI van de property is opgeslagen in de Resource klasse die die zou de prefix terug kunnen geven dmv de URI klasse en de getScheme() methode.
3. De URI van de property is bekend en is opgeslagen in de property klasse.
4. Het XSD datatype is niet bekend en zou verkregen kunnen worden door de datatype property klasse.
   dmv. Het dataRangetype die een owldatatype object terug kan geven. Dit object kan een XSD enumeration object terug geven.
5. De rangeclassfrequentielijst is niet bekend en zou opgebouwd moeten worden in de ontology klasse.
   De rangeklasse hiërarchie is bekend. Elke klasse heeft een parents array.
6. Het functional of inverse functional zijn van een property is niet bekend van een property.
7. De property frequentie lijst is nog onbekend en moet door de ontology klasse bepaald worden.

Implementatiewijzigingen

1. De labels worden uit de bronontologie gelezen en aan het property object toegekend in de OWLImport methode met behulp van de `findPropertyLabel` methode.
2. In de klasse resource is de String uri vervangen door de klasse URI. Er is een methode toegevoegd getPrefix die het schema gedeelte van de URI terug geeft. Dus van foaf:name wordt foaf terug gegeven.
3. De methode getDatatype toegevoegd aan de dataproperty klasse.
4. Properties functional en inversefunctional toegevoegd aan de property klasse. De owldataproperty klasse heeft wel een isfunctional methode maar geen isInverseFunctional methode. De owlObjectProperty ondersteunt beide methoden. Default zijn beide properties false.

References

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (25 de 01 de 2015). *Fresnel - Display Vocabulary for RDF*. Obtenido de W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (01 de 02 de 2015). *Cascading Style Sheets, level 2 CSS2 Specification*. Obtenido de W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (pág. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (12 de 13 de 2014). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (s.f.). *Help:Import vocabulary*. Recuperado el 3 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (s.f.). *Help:RDF Export*. Recuperado el 2 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (págs. 1-27). Springer.

The Internet Engineering Task Force. (17 de 01 de 2015). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Obtenido de https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (págs. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (págs. 1-10). Bucharest: Matrix ROM.

W3C. (01 de 05 de 2015). *Fresnel - Display Vocabulary for RDF*. Recuperado el 11 de January de 2015, de W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage XIII: Fresnel Forms user interface

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inleiding

De user interface van de plug-in is aangepast zodat het mogelijk is om de DBpedia ontologie te laden, lenzen te selecteren en deze selectie te exporteren naar wiki formaat.

Samenvatting

De volgende aanpassingen zin gemaakt aan de bestaande user interface.

1. Bij het laden van de ontologie worden de lenzen worden zonder attributen getoond.
2. Het rechtermuismenu van de lensbox heeft een optie om een lensbox uit en in te klappen.
3. Met de linkermuis knop kan een lens geselecteerd worden. De lensbox en de hiërarchie pijlen worden rood gekleurd. De attribuut lijnen worden groen gekleurd.
4. De grootte van het canvas is afhankelijk van het schema en heeft de minimale grootte die nodig is om het schema te tonen.
5. Er is een extra rechtermuis knop menu voor het canvas met de mogelijkheid om
    a. alle lenzen in of uit te klappen.
    b. Alle lenzen actief of niet actief te zetten.
    c. Het schema opnieuw laten berekenen.

Instructie voor testen van de user interface:

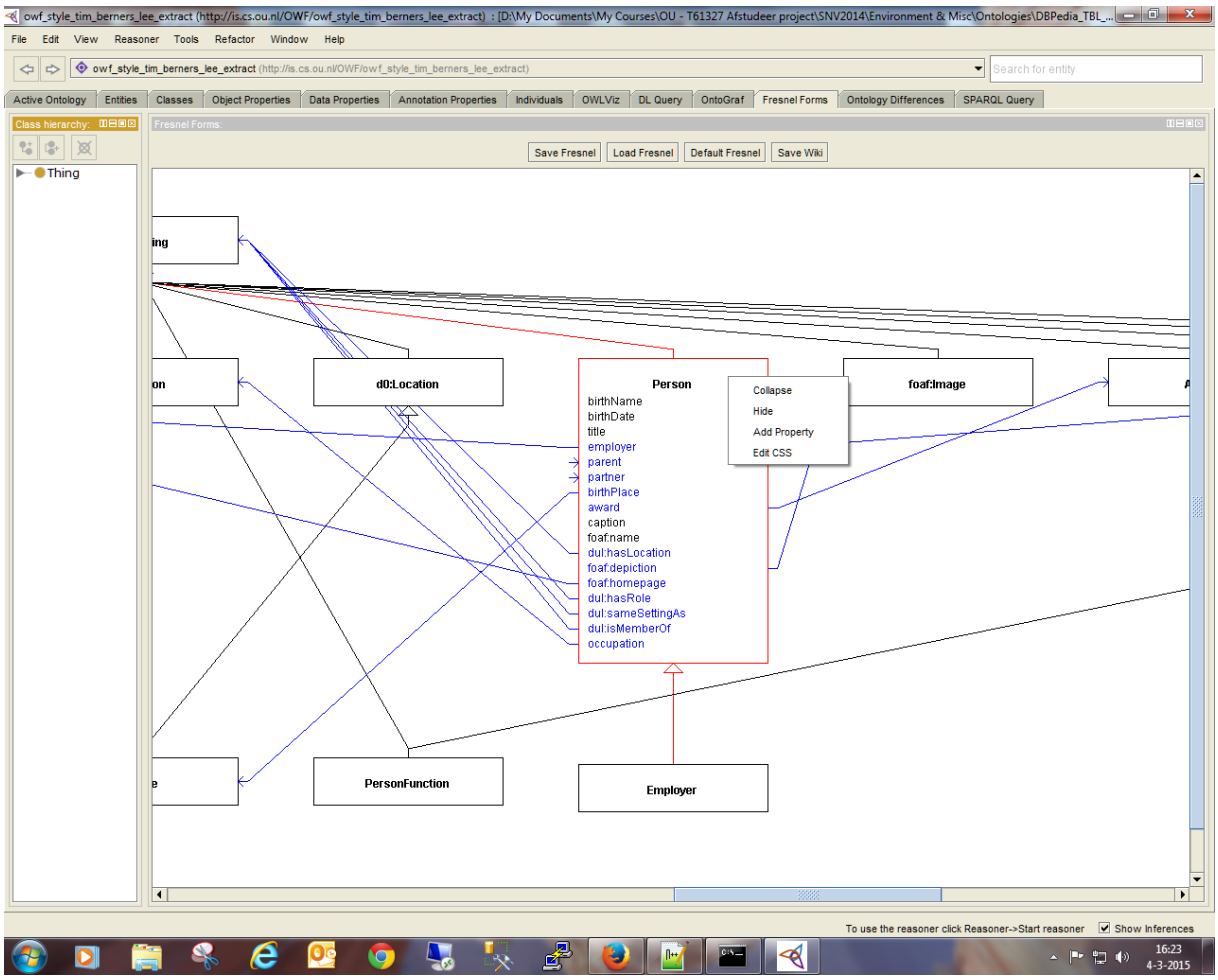
Volg onderstaande stappen om de nieuwe mogelijkheden te ervaren.

1. Laad de ‘owf_style_TBL_extract.owl’ ontologie in Protégé.
2. Activeer de tab ‘Fresnel Forms’.
3. Klik op de knop ‘Default fresnel’.
4. Klik met de rechtermuis knop ergens op het witte vlak.
5. Kies uit het menu 'Expand all', alle lensboxen klappen uit.
6. Klik met de rechtermuis knop ergens op het witte vlak.
7. Kies uit het menu 'Collapse all', alle lensboxen klappen in.
8. Klik met de rechtermuis knop ergens op het witte vlak.
9. Kies uit het menu 'Hide all', alle lensboxen worden grijs.
   De hidden lensboxen worden niet geexporteerd naar het wiki XML formaat.
10. Kies uit het menu 'Show all', alle lensboxen worden zwart.

11. Klik met de rechtermuis knop op de lensbox ‘Person’.
12. Kies menuitem 'Expand', de lensbox laat alle attributen zien.
13. Klik met de rechtermuis knop ergens op het witte vlak.
14. Kies uit het menu 'Rearrange', het schema wordt opnieuw berekend en getoond.

15. Klik met de linkermuis knop in de Person lensbox.
16. Het kader van de lensbox wordt nu rood en de hierarchielijnen naar de ouder en kind lensboxen ook.

Het resultaat zou moeten zijn:



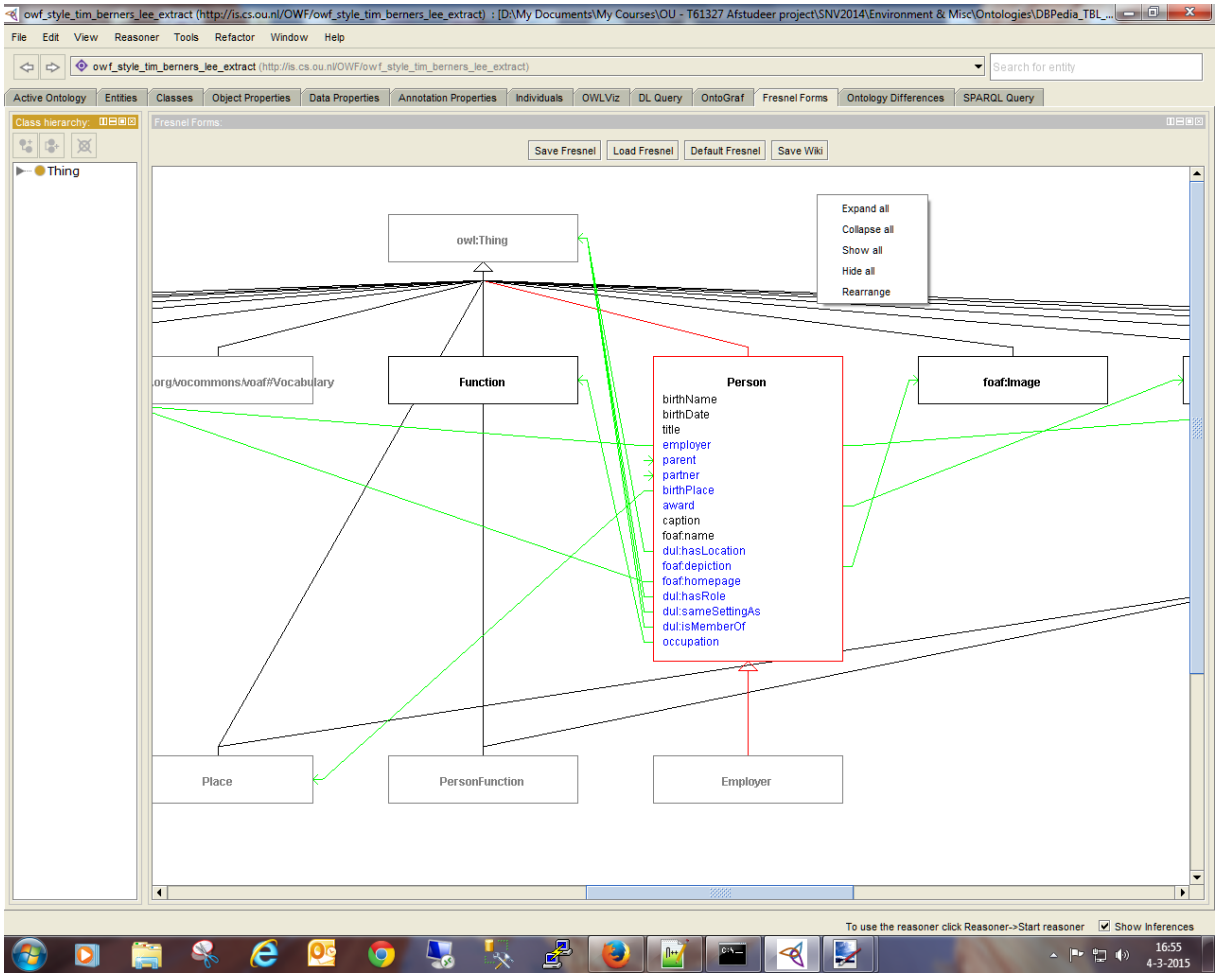
Figuur 38: Een geselecteerde lens met rechtermuismenu.

Vervolg instructie

Vervolg met onderstaande stappen.

1. Beweeg de muis van de lensbox af en de attribuut lijnen veranderen van blauw naar groen
   en blijven zichtbaar.
2. Klik met de rechtermuis knop ergens op het witte vlak.
3. Kies uit het menu 'Hide all', alle lensboxen worden grijs.
4. Klik met de rechtermuis knop in de Person lensbox.
5. Kies menuitem 'Show' to enable deze lensbox.
6. Klik met de rechtermuis knop in de Function lensbox.
7. Kies menuitem 'Show' to enable deze lensbox.
8. Volg de groene lijnen en enable de bijbehorende lensboxen en
   herhaal dit voor de lensboxen: 'http://schema.org/Organisation', 'foaf:Document', 'foaf:image', 'Award'

Het resultaat zou moeten zijn:



Figuur 39: Geselecteerd lensbox met attribuut lijnen en het rechtermuismenu voor het canvas.

Klik op de knop 'save Wiki' om voor de 'Person' lens met zijn refererende lenzen de wikipagina's te
genereren.

Op deze manier kan dus ook voor een geselecteerde lens uit de DBpedia een wiki pagina gemaakt worden. Vanwege de omvang van de Dbpedia ontologie is het niet raadzaam om voor 'Expand all' te kiezen, het systeem wordt dan onwerkbaar traag.

Bijlage XIV: Statustabel Fresnel Forms

De tabel, welke in dit document is opgenomen, geeft de status weer van de implementatie van Fresnel Forms. Deze tabel is opgesteld door prof. Lloyd Rutledge.

Ontology				Fresnel	MediaWiki and extensions			Description	
Foundation	URIs	Whole			[[EquivalentURI::...]] SMW			Ontology URI's used in RDF export	
		Namespace	Identifier		Identifier/prefix specs OWF	[[Imported from::...]] SMW			
			Prefix			Wiki page name prefixes MW		Secondary prefix for class and property page names	
		owl:Thing		:allProperties	Each gets own template and form for domainless properties OWF			Form and template for general properties from ...	Namespace (2ary level)
				:hideProperties					Top level form and template
		owl:Ontology							Ontology (2ary level)
		rdfs:isDefinedBy						Wiki page name namespace has levels	Source URI (2ary level)
		owl:imports							New sublevel
		rdfs:seeAlso			Loaded as part of ontology to process OWF				
		Fragment identifier			Link from page for property or category OWF				
	Text	rdfs:label		:label	Pagename MW , Label on form & template OWF			Labels for property and class on templates and forms	
		skos:prefLabel							
		rdfs:comment			Mouseover on label OWF, content on page OWF			Shows description	
		xml:lang			Selection of text display from user for label, comment, etc.			Adapts language to user	
		Delimiters	*Default Frensel*	Additional content	delimiter= SF				
			Cascading Fresnel		Put text: before, after, between, starting, ending, if empty OWF				
	CSS			Psuedo-classes	CSS MW	Link style MW			
				:containerStyle		For template and form tabel SF	Whole table		
				:resourceStyle			Row	Multiple	
				:propertyStyle				Single	
				:labelStyle			Cell	Left	
				:valueStyle				Right , class= SF for forms	
rdfs:	rdf:	:type			[[Category:]] MW			Wiki page is member of given category	
		Containers		:member in :showProperties	#arraymap/list SF			User can enter sorted list as comma-delimited	
		:Property			Page	Property: SMW		Recognized in wiki data system as	Property
		:Class		:Lens		Category:... MW , *also:* [[Category:...]] in template MW			Category
						Template:... MW (if domain)			Data display table/template
						Form:... SF (if domain)			Form for data entry
				:classLensDomain	[[Has default form::...]] on category SF			Clicking to new pages leads to given form for data entry	
					Check box to assign classes if not domain OWF				
			owl:oneOf		[[Category:...]] on individuals MW				
	:sub-	:subClassOf			[[Category:...]] on category page MW			Wiki and queries recognize as subcategory	
					Nested form access OWF				
		:subPropertyOf			[[Subproperty of::...]] on property page SMW			Queries using property also recognize superproperties	

						Grouped together on forms and templates by default OWF			
:domain			:showProperties					Placed	
		Fresnel not automatically generated from ontology	:hideProperties	Un-	Asssign property in template OWF		Properties in template and forms are	Removed	
			:showProperties rdf:List		Sort properties in template OWF			Sorted	
			:mergeProperties :alternateProperties		Template row queries multiple properties for one display SF		Values from muliiple properties display as if from one		
:range value is		Class			autocomplete on category=... SF		Pull down list shows current pages in target class(es)		
					[[Has default form::...]] on property SF		Clicking to new pages leads to given form for data entry		
		Default w/o :range	owl:ObjectProperty	[[Has type::...]] SMW	Page		Entered as wiki page name Links to wiki page		
			owl:DataProperty		String		Entered and displayed as unlinked string		
			owl:AnnotationProperty			[[Allows value::...]] SMW	User selects from fixed values		
		Literal			Number		Entered and displayed as number		
		xsd: textual[1]	owf:datatype		Date		Form for entering date components and displayed as date		
		xsd: :language, date parts [2]			Boolean		Select from two values		
		xsd: numeric[3]			URL	shows as: Linked URL text MW			
		xsd: :gYear				Image MW	Image itself appears directly		
		xsd: :date(Time)			String	Unlinked URL text SMW			
		xsd: :time							
		xsd: :Boolean	:Boolean						
		xsd: :URI	:uri						
		Fresnel not automatically generated from ontology	:value :image						
			:externalLink						
		owl:oneOf	owf:oneOf		[[Allows value::...]] SMW		User selects from fixed values		
			owf:datatype		Geographic coordinate, Code, Temperature		Proper entry and display for given data type		
owl:	RDFS-Plus		:inverseOf		Inverse property query SMW		Template also shows incoming links	In separate inverse property	
		Chains	:SymmetricProperty					With same property	
		Chains	:TransitiveProperty		Chained query or assignment SMW		Template shows all pages in chain with property		
		Equiv	:sameAs		All data and content put on unified page on wiki OWF				
		Equiv	:EquivalentClass						
		Equiv	:EquivalentProperty						
		Key	:FunctionalProperty		Data entry prevents duplication of keys OWF				
		Key	:InverseFunctionalProperty						
		Key	:hasKey						
	Values	:hasValue			#forminput: SF	current user as property value	Form for new member of class has prefilled values	Helps user register self	
					restricted SF		User cannot change value		
		:DataRange w/ :oneOf			[[Allows value::...]] SMW		User selects from fixed values		
		:allValuesFrom			autocomplete on category=... SF		Pull down list shows current pages in target class(es)		
		:someValuesFrom			*#arraymap/list* SF	*repeated fields* SF	*mandatory* SF		
					by default	by default *not*	ü	User must enter something	
	Cardinality	min	= 1		û			User must enter exactly one	
		max						User cannot enter more than one	
		max	> 1			by default *not*		User can enter more than one	
		min			ü	ü		User must enter at least given number	
		Default				û		User can enter any number without restrictions	
	Set	In range	Single class		autocomplete on category=... SF		Pulldown shows pages from	Class	
		In range	Union of classes					Any of the classes	
		In range	Multiple classes		autocomplete SF			Class intersection	
		Disjointness			Unique name assumption for wiki page names SMW				
Key	Status:	implemented	planned		in PHP	in Fresnel Forms	Planned	Priority	milestone Forms

Tools: MW MediaWiki, SMW Semantic MediaWiki, SF Semantic Forms, OWF OWF-only

[1] :string, :normalizeString, :XMLLiteral, :Name, :token, :NMTOKEN, :NCName, [2] :gYearMonth, :gMonthDay, :gDay, :gMonth

	[3] :double, :float, :int, :integer, :long, :short, :negativeInteger, :positiveInteger, :nonPositiveInteger, :nonNegativeInteger, :unsignedLong, :unsignedInt, :unsignedShort

Bijlage XV: Kennisdeling Wiki

Installatie Virtuele Server

De virtuele server van ABI - Team 30 is ingericht mbv het pakket virt-manager op een Ubuntu 14.04.1 64 bits platform. Het pakket kan geinstalleerd worden met het volgende commando

```
sudo apt-get install virt-manager
```

Er is hierbij gebruik gemaakt van een standaard Ubuntu 14.04.1 64 bits ISO image om de virtuele server in te richten (te installeren), met standaard opties.

Op het systeem zijn de volgende gebruikers geïnitialiseerd:

- alex
- joop
- teun

Naast het instellen van de hoofdgebruiker tijdens het installeren, kunnen de overige gebruikers als volgt ingesteld worden:

```
sudo adduser <username> -p <password> -G sudo
```

De volgende services zijn vooralsnog geïnstalleerd:

- sshd (hieronder beschreven)
- Fuseki
- MediaWiki

[edit]sshd

De service sshd wordt gebruikt voor het verbinden op basis van Secure Shell of Secure Copy. Mogelijke applicaties op Windows hiervoor zijn

- Putty
- WinSCP

Het pakket kan geinstalleerd worden door het volgende commando uit te voeren:

```
sudo apt-get install sshd
```

[edit]MediaWiki

zie MediaWiki Installatie

MediaWiki Installatie

Deze pagina beschrijft de installatie van MediaWiki op een server geinstalleerd zoals beschreven in Installatie Virtuele Server.

MediaWiki is als volgt geinstalleerd:

```
sudo apt-get install mediawiki
```

Vervolgens zijn de volgende acties nog uitgevoerd:

- bij de derde regel van /etc/mediawiki/apache.conf is het hekje vooraan de regel weggehaald
- bij /etc/apache2/ports.conf is de regel Listen 80 gewijzigd in Listen 3080
- mediawiki is als volgt toegevoegd aan de apache webserver:

```
sudo a2enconf mediawiki
```

- de apache server is als volgt opnieuw gestart:

```
sudo service apache2 restart
```

Semantic MediaWiki Extension

Ga naar https://www.semantic-mediawiki.org/wiki/Help:Installation , Installation, MediaWiki 1.19.x to 1.21.x . Volg de aanwijzingen, geen problemen tegengekomen. Om de extensie ook op de versiepagina van deze wiki te vinden kopieëer de require_once "$IP/extensions/ExtensionInstaller/ExtensionInstaller.php"; regel ook in de etc/mediawiki map.

Semantic Forms Extension

Ga naar: https://www.mediawiki.org/wiki/Extension:Semantic_Forms/Download_and_installation en volg de aanwijzingen.

Wil je deze extensie ook op versiepagina van deze wiki zien, kopieëer include_once "$IP/extensions/SemanticForms/SemanticForms.php"; weer in de etc/mediawiki map.

Ontwikkelomgeving

Deze pagina beschrijft de configuratie van de ontwikkelomgeving voor ontwikkeling van de Protégé Fresnel Forms plugin voor Protégé Desktop 5.

Voor de ontwikkelomgeving van MDDFresnel voor Protege 4, zie Ontwikkelomgeving Protege 4

[edit]Uitgangspunten

Als ontwikkelomgeving wordt uitgegaan van:

- Microsoft Windows 64 bits besturingssysteem
- Eclipse 4.4.1 (Luna)
- Java SE 7u71

[edit]Benodigdheden

- Eclipse IDE for Java EE Developers (Luna SR1)
- Java SE Development Kit 7u71
- TortoiseSVN 1.8.8
- Git for Windows 1.9.4
- TortoiseGit 1.8.11 (optioneel)
- Maven 3.2.3
- Protégé Desktop 5 Git Repository
- Protégé MDDFresnel ABI - Team 30 Repository

[edit]Installatie

Het installatieproces verloopt in de volgende stappen:

1. Installeren Paketten
    1. Installeren TortoiseSVN (indien nog niet geinstalleerd, Triviaal)
    2. Installeren Git for Windows (Triviaal)
    3. Installeren TortoiseGit (Triviaal en optioneel)
    4. Installeren Java SE Development Kit
        1. Uitvoeren installatie executable
        2. Omgevingsvariabele JAVA_HOME definiëren als C:\Program Files\Java\jdk1.7.0_71
    5. Installeren Maven
        1. Pak zip bestand uit in een nieuwe map, hierna <maven> genoemd
        2. Voeg <maven>\bin toe aan de omgevingsvariabele PATH
2. Bouwen Protégé 5

    1. Klaarzetten Protégé repository
    2. Bouwen Protégé
3. Bouwen Fresnel Forms
    1. Klaarzetten en bouwen Fresnel Forms working copy
    2. Bouwen Fresnel Forms
4. Configureren Eclipse
    1. Installeren en Configureren Eclipse
    2. Installeren Checkstyle
    3. Importeren Fresnel Forms project

Logging

Voor het loggen van interessante gegevens wordt in Fresnel Forms gebruik gemaakt van log4j. Dit, omdat Protégé er zelf ook gebruik van maakt en we derhalve gebruik kunnen maken van de configuratie en bijbehorende bestanden van Protégé.

[edit]Configuratie

Voor het configureren van welke logging getoond dient te worden in Protégé wordt het bestand log4j.xml aangepast, welke in dezelfde map staat als waar Protégé geïnstalleerd is (en waar bijvoorbeeld ook het bestand run.bat staat). Om bijvoorbeeld een TRACE-level logging van de FresnelForms plugin te krijgen kan het volgende gedaan worden:

1. Allereerst dient de regel

```
<param name="Threshold" value="INFO"/>
```

onder sectie `<appender name="console" class="org.apache.log4j.ConsoleAppender">` verwijderd te worden om andere logging levels toe te staan.

2. Vervolgens kan de volgende sectie toegevoegd worden ergens boven de `<root...` sectie:

```
<logger name="nl.ou.fresnelforms"><level value="trace"/></logger>
```

Wat ook handig kan zijn is het aanpassen van de ConversionPattern param behorende bij de layouts van appenders. Zo kan het volgende gebruikt worden om bijv. timestamps mee te nemen bij het loggen:

```
%d %-5p [%t] %c %x %m%n
```

[edit]Voorbeeld

Als compleet voorbeeld van een log4j.xml kan het volgende gebruikt worden:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">

<log4j:configuration xmlns:log4j='http://jakarta.apache.org/log4j/'>

  <appender name="console" class="org.apache.log4j.ConsoleAppender">
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d %-5p [%t] %c %x %m%n"/>
```

```
    </layout>
  </appender>

  <appender name="file"
            class="org.protege.common.log.ProtegeRotatingAppender">
    <param name="rotationCount"   value="7"/>
    <param name="directory" value="logs"/>
    <param name="relativeFile" value="protege-%u.log"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern"
             value="%d %-5p [%t] %c %x %m%n"/>
    </layout>
  </appender>

  <category name="org.mindswap.pellet">
    <priority value="error"/>
  </category>

  <!-- fear not, there is not too much here and it will be
       useful for a bit -->
  <category name="org.protege.editor.core.ProtegeApplication">
    <priority value="debug"/>
  </category>

  <logger name="nl.ou.fresnelforms">
    <level value="trace"/>
  </logger>

    <!-- Debugging session specific settings -->
    <!-- debug the parsing of the axioms from the owl api triple store
model
    <category name="org.protege.owl.rdf.impl.OwlTripleStoreImpl">
        <priority value="debug"/>
    </category>
    -->

         <!-- To debug automatic updates and commits from client
         <category
name="org.protege.owl.server.connection.servlet.ServletClientConnection">
               <priority value="debug"/>
```

```
          </category>

          <category
name="org.protege.editor.owl.client.ClientOntologyBuilder">
                        <priority value="debug"/>
          </category>
         -->
    <!-- End of Debugging session specific settings -->
  <root>
    <priority value ="info" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
  </root>

</log4j:configuration>
```

Opleveringen

Het maken van een oplevering van FresnelForms bestaat uit de volgende stappen:

Contents

[hide]

1 Het opwaarderen van het versienummer
2 Het bouwen van Fresnel Forms
3 Het testen van Fresnel Forms
4 Het committen van wijzigingen
5 Het maken van een tag
6 Het opwaarderen van het nieuwe SNAPSHOT versienummer en opnieuw committen
7 Het verspreiden via GitHub

[edit]Het opwaarderen van het versienummer

Om het versienummer voor een oplevering te wijzigen hoef je vaak slechts de -SNAPSHOT suffix te verwijderen, dus 1.0.5-SNAPSHOT wordt dan 1.0.5. Dit kun je in eclipse of rechtstreeks in het bestand pom.xml aanpassen.

- In eclipse kun je dit doen in het veld Version onder sectie Artifact.
- In het bestand pom.xml kun je het element version van element project aanpassen (in bijv. een tekst editor).

[edit]Het bouwen van Fresnel Forms

Voer een 'mvn clean install' uit op de command-line.

[edit]Het testen van Fresnel Forms

Door de gebouwde plugin vanuit de target map te kopiëren, kun je 'm onder Protégé testen om te zien of er regressie is opgetreden. Ook het versienummer kun je nu controleren door in Protégé het About venster te bekijken (menu: Help -> About), hier worden alle plugins inclusief versienummer getoond.

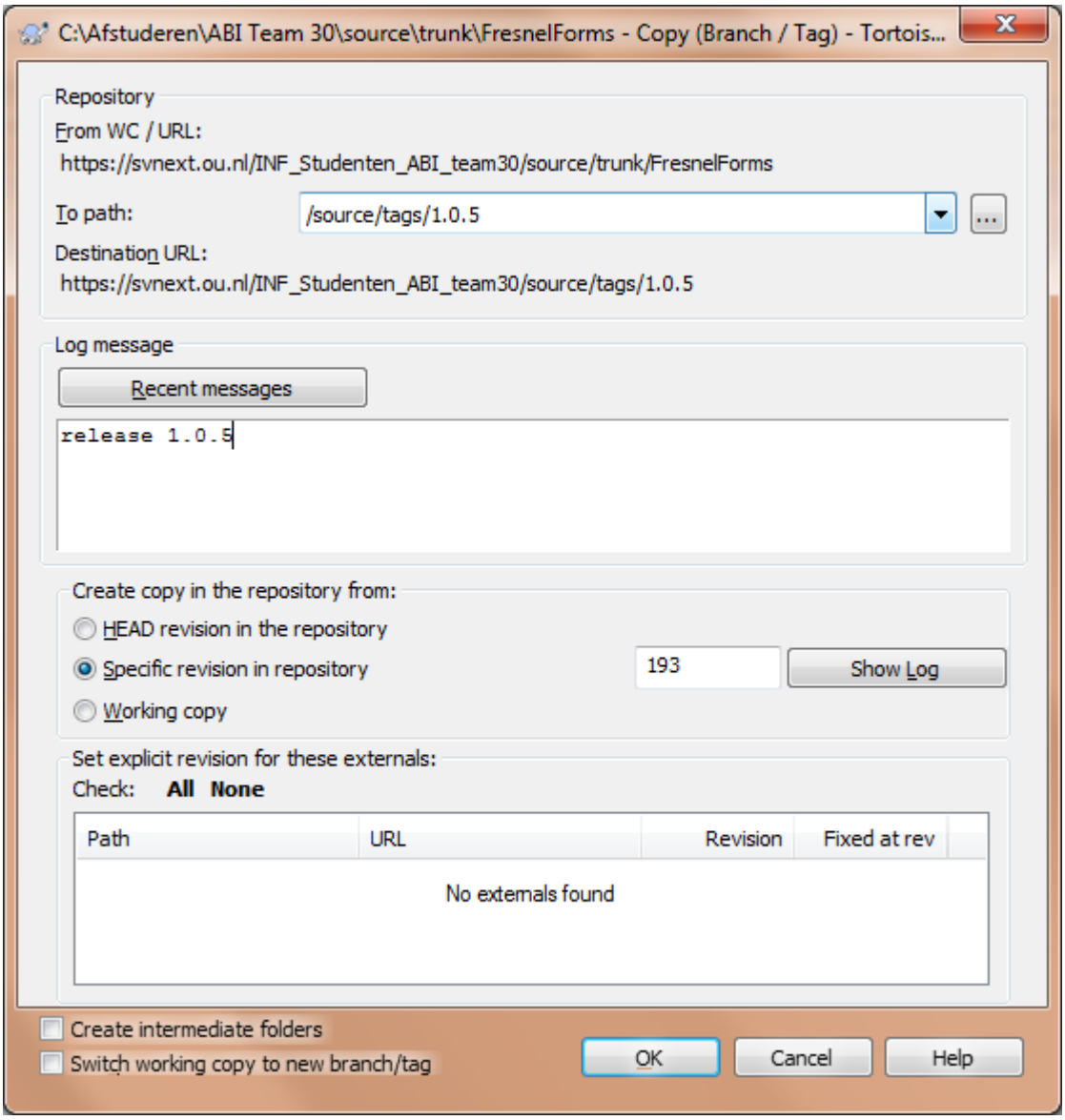
[edit]Het committen van wijzigingen

De wijzigingen, in het geval van een oplevering alleen het versienummer in het pom.xml bestand, dien je te committen met een toepasselijk commentaar, bijv. 'version updated for release 1.0.5'.

[edit]Het maken van een tag

Om later de oplevering eenvoudig te kunnen reproduceren, maken we een tag aan. Dit gaat als volgt:

1. Kies in het verkennervenster in de FresnelForms map in het contextmenu achter de rechtermuisknop voor TortoiseSVN -> Branch/Tag...
2. Het volgende venster verschijnt dan, vul dit als volgt in met het juiste versienummer en het juiste revisienummer van trunk.



[edit]Het opwaarderen van het nieuwe SNAPSHOT versienummer en opnieuw committen

Om de trunk klaar te zetten voor verdere ontwikkeling, waarderen we het versienummer op door een hoger nummer te kiezen en -SNAPSHOT als suffix toe te voegen aan pom.xml. Daarna moet ook dit weer gecommit worden aan versiebeheer met bijvoorbeeld een melding als 'preparations for version 1.0.6 (updated to 1.0.6-SNAPSHOT)'

[edit]Het verspreiden via GitHub

De gebouwde plugin kan via GitHub als volgt verspreid worden:

1. Ga naar https://github.com/ABI-Team-30/Fresnel-Forms/releases/new
2. Vul bij 'Tag version' het versienummer van de oplevering in (bijv. 1.0.5)
3. Geef bij 'Release title' een passende titel, bijv. 'Fresnel Forms 1.0.5 (bugfix release)'
4. Geef bij 'Describe this release' ten minste aan welke issues zijn opgelost, bijv. 'In this release, #1 and #8 are solved.'
5. Upload bij 'Attach binaries by dropping them here or selecting them.' het nl.ou.cs.is.protege.plugin.fresnelforms.jar bestand uit de target map.
6. Klik op 'Publish Release'

Tussentijds plugin bouwen

Dit kan vanuit Eclipse óf vanaf de command line met Maven. Het bouwen met maven bestaat uit het uitvoeren van het commando 'mvn clean install' vanaf de command line in de map waar de pom.xml staat (dus bv. trunk/FresnelForms). Het bouwen met eclipse bestaat uit het met de rechtermuisknop uitvoeren van 'Run As -> Maven install' voor het fresnelforms project.

In beide gevallen vind je de gebouwde plugin (nl.ou.cs.is.protege.plugin.fresnelforms.jar) in de target map. Deze kun je kopiëren naar de Protégé plugins map en voilá.

Bijlage XVI: Onderzoekscontext Alex

Inleiding

De paper submission**Invalid source specified.** beschrijft een systeem waarbij Fresnel gebruikt wordt als presentatiedefinitie en een Semantic Wiki gebruikt wordt als platform om gegeven ontologieen weer te geven en eenvoudig te vullen met individuals.

In dit document wordt onderzocht welke invloed Fresnel Forms heeft op de context waarin het onderzoek plaatsvindt. Er wordt hierbij specifiek ingezoomd op de brug die het slaat tussen enerzijds het Semantic Web en anderzijds Semantic MediaWiki. Hierbij wordt de equivalentie tussen het Semantic Web en Semantic MediaWiki onder de loep genomen en gekeken hoe de definitie van de user interface presentatie binnen de MDD aanpak past.

Dit document wordt bovendien gebruikt als basis voor de volgende consultvragen bij de onderzoeker Lloyd Rutledge:

1. (inleidende vraag):
   Gegeven het feit dat het Semantic Web en Semantic MediaWiki geen volledig equivalente semantische informatiesystemen zijn, constateren we dat gegevens die middels een door Fresnel Forms gegenereerde user interface voor Semantic MediaWiki ingevoerd zijn, niet in alle gevallen RDF tripels opleveren die aan de uitspraken in de bron-ontologie voldoen. Vindt u dit een beperking voor de gekozen oplossing?

   Zouden, in aansluiting op de vorige vraag, de owf:list en owf:mandatory properties niet beter direct uit de bronontologie herleid kunnen worden (uit owl:min- en owl-maxCardinality) in plaats van aanpasbaar te zijn in Fresnel Forms en ruimte te geven voor discrepanties met de bronontologie?
2. *Het lijkt haalbaar, doch niet triviaal, dat middels een samengestelde extensie voor Semantic MediaWiki een betere aansluiting gevonden wordt van Semantic MediaWiki met de datatypes zoals die in het Semantic Web gebruikt worden. Is het volgens u de moeite waard om dit verder te onderzoeken in een volgend ABI project?*
3. *Fresnel is slechts geschikt als presentatiedefinitie en niet als invoerdefinitie; de presentatie van een formulier in Semantic Forms kan, afgezien van de volgorde van velden, niet middels Fresnel gedefinieerd worden. De benodigde extra*

invoerproperties zijn nu als kleine uitbreidingen op de mogelijkheden van een Fresnel Format gedefinieerd, maar vindt u ook dat dit een meer prominente plaats verdient in een platformonafhankelijke user interface ontologie bijvoorbeeld op basis van W3C normen als XForms? Zo ja, komt ook dit dan in aanmerking om verder te onderzoeken in een volgend ABI project?

In de volgende hoofdstukken van dit document zijn de aanleidingen voor deze vragen verder uitgewerkt.

Equivalentie tussen Semantic Web en Semantic MediaWiki

Zoals in de paper submission beschreven wordt, ondersteunt de semantic Wiki het creeren van systemen met enige equivalente Semantic Web functionaliteiten zoals data annotations en queries.

Globaal kent Semantic MediaWiki, inclusief de extensie Semantic Forms de volgende Semantic Web equivalente functionaliteit:

Semantic Web	**MediaWiki**	**Semantic MediaWiki**	**Semantic Forms**
rdfs:Class	Category:... page waar individuals (instanties) van de klasse lid van zijn. Individuals (pages) hebben een [[Category:...]] verwijzing.	Pages welke een bijbehorende {{template:…}} gebruiken om de individuals en hun properties weer te geven	Form:... page welke het invullen en wijzigen van een instantie van een klasse ondersteunt. [[Has default form::...]] op Category:... page
rdfs:subClassOf	Category:... Page met de speciale property [[Category:...]]		
rdf:Property		Property:... Page welke als ‘categorie’ geldt voor propertywaarden	
rdfs:subPropertyOf		Property:... Page met de speciale property [[Subproperty of::...]]	
owl:ObjectProperty		Property:... Page met de speciale property [[Has type::Page]]	
owl:DataProperty		Property:... Page met een [[Has type:...]] anders dan Page.	

Tabel 9: Equivalentie tussen Semantic Web en Semantic MediaWiki

Het typesysteem voor dataproperties is in Semantic MediaWiki echter anders dan dat van het Semantic Web. Zoals in de figuur hierboven kan worden gezien, is er in Semantic MediaWiki één speciale Property ([[Has type::…]]) welke definieert of iets een Objectproperty is of een DataProperty. Als het een DataProperty betreft, definieert het ook van welk type de DataProperty is. Zoals we zullen zien is dit type niet volledig equivalent met de XML Schema types zoals ze in het Semantic Web gelden.

Semantic MediaWiki Property types

Voor Semantic MediaWiki Property pages, het equivalent van rdf:Property binnen het Semantic Web, wordt het type bepaald door de Speciale Property [[Has Type:...]]. De volgende types zijn binnen Semantic MediaWiki gedefinieerd:

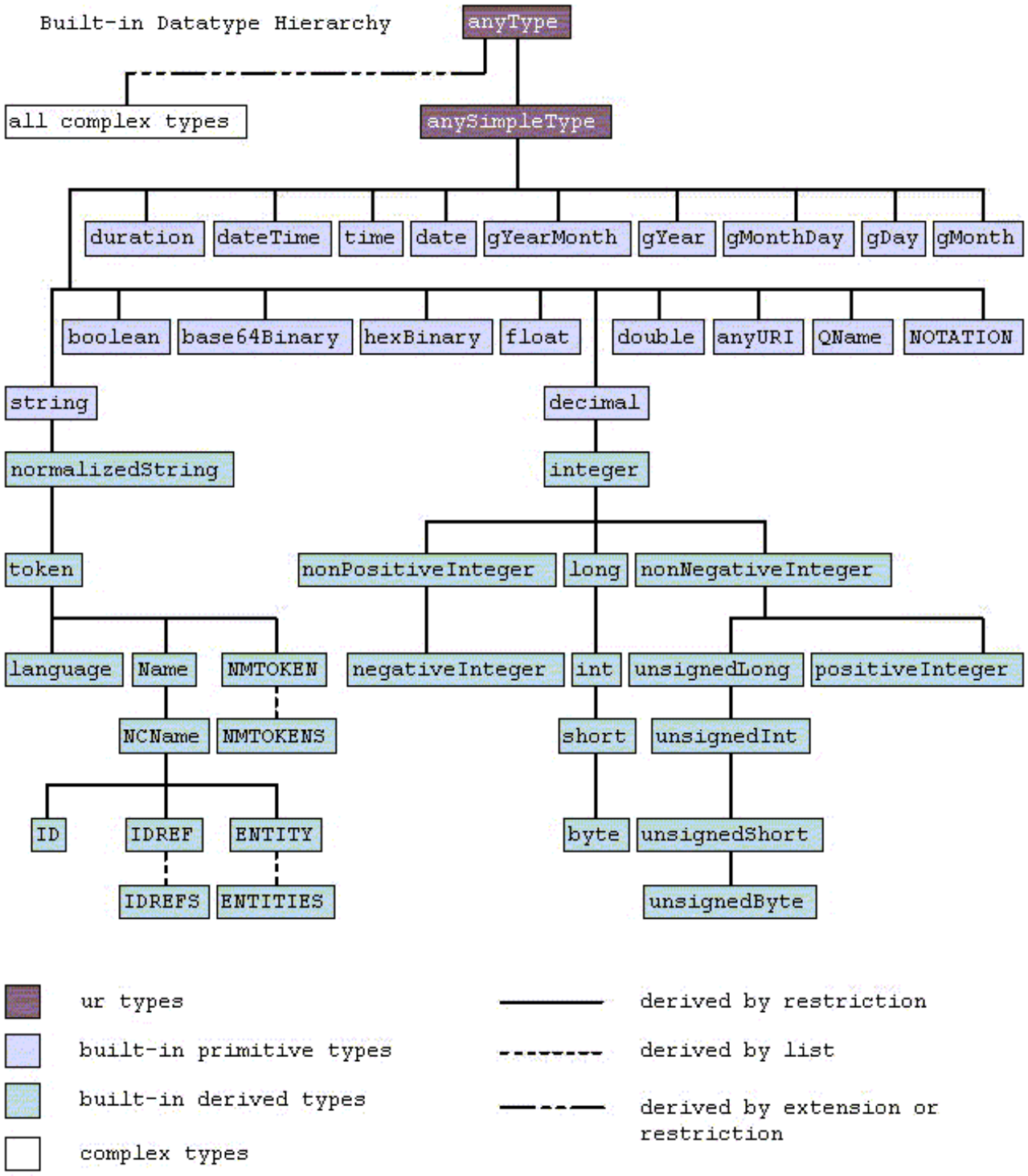
Type	**Omschrijving**
Annotation URI	URI (klein technisch verschil met URL)
Boolean	booleaanse waarde (waar/onwaar)
Code	voorgeformatteerde text
Date	tijdstempel (altijd met datum component, tijdscomponent optioneel)
Email	e-mailadres (klikbare mailto link)
Geographic coordinates	geografische coördinaten (Semantic Maps)
Number	numerieke waarde
Page	pagina (invulbaar als link op een template)
Quantity	waarde met een eenheid
Record	Samengesteld type van waarden met vaste volgorde en typen.
Telephone number	telefoon nummer (RFC 3966)
Temperature	temperatuur (soortgelijk Quantity)
Text (of String)	vrije text
URL	IRI, URI of URL

Tabel 10: Property types in Semantic MediaWiki

Deze types zijn binnen Semantic MediaWiki hard gecodeerd en niet uit te breiden.

Semantic Web Datatypes

In het Semantic Web wordt het gebruik van de ingebouwde RDF compatible XML Schema types aangeraden**Invalid source specified.**. Voor dit document doen we daarom de aanname dat de datatypes, zoals aangeleverd door de bronontologie, simple XML Schema datatypes zijn. De standaard XML Schema datatypes zijn volgens **Invalid source specified.** als volgt gestructureerd:



Figuur 40: Built-in datatypes XML Schema

Vertaling van types

Wanneer beide semantische systemen equivalent zouden zijn, zou er een bijectieve relatie zijn tussen datatypes in het Semantic Web en Property types in Semantic MediaWiki.

Omdat het aantal datatypes van beide semantische systemen verschillend is, is dit echter niet het geval en dient er een vertaling gemaakt te worden van datatypes met name literal waarden van de verschillende semantische systemen aan elkaar te kunnen relateren

Vertaling van Semantic Web datatypes naar Semantic MediaWiki Property types

Een vertaling van Semantic Web datatypes naar Semantic MediaWiki property types kan op de volgende manier gemaakt worden:

owl:DataProperty **rdfs:Datatype**	**Semantic MediaWiki type** (+ bijzonderheden)
	Text (of String)
rdfs:Literal	(arbitraire literal)
rdf:XMLLiteral	(geldige XML content)
xs:string	(arbitraire string)
xs:normalizedString	(vervang whitespace)
xs:token	(vervang whitespace en voeg samen)
xs:language	(RFC 1766)
xs:NMTOKEN	(token welke aan de XML NMTOKEN productieregels voldoet)
xs:Name	(NMTOKEN welke begint met een letter, '_' of ':')
xs:NCName	(Name welke geen ':' bevat)
xs:duration	(iso8601 duration)
xs:gMonthDay	(dag van een maand)
xs:time	(tijd zonder datum)
xs:boolean	**Boolean**
	Number
xs:double	(64-bit double precision floating point)
xs:float	(32-bit single precision floating point)
xs:decimal	(subset of real numbers)
xs:integer	(gehele getallen)
xs:nonPositiveInteger	(niet-positieve gehele getallen)
xs:negativeInteger	(negatieve gehele getallen)
xs:long	(gehele getallen van -2^63 tot 2^63)
xs:int	(gehele getallen van -2^31 tot 2^31)
xs:short	(gehele getallen van -2^15 tot 2^15)
xs:byte	(gehele getallen van -2^7 tot 2^7)
xs:nonNegativeInteger	(positieve gehele getallen én nul)
xs:unsignedLong	(gehele getallen van 0 tot 2^64)
xs:unsignedInt	(gehele getallen van 0 tot 2^32)
xs:unsignedShort	(gehele getallen van 0 tot 2^16)
xs:unsignedByte	(gehele getallen van 0 tot 2^8)
xs:positiveInteger	(positieve gehele getallen)
xs:gMonth	(nummer van de maand)
	Date
xs:dateTime	(CCYY-MM-DD'T'hh:mm:ss('.'s+))
xs:date	(CCYY-MM-DD)
xs:gYear	(CCYY)
xs:gYearMonth	(YYCC-MM)
xs:anyURI	**URL**

Tabel 11: Vertaling van Semantic Web datatypes naar Semantic MediaWiki Property types

Vertaling van Semantic MediaWiki Property types naar Semantic Web datatypes

Een zuivere vertaling van Semantic MediaWiki Property type naar Semantic Web datatypes is niet te maken omdat er meer Semantic Web Datatypes gedefinieerd zijn dan Semantic MediaWiki Property types, zelfs met de aanname dat hier slechts simple XML Schema datatypes gebruikt worden.

Om een, in Semantic MediaWiki (of Semantic Forms) aangemaakte waarde te kunnen laten voldoen aan het XML Schema datatype zoals dat in het rdfs:Datatype in de ontologie is gedefinieerd, moet in ieder geval de invulling ervan beperkt worden tot de mogelijkheden van de lexicale ruimte van de betreffende rdfs:Datatype. Als voorbeelden:

- Een Text Property in Semantic MediaWiki is slechts te vertalen naar een xs:token wanneer alle whitespace karakters erin vervangen zijn door de XML gecodeerde waarde (%20) én zijn samengevoegd.
- Een Number Property in Semantic MediaWiki is slechts te vertalen naar een xs:nonNegativeInteger wanneer het niet negatief is.

Deze vertaalslag is belangrijk om vanuit Semantic MediaWiki RDF te kunnen exporteren welke past bij de ontologie waar het systeem voor ontworpen is.

Op dit moment vindt in de Semantic MediaWiki RDFexport functie de volgende vertaling plaats:

Semantic MediaWiki Property type	**Semantic Web Datatype**
[[Has type::Number]]	xs:double
[[Has type::Date]]	Afhankelijk van de aanwezige datum/tijd componenten wordt het: xs:gYear (bij alleen een jaartal) xs:gYearMonth (bij jaartal en maand) xs:date (bij een datum zonder tijd) xs:datetime (bij een datum en tijd)
[[Has type::Boolean]]	xs:Boolean
[[Has type::Geographic coordinates]]	wordt niet ondersteund
...alle andere typen...	xs:string

Tabel 12: Vertaling van Semantic MediaWiki Property types naar Semantic Web datatypes

Om er bij het creëren en bewerken van semantische gegevens in Semantic Forms voor te zorgen dat de invoer beperkt wordt tot de lexicale ruimte van het betreffende datatype, zullen de invoertypen hiervoor ook ondersteuning moeten bieden.

In Semantic Forms kan middels [[Allows value::...]] de waarden voor een Property beperkt worden tot een vooraf gedefinieerde set van waarden. Voor (bijna) oneindig aftelbare datatypes als xs:unsignedInt is dit echter niet voldoende aangezien de lijst van waarden dan ook (bijna) oneindig zou zijn (xs:unsignedInt bevat nul plus alle gehele positieve getallen tot en met 4294967295).

De Semantic Forms extensie Semantic Forms Inputs **Invalid source specified.** zou kunnen ondersteunen in het beperken van de invoer met bijv. de extra gedefinieerde invoertypen Time picker (timepicker field type) en Regular expression filter (regexp field type). Voor

numerieke types is echter nog geen extensie aanwezig om waarden te beperken op basis van het bereik van het type.

Om Semantic MediaWiki beter equivalent te maken met het Semantic Web zal een extensie gemaakt moeten worden welke de invoer van waarden van beperkte (XML Schema) typen mogelijk maakt en afdwingt of valideert. Daarnaast dient de resulterende waarde ook correct geexporteerd te worden naar RDF.

Afhankelijk van het doel is het in Semantic MediaWiki mogelijk om extensies te bouwen of gebruiken voor:

1. Het uitbreiden van de wiki markup voor het definieren van Articles. Dit type extensie valt onder de categorie Parser extensions [47] waarmee uitgebreide mogelijkheden geboden worden om de wiki syntax aan te passen naar behoefte.
2. categorie Special page extensions[48]. OWL Wiki Forms valt onder deze categorie.
3. Het veranderen van het uiterlijk van MediaWiki. Dit type extensie valt onder de categorie User interface extensions[49].
4. Het verbeteren van de veiligheid via eigen authenticatiemechanismen. Dit type extensie valt onder de categorie Authentication and authorization extensions[50].

In het geval van het creëren van een Semantic MediaWiki extensie voor het toevoegen van ondersteuning voor XML schema datatypes is minstens een extensie nodig die de functies van categorie 1, 2 & 3 bevat.

- Een parser extension zal ondersteuning kunnen bieden voor de typedefinitie op de Property pages.
- Een User interface extensionbij zal ondersteuning kunnen bieden bij de invoer en beperking of validatie in een Semantic Form.
- Een Special page extension kan dan de betreffende datatypen één op één exporteren opdat de geëxporteerde RDF volledig voldoet aan de definitie van de bronontologie.

Definitie van de User Interface

CRUD

Als we kijken naar de vier basisoperaties die volgens de MDD aanpak **Invalid source specified.** op duurzame gegevens kunnen worden uitgevoerd, namelijk C(reate), R(ead), U(pdate) en D(elete), dan is het semantische platform , hier Semantic MediaWiki plus extensies, op te delen in de volgende functioneel gescheiden delen:

- De (Semantic MediaWiki) info(rm)box wordt als doel gesteld om, middels vulling van een template, de geselecteerde statements voor de subjecten uit de bronontologie weer te geven, de R(ead) operatie dus.
- Een Semantic Form wordt gebruikt om de geselecteerde statements van een subject, van objecten te voorzien én bij te werken, de C(reate) én U(pdate) operaties dus.

[47] http://www.mediawiki.org/wiki/Category:Parser_extensions

[48] http://www.mediawiki.org/wiki/Category:Special_page_extensions

[49] http://www.mediawiki.org/wiki/Category:User_interface_extensions

[50] http://www.mediawiki.org/wiki/Category:User_access_extensions

- De standaard manier van MediaWiki om een Page te verwijderen kan worden gebruikt om Pages, Categories & Properties te verwijderen, de D(elete) operatie dus.

Voor de R(ead) operatie in Semantic MediaWiki is in de paper submission aangetoond dat deze te definiëren is m.b.v. de Fresnel Forms plugin onder Protégé. Aan de D(elete) operatie wordt in Fresnel Forms geen aandacht besteedt en dit kan eenvoudig (doch arbeidsintensief bij veel Pages) in MediaWiki zelf vorm gegeven worden.
Voor de C(reate) en U(pdate) operaties, welke ingevuld worden door Semantic Forms, kan echter de ondersteuning in Fresnel Forms nog verbeterd worden.

Semantic Forms Input types

Een Semantic Form **Invalid source specified.** is opgebouwd uit velden welke standaard een bepaalde manier van invoer hebben, afhankelijk van het datatype van de property of het feit dat het een objectproperty is (een property met type Page).

Daarnaast heeft elk veld voor een property met een bepaald datatype, één of meer alternatieve manieren van invoer welke in de Semantic Forms broncode ingesteld kunnen worden. Zo krijgen properties met datatype Page (objectproperties dus) standaard het invoertype ‘Text with autocompletion’, maar er kan ook gekozen worden voor combobox, dropdown, tree etc.

Op dit moment zijn de mogelijke invoertypen voor velden met een enkele waarde:

Data type	Default input type	Default size	Other allowed input types
URL	text	100	textarea
Text, String (SMW >= 1.9)	textarea	5 x 30	text
String (SMW < 1.9)	text	35	text with autocomplete, combobox, textarea, textarea with autocomplete
Page	text with autocomplete	35	text, combobox, dropdown, textarea, textarea with autocomplete, tree
Number	text	10	textarea
Geographic coordinate	openlayers		googlemaps
Enumeration (any SMW property, or Cargo field, with defined "allowed values")	dropdown		radiobutton
Date	date		datetime (*includes hours, minutes, seconds and AM/PM indicator*), year (*year only*)
Code	textarea	5 x 30	text
Boolean	checkbox		dropdown, radiobutton

Tabel 13: Semantic Forms Input types voor velden met een enkele waarde

Voor velden met de mogelijkheid voor meerdere waarden geldt:

Data type	Default input type	Default size	Other allowed input types
Page	tokens	100	text, textarea, text with autocomplete, textarea with autocomplete, tree, checkboxes
String	text	100	textarea, textarea with autocomplete, text with autocomplete
Enumeration	checkboxes		listbox

Tabel 14: Semantic Forms Input types voor velden met de mogelijkheid voor meerdere waarden

Er zijn in Fresnel Forms geen mogelijkheden om keuzes te maken in welk invoertype gebruikt dient te worden voor een bepaald veld op een bepaald Form.

Fresnel

Voor het opslaan en inlezen van de user interface definitie wordt binnen de gekozen oplossing Fresnel **Invalid source specified.** gebruikt.

Fresnel is een eenvoudige, browser-onafhankelijke vocabulaire voor de manier waarop de informatie uit een RDF model weergeven moet worden. Dit betekent dat het grotendeels informatie bevat waarmee de R(read) operatie vorm kan worden gegeven. Met de Tim Berners-Lee infobox als doel heeft Fresnel inderdaad alles in huis om de huidige weergave vorm te geven. Ook de owf:delimiter property welke een scheidingsteken aanduidt, kan met Fresnel gedefinieerd worden door gebruik te maken van fresnel:contentBefore, fresnel:contentAfter, fresnel:contentFirst, fresnel:contentLast en fresnel:contentNoValue. Wanneer bijv. fresnel:contentBefore “,” geldt en fresnel:contentFirst “”, dan is effectief owf:delimiter “,” gedefinieerd.

Er zijn in Fresnel geen mogelijkheden aanwezig om aan te geven wat voor invoertype gebruikt moet worden om een waarde aan te maken of te wijzigen (voor de C(reate) en U(pdate) operaties dus) en wat voor regels ervoor gelden (bijv. autocompletion of niet, restrictie op waarden, meerdere waarden, het verplicht invullen ervan en meer generieke validatie van ingevoerde gegevens).

In Fresnel Forms hebben we voor een paar van deze mogelijkheden uitbreidingen gemaakt in de presentatieontologie (owf:mandatory, owf:list, owf:autoCompleteFromClass en owf:datatype).

Wanneer echter de aanname wordt gedaan dat de gegenereerde interface geldige data moet kunnen leveren voor de bronontologie, dan zijn deze extra properties rechtstreeks af te leiden uit de bronontologie. Wanneer bijvoorbeeld owl:minCardinality voor een property gelijk is aan 1, dan is het resulterende invoerveld verplicht om te voldoen aan de definitie van de ontologie.

Wat daarnaast nog opmerkelijk is, is dat de styling properties (van type fresnel:stylingInstructions) voor het grootste gedeelte geen invloed hebben op het uiterlijk van de invoertypen. Semantisch is er dus een verschil tussen de invloed die de Fresnel definitie heeft op enerzijds de R(ead) operatie en anderzijds de C(reate) en U(pdate) operaties.

In plaats van het uitbreiden van de (owf) properties van een Format kan ook gedacht worden aan een apart type, naast Lenses, Format en Groups. Hierin kunnen dan de eigenschappen

voor een gewenst invoertype gedefinieerd worden (bijv. owf:Control). Een Fresnel Lens kan dan nog steeds gebruikt worden om de Properties voor een Form te selecteren en de bijbehorende owf:Control definitie kan dan op een soortgelijke manier cascaden als een Fresnel Format. Om de user interface enerzijds platform onafhankelijk te houden en anderzijds standards compliant, zou hierbij goed gekeken kunnen worden naar hoe andere User Interface talen gedefinieerd zijn. Mogelijke candidaten zijn:

- XIML[51]
- UIML[52]
- UsiXML[53]
- XForms[54]

Met name XForms lijkt interessant aangezien deze de Controls abstract definieert en de mogelijkheid geeft om een subtype te definieren (bijv. compact, full of minimal). Zo kan een platformonafhankelijke opzet worden gemaakt van de User Interface voor de C(reate) en U(pdate) operaties.

In eerste instantie zou XForms slechts als richtlijn gebruikt kunnen worden om de User Interface platformonafhankelijk te definieren, terwijl het dan op een later moment ook eenvoudig gebruikt zou kunnen worden als platform om de Annotation and Browse interface op te baseren.

Hoewel XForms zelf op dit moment geen ondersteuning heeft voor autocompletion, kan dit wellicht in XForms 2.0 middels Asynchronous Javascript Function Calls geboden worden. In Semantic Forms worden hiervoor overigens de javascript library select2[55] en de jQuery UI Autocomplete Widget[56] voor gebruikt.

Bibliography

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

[51] http://www.ximl.org/

[52] http://docs.oasis-open.org/uiml/v4.0/cd01/uiml-4.0-cd01.html

[53] http://www.usixml.org/en/home.html?IDC=221

[54] http://www.w3.org/TR/xforms/

[55] https://select2.github.io/

[56] http://api.jqueryui.com/autocomplete/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage XVII: Onderzoekcontext MDD

ABI team 30
Joop van de Heijning

Open Universiteit

Onderzoekcontext MDD

Introductie

In dit verslag worden verbanden tussen de onderzoekcontext van dit project met Model Driven Development (MDD) besproken. Het doel is om meer inzicht te krijgen in de problemen en eventueel gevonden oplossingen die in de literatuur omtrent dit vakgebied te vinden zijn. Dit inzicht kan specifieke nieuwe requirements opleveren voor ons project, of dit inzicht kan zelfs laten zien welke nieuwe wegen ons project in de onderzoekcontext geopend heeft.

Gezien de aard van het project (op bachelorniveau) en de beperkte ruimte in dit verslag is er voor gekozen om de bestudering van de literatuur te beperken tot die onderwerpen die tot de bredere basis van MDD gerekend kunnen worden. Ook bleek na deze beperking het domein nog voldoende vruchtbaar te zijn voor onderzoek. Die onderwerpen die relevant leken zijn hieronder besproken.

In een consult zijn vragen aan de onderzoeker gesteld naar aanleiding van de bevindingen in hoofdstukken Gebruik maken van bestaande systemen, Modellen en standaarden, Annotaties, en Low ceiling. Diens antwoorden zijn aan het einde van deze hoofdstukken in dit verslag verwerkt.

Automatisering en grafische User Interface

Eén van de aantrekkelijkste aspecten van MDD is wellicht de automatisering van het schrijven van code. Het verhoogt de productiviteit van producenten van software en verbetert de betrouwbaarheid van het eindproduct (Selic, 2003).

De meeste verkopers van MDD tools hebben zich geconcentreerd op het automatisch produceren van code vanuit modellen (Atkinson & Kühne, 2003). Dit heeft wel tot gevolg dat er een drempel tot gebruik opgeworpen wordt: Programmeurs zijn gedwongen om een nieuwe taal te leren voor het specificeren van de modellen (Myers, Hudson, & Pausch, 2000). Gelukkig kunnen de meeste modellen gebouwd worden in een grafische omgeving (Vanderdonckt, 2008).

Grafisch modelleren sluit goed aan bij het menselijke visuele perceptievermogen en wordt dan ook gezien als één van de fundamenten van de ondersteuning voor MDD. In het bijzonder wordt het gebruik van object georiënteerd modelleren aanbevolen als technologie met de beste staat van dienst (Atkinson & Kühne, 2003).

Fresnel Forms[57], het eindproduct van dit afstudeerproject, bouwt voort op MDD Fresnel, een plug-in voor Protégé [58] (Brenninkmeijer & Zwanenberg, 2014). Deze tool produceert automatisch code (zowel platformonafhankelijk als –specifiek in de door ons uitgebreide versie) en maakt gebruik van een UML klassediagrammen-achtige Graphical User Interface voor het specificeren van het model. Dit sluit dus goed aan bij bovenstaande bevindingen uit de literatuur.

[57] http://is.cs.ou.nl/OWF/index.php5/Fresnel_Forms

[58] http://protege.stanford.edu/

Gebruik maken van bestaande systemen

Een verstandige en praktische manier om nieuwe technieken zoals MDD te introduceren is als extensie van een bestaand project. Dit verlaagt de risico's en maakt gebruik van al gemaakte investeringen (Selic, 2003). Fresnel Forms maakt handig gebruik van het beheren van een ontologie door Protégé. Wellicht dat deze succesvol gebleken aanpak nieuwe wegen kan openen voor onderzoek naar nieuwe tools die hierop voortbouwen?

Professor Rutledge merkt in het consult op dat deze aanpak onderzoekstechnisch niets nieuws oplevert, wel dat Frensel Forms in combinatie met Protégé een belangrijke aanwinst voor de praktijk is.

Modellen en standaarden

Er wordt verwacht dat User Interfaces (UI's) op steeds meer verschillende platformen zullen moeten draaien in de toekomst (Van den Berg, et al., 2010). Daarnaast is er de wens om software-artefacten te beschermen tegen veranderingen op platformniveau, om zo de levensverwachting van het artefact te verhogen. Vandaar dat ontwikkelaars zo veel als mogelijk het automatisch generen van code vanuit platformonafhankelijke modellen moeten nastreven (Atkinson & Kühne, 2003).

Een probleem van modellen is dat deze vaak sterk verbonden zijn met het model-gebaseerd systeem en niet kunnen worden geëxporteerd (Vanderdonckt, 2008). Daar tools altijd aan verandering onderhevig zijn, beperkt dit de levensverwachting van het software-artefact. Een technische requirement van een tool is dan ook dat modellen worden opgeslagen in formaten die andere tools ook kunnen gebruiken (Atkinson & Kühne, 2003).

Hier tekent de behoefte aan standaardisatie in formaten zich af. Standaardisatie heeft nog meer voordelen behalve het faciliteren van gebruik door verschillende tools, bijvoorbeeld het codificeren van best practices, aanmoedigen van hergebruik en specialisatie (Selic, 2003).

Het automatisch genereren van modellen gebeurt door mapping functions. Deze geven een verhoging van de productiviteit waarbij de verhoging door alleen het gebruiken van modellen in het software proces bij in het niet valt. Mapping functions zorgen er voor dat kennis ingezet bij het ontwikkelen van een software-artefact niet verloren gaat. Modellen gebruikt in een ontwikkelingsproces kunnen zich zo onafhankelijk ontwikkelen, wat bijdraagt aan een langere levensverwachting van de modellen.

Fresnel Forms maakt gebruik van een platformonafhankelijk model in een gestandaardizeerde vocabulary Fresnel (Pietriga, Bizer, Karger, & Lee, 2006), met een extensie genaamd OWF. Ook wordt verwacht dat het mogelijk zal zijn om een groot deel van de gebruikte mapping function van Fresnel naar Semantic MediaWiki her te gebruiken voor eventueel een mapping naar een ander platform.

Zal het onderzoek van Professor Rutledge in een vorig artikel **Invalid source specified.** en het artikel op basis van onder andere dit project bijdragen aan een verdere acceptatie van Fresnel als standaard vocabulary voor UI's van informatiesystemen gebaseerd op het Semantic Web, zoals Fresnel Forms produceert?

Na de introductie in 2006, zegt Professor Rutledge in het consult, werd Fresnel een korte tijd als veelbelovend gezien. Uiteindelijk werd het niet opgepakt, omdat mensen niet het Semantic Web gingen gebruiken zoals zij het World Wide Web gebruiken. Maar in feite

wordt er wel veel gebrowsed op het Semantic Web, wikipedia[59] is namelijk een Semantic Web browser. Met Fresnel Forms op Semantic Forms[60] probeert Professor Rutledge te laten zien dat het toevoegen van informatie en browsen op het Semantic Web efficiënter kan. Wellicht wordt Fresnel zo toch wel relevant.

Onvoorspelbaarheid

Een punt van zorg is de onvoorspelbaarheid die MDD met zich meebrengt. Het verband tussen specificatie en uiteindelijk resultaat is niet altijd duidelijk. Een goed voorbeeld van onvoorspelbaarheid in Fresnel Forms is het door ons geïmplementeerde heuristische sorteer-algoritme van Falco Paul (Paul, 2014). Één oplossing is om een expliciete documentatie over de werking van de tool aan te bieden, een andere het geven van de mogelijkheid tot voorvertoning van het resultaat van een toepassing van een regel (Vanderdonckt, 2008).

ABI Team 30 zal aanwijzingen aanbieden op de Fresnel Forms website voor het installeren van Maven Site projectdocumentatie[61]. Daarnaast zal de tool goed gedocumenteerd moeten worden in de scriptie.

Prototyping

Het geven van een mogelijkheid tot voorvertoning haakt in op het gegeven dat experimenteren één van de meest fundamentele manieren van leren is (Selic, 2003). Specifiek met betrekking tot het ontwerpen van UI's helpen tools met het automatisch produceren van code. Dit stelt de ontwikkelaar in staat om sneller prototype's te maken en meer iteraties te doorlopen, wat leidt tot UI's van hogere kwaliteit (Myers, Hudson, & Pausch, 2000). Ook Selic benadrukt het belang van een snelle doorlooptijd van iteraties voor productiviteit.

Voor het produceren van een prototype van een wiki UI met Fresnel Forms is het nu nog nodig om het geproduceerde platformspecifieke model handmatig te importeren op een MediaWiki server met special:import[62]. Het is te verwachten dat het versnellen van dit proces met behulp van een API endpoint[63] de productiviteit van eindgebruikers van Fresnel Forms zal verhogen.

Annotaties

Niet alle informatie gerelateerd aan UI objecten kunnen bevat worden in een specifieke tool die alle doeleinden dient. Daarom is er behoefte aan enige steun voor design gebaseerd op annotaties (Vanderdonckt, 2008). Fresnel Forms geeft deze mogelijkheid niet. Het is te overwegen om een optie in de GUI toe te voegen die eindgebruikers de mogelijkheid geeft

---

[59] http://www.wikipedia.org/

[60] https://www.mediawiki.org/wiki/Extension:Semantic_Forms

[61] https://maven.apache.org/plugins/maven-site-plugin/

[62] http://www.mediawiki.org/wiki/Manual:Importing_XML_dumps#Using_Special:Import

[63] http://www.mediawiki.org/wiki/API:Main_page

om commentaar toe te voegen aan lenzen of properties. Zouden deze kunnen worden doorgegeven aan het platformspecifieke model (Fresnel met OWF) in de form van een OWF formatting property?

In het consult geeft Professor Rutledge aan dat zulke annotatie properties al bestaan, RDFS heeft bijvoorbeeld label en comment, ook XML heeft comments. Er is waarschijnlijk geen behoefte aan een optie in de GUI, andere MDD tools hebben deze namelijk ook niet. Een ontwikkelaar kan, indien nodig, annotaties handmatig in de code toevoegen.

Low ceiling

Één van de moeilijkst op te lossen problemen van het toepassen van MDD is de kwaliteit van de resulterende UI's (low ceiling probleem). MDD systemen maken het ook moeilijk om een breed scala van oplossingen te verkennen (wide walls) (Vanderdonckt, 2008). Een optimale ontwikkelmethode zou zijn het zowel ondersteunen van systematische MDD, als meer handmatige informele methoden (Van den Berg, et al., 2010).

De mogelijkheden tot individualisatie van UI's die Fresnel Forms biedt zijn beperkt tot opties in de GUI en het invullen van CSS. Er bestaan extensies van MediaWiki die het toelaten om JavaScript code te laden, zoals de JavaScript extensie[64]. Zou het interessant zijn om te onderzoeken in hoeverre dit kan helpen om het Fresnel Forms platformspecifieke model verder te individualiseren?

Professor Rutledge geeft in het consult aan dat JavaScript eventueel net als CSS gestructureerd toegevoegd zou kunnen worden aan Fresnel. Maar in feite heeft hij zelf al een oplossing voor het probleem gepresenteerd in 2013. In een gegenereerde MediaWiki sjabloon wordt namelijk gecontroleerd op aanwezigheid van een "InformboxTop". Indien een wiki-ontwikkelaar deze al had gecreëerd, wordt deze bijgevoegd. **Invalid source specified.**.

Conclusie

Het is gebleken dat Fresnel Forms gebruikt maakt van de belangrijkste aspecten van MDD, zoals het gebruik maken van automatisering, modellen en prototyping. Ook problemen van MDD zoals beschreven in de literatuur komen terug in Fresnel Forms, zoals onvoorspelbaarheid en low ceiling. Mogelijke oplossingen voor deze problemen zijn beschreven in de desbetreffende hoofdstukken.

Fresnel Forms laat zien hoe een MDD benadering het maken van informatiesystemen voor het Semantic Web vereenvoudigt. Een belangrijke innovatie van Fresnel Forms is dat de resulterende informatiesystemen het browsen en toevoegen van informatie op het Semantic Web makkelijker maken.

[64] `http://www.mediawiki.org/wiki/Extension:Javascript`

Referenties

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (2015, 01 25). *Fresnel - Display Vocabulary for RDF*. Opgehaald van W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (2015, 02 01). *Cascading Style Sheets, level 2 CSS2 Specification*. Opgehaald van W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (p. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (2014, 13 12). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (sd). *Help:Import vocabulary*. Opgeroepen op May 3, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (sd). *Help:RDF Export*. Opgeroepen op May 2, 2015, van Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (pp. 1-27). Springer.

The Internet Engineering Task Force. (2015, 01 17). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Opgehaald van https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (pp. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (pp. 1-10). Bucharest: Matrix ROM.

W3C. (2015, 05 01). *Fresnel - Display Vocabulary for RDF*. Opgeroepen op January 11, 2015, van W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking

Bijlage XVIII: De onderzoeks context van het ABI project Fresnel Forms

Open Universiteit
Faculteit Informatica

18-9-2017
versie 1.0
J. N. Theunissen
838573218

Inhoud

De onderzoeks context van het ABI project Fresnel Forms 170
Doel 172
Inleiding 172
Zoektocht 173
Vragen 174
Antwoorden 174
Conclusie 175
References 176

Doel

Doel van dit document is het bepalen van de onderzoekscontext van het Afstudeer Bachelor Informatica (ABI) project “Fresnel Forms”. Om dit te bepalen wordt eerst bekeken hoe het project binnen het onderzoek van Lloyd Rutledge valt zoals beschreven in zijn paper (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013). Daarna wordt bekeken hoe dit onderzoek staat ten opzichte van het onderzoek van het Semantische Web.

Het ABI project is ook een uitwerking van het onderzoek dat Falco Paul gedaan heeft op het gebied van sortering van properties van klassen zoals beschreven in zijn master scriptie **Invalid source specified.**. Er wordt gekeken hoe dit onderzoek valt binnen het onderzoek op het gebied van het semantisch web.

Inleiding

Het onderzoek van Lloyd Rutledge richt zich op het automatisch genereren van userinterfaces om een bron ontology te onderhouden. Hij heeft zich gericht op de Semantic Media Wiki als content management systeem en Protege als ontology editor.
De mapping tussen de ontology en de Semantic Media Wiki wordt tot stand gebracht door middel van Fresnel **Invalid source specified.** en onze uitbreidingen op de Fresnel ontology. Ons project laat zien dat deze mapping mogelijk is. Aangezien we slechts een klein deel van de voorgestelde mappings geimplementeerd hebben, is de eerste vraag aan Lloyd Rutledge of voor de nog openstaande mappings al oplossingen bedacht zijn of dat het nog open punten zijn waarvoor oplossingen bedacht moeten worden. Dat bepaald de belangrijkheid van de ABI projecten voor het onderzoek, zijn het slechts projecten die bestaande oplossingen implementeren of zijn het projecten die bewijzen dat de onderzoeksvraag beantwoord kan worden.

Om de onderzoeks context van ons project goed te bepalen hebben we een onderzoek gedaan
op het web naar verschillende onderzoek papers die gepresenteerd zijn op de jaarlijkse het International Semantic Web Conference (ISWC) congressen.

Op basis van onze zoektocht naar relevante onderzoeken krijgen we een goed beeld van wat er gaande is op het gebied, userinterface generatie, waar ons project zicht op afspeelt. Daarna kunnen we vragen opstellen aan Lloyd Rutledge om duidelijk te krijgen welke keuzes hij gemaakt heeft voor dit onderzoek en welke rol ons project daarin gespeeld heeft.

Zoektocht

Om een indruk te krijgen van het probleem gebied is gezocht naar de papers/projecten die ontologien en userinterfaces als onderwerp hadden. Hieronder een overzicht op volgorde van publicatie jaar.

1. 2005: Arago is een op fresnel gebaseerde presentatie engine.
   From Graph To GUI: Displaying RDF Data From the WEB with Arago
   Invalid source specified.
2. 2006: Introductie Fresnel
   Invalid source specified.
3. 2010: RaUL: RDFa User Interface Language – A Data Processing Model for Web Applications
   Dit document beschrijft RDFa een taal voor embedding RDF in attributes on html.
   http://www.w3.org/TR/rdfa-syntax/
   http://www.w3.org/TR/rdfa-primer/
   http://microformats.org/
   Invalid source specified.
4. 2010, OWL-PL: A Presentation Language for Displaying Semantic Data on the Web
   Onderwerp is de OWL-PL taal om RDF te transformeren naar XHTML en CSS.
   Invalid source specified.
5. 2014: Levert ‘LD Viewer – linked data presentation framework op’ uit 2014
   Document beschrijft hoe de LD viewer bedoelt voor de dbpedia ontology uitgebreid is om algemener toegepast te worden.
   Invalid source specified.

Om alternatieven te vinden voor Fresnel is gezocht naar papers gepubliceerd na 2006 omdat Fresnel is gepresenteerd op het ISWC van 2006. Er zijn echter geen papers over alternatieven voor Frensel kunnen gevonden, dit leidt tot de vraag aan Lloyd Rutledge of hij bekend is met enig alternatief.

Daarna is er verder gezocht op projecten/papers met als onderwerp het automatisch genereren van userinterfaces of presentaties voor ontologien.

Op volgorde van tijd zijn de volgende papers/ projecten gevonden:

1. 2006: Ontowiki: Ontowiki is gemaakt door de universiteit van Leipzig. Deze wiki is speciaal opgezet om een bron ontology automatisch te onderhouden. Vraag aan Lloyd of hij bekend is met dit project en wat de voordelen van zijn aanpak zijn t.o.v. van dit project.
2. 2010: GAFFE framework, beschrijft een methode om van ontology naar wiki te geraken. Met forms en infoboxes. Zie bestand ‘di-iorio-2010-ontology-driven-generation-wiki.pdf’
3. 2013: ActiveRaUL: A Web form-based User Interface to create and maintain RDF data
4. 2013: Dominik Filipiak, Agnieszka Lawrynowicz. Generating Semantic Media Wiki Content from Domain Ontologies (short paper).

Om een indruk te krijgen van de onderzoeken op het gebied van het automatisch bepalen van volgorde is gezocht op sortering en ranking van properties. Er zijn geen relevante onderzoeken gevonden op dit gebied. Wel wordt bij het activeRaUL project vermeld dat ranking of properties als een toekomstige uitbreiding wordt gezien. Bij Lloyd Rutledge navragen waarom dit onderwerp zo weinig aandacht krijgt.

Vragen

1. Ons project heeft enkele mappingen zoals in het “OWFdev.xlsx” excell sheet gedefinieert geimplementeerd. Wat betekend het voor uw onderzoek als alle mappingen zijn geimplementeerd?
2. Wat zijn de voordelen van de aanpak zoals voorgesteld door Lloyd Rutledge zijn paper met Semantic Media Wiki t.o.v. het OntoWiki project?
   http://ceur-ws.org/Vol-273/paper_91.pdf
   http://en.wikipedia.org/wiki/OntoWiki
   https://github.com/AKSW/OntoWiki/wiki
   http://aksw.org/view/OntoWiki
3. Wat is de reden geweest om het onderzoek van Falco Paul te starten?
4. Waarin onderscheid het onderzoek van Lloyd Rutlegde zich ten opzichte van het onderzoek van Agnieszka Lawrynowicz.
   http://www.cs.put.poznan.pl/alawrynowicz/swcs14.pdf
   https://github.com/mimol/owl2wiki

Antwoorden

1. Het abi project bewijst dat de aanpak al of niet goed is. Het project hoeft dus niet volledig geimplemeneteerd te worden om dat aan te tonen. Hier geld de 20-80 procent regel 20 % van de code geeft 80 % van de functionaliteit en toont aan dat de aanpak al of niet het gewenste resultaat geeft. Veel onderzoeks projecten zijn op deze manier opgezet en zijn dus vaak niet af.
   Het doel van het onderzoek is bepalen of een semantic media wiki’s een bruikbare userinterface is voor de machinable readable data van het semantic web. Daarvoor wordt aangetoond dat er een makkelijker manier is om de wikipedia infoboxes te genereren.
2. Ontowiki is genoemd in de papers van Loyd Rutledge van 2011 **Invalid source specified.** en 2013 (Rutledge, From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces, 2013). Deze papers geven aan dat de functionaliteit van Semantic Media Wiki veel breder is dan het onderzoeks project OntoWiki. De integratie met Linked Data Sources en de ondersteuning van OWL constructies maken Semantic Media Wiki een betere kandidaat. Het OntoWiki project is opgezet om rdf triples te genereren. Het argument dat de export van de Semantic Media Wiki niet toereikend is vanwege het beperk aantal datattypes is niet doorslag gevend. De Semantic Media Wiki bied daar een oplossing voor in de vorm van een plugin/extensie. Daarnaast wordt in een ontology, restricties niet zo strikt nageleeft als in een RDB. Een ontology is opgezet met het Open world assumption en een RDB is opgezet met de Closed world assumption.
3. Voor een goede userinterface is de groepering en sortering van properties belangrijk. Dit inzicht komt voort uit het MDD methode van software ontwikkelen. Alle MDD systemen bieden mogelijkheden om de sortering en groepering van propeties aan te passen. Bij deze systemen hoeft de groepering en sortering echter niet automatisch bepaald te worden. Daarom is er ook weining onderzoek naar gedaan. Falco Paul heeft met zijn onderzoek aangetoond dat het mogelijk is om automatisch een goede interface te genereren.

4. Het onderzoek van Agnieszka Lawrynowicz verraste Lloyd Rutledge enigszins. Het onderzoek volgde op de presentatie van Lloyd Rutledge van zijn onderzoek. De aanpak van Lloyd Rutledge is dat de gegenereerde userinterface op Fresnel is gebaseerd. Daarmee zou de gegenereerde userinterface-definitie toepasbaar zijn voor alle op frensel gebaseerde userinterface systemen. Helaas is het zo dat het heel stil is in de onderzoekstak van de op fresnel gebaseerde browser voor het Semantic Web.

   Wat zijn de verschillen in aanpak?

   De conversie geschied op basis van 3 configuratie files en een of meerder ontology bestanden. De configuratie files dienen handmatig opgesteld te worden. De Template file bepaald welke attributen meegenomen worden. De mapping file koppelt de templates aan entity URI's. De conversie genereerd artikel, category en template pages.
   Voor de artikel pagina's wordt de categorien bepaald aan de hand van de classe, de equivalente classe en de parent classe. Dat doen wij niet, een lens staat voor een classe en zijn equivalente classe. Per lens wordt een categorie aangemaakt.
   Per individual wordt een artikel pagina aangemaakt. Dat doen wij niet, individuals worden niet meegenomen. Frensel Forms genereerd wel een forms pagina voor de invoer van individuals in de Semantic Media Wiki.

Conclusie

Hoe valt dit project binnen het onderzoek van Lloyd Rutledge?

Het ABI project heeft aangetoond dat de aanpak, zoals Lloyd Rutledge heeft gepresenteerd in zijn paper, het mogelijk maakt om automatisch een goed gelijkende infobox te genereren op basis van een bron ontology. Daarnaast is aangetoond dat een invoer formulier gegenereerd kan worden waarmee data ingevoerd kan worden op een gebruikersvriendelijke manier. De autocompletion is daar een mooi voorbeeld van. De Semantic Media Wiki biedt de mogelijkheid om de ingevoerde gegevens te exporteren naar een RDF bestand.

Relatie van dit onderzoek Lloyd Rutledge t.o.v. het onderzoek op het SW?

Het onderzoek van Lloyd Rutledge valt binnen het onderzoek op het gebied van het Semantic Web om de machine readable gegevens toegankelijk te maken voor gebruikers. Hiervoor wordt onderzoek gedaan naar userinterfaces voor semantische gegevens. Het onderzoek van Lloyd is een onderdeel van de onderzoeken die zich richten op het gebruik van Semantic Media Wiki's als interface naar het semantische web. De hoop volgens LR is dat deze userinterfaces voor het semantic web dezelfde boost in semantische gegevens kunnen geven als de webpage editors het internet populair gemaakt hebben. In beide gevallen wordt de gebruiker afgeschermt van de achterliggende complexiteit.

Doel onderzoek Falco Paul

Het onderzoek van Falco Paul naar het automatisch bepalen van volgordelijkheid van properties van een klasse is een onderdeel van het onderzoek naar het automatisch genereren van userinterfaces die aantrekkeijk zijn voor gebruikers. Als basis is de Wiki Infobox genomen die naar jaren finetunen gezien worden als aantrekkelijk en informatief. De

volgorde en de groepering van de properties zijn een belangrijke parameter in de kwaliteit van de infobox.

References

Allemang, D., & Hendler, J. (2008). *Semantic Web for the Working Ontologist.* Burlington: Elsevier Inc.

Antoniou, G., & van Harmelen, F. (2008). *A Semantic Web Primer.* Massachusetts Institute of Technology.

Atkinson, C., & Kühne, T. (2003). Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 36-41.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 29-37.

Bizer, C., Lee, R., & Pietriga, E. (25 de 01 de 2015). *Fresnel - Display Vocabulary for RDF*. Obtenido de W3C: http://www.w3.org/2005/04/fresnel-info/manual/

Bos, B., Håkon, W., Lilley, C., & Jacobs, I. (01 de 02 de 2015). *Cascading Style Sheets, level 2 CSS2 Specification*. Obtenido de W3C: http://www.w3.org/TR/REC-CSS2/

Brenninkmeijer, T., & Zwanenberg, T. (2014). *Protégé-OWL MDD Fresnel Plug-in.* Bachelor's group project thesis, Open Universiteit.

Ławrynowicz, A., & Filipiak, D. (2014). Generating Semantic Media Wiki Content from Domain Ontologies. *SWCS'14 Third International Workshop on Semantic Web Collaborative Spaces*, (pág. 49).

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., & Bizer, C. (2014). DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*.

Martinez-Cruz, C., Blanco, I., & Vila, M. (2012). Ontologies versus relational databases: Are they so different? A comparison. *Artificial Intelligence Review*, 271-290.

Mekkering, A. (12 de 13 de 2014). Verbeteren door hergebruik. Olst, Overijssel.

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction in the new millennium, Part 1*, 3-28.

Paul, F. (2014). *Property Ranking Approaches For Semantic Web Browsers - A Review of Ontology Property Ranking Algorithms.* Master's thesis, Open Universiteit.

Pietriga, E., Bizer, C., Karger, D., & Lee, R. (2006). Fresnel: A Browser-Independent Presentation Vocabulary for RDF. *The 5th Semantic Web Conference.* Athens, Georgia.

Rutledge, L. (2013). From Ontology to Wiki – Generating Cascadable Default Fresnel Style from Given Ontologies for Creating Semantic Wiki Interfaces. *Proceedings Workshop on Semantic Web Collaborative Spaces (SWCS2013).* Montpellier.

Rutledge, L. (2015 verwacht). From Ontology to Semantic Wiki – Designing Annotation and Browse Interfaces for Given Ontologies. *Semantic Web Collaborative Spaces (SWCS) 2012/2013/2014 LNCS post-proceedings.* Springer-Verlag.

Selic, B. (2003). The Pragmatics of Model-Driven Development. *IEEE Software*, 19-25.

Semantic MediaWiki. (s.f.). *Help:Import vocabulary*. Recuperado el 3 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:Import_vocabulary

Semantic MediaWiki. (s.f.). *Help:RDF Export*. Recuperado el 2 de May de 2015, de Semantic MediaWiki: https://semantic-mediawiki.org/wiki/Help:RDF_export

Szekely, P. (1996). Retrospective and challenges for model-based interface development. *Design, Specification and Verification of Interactive Systems '96* (págs. 1-27). Springer.

The Internet Engineering Task Force. (17 de 01 de 2015). *RFC 3986 - Uniform Resource Identifier (URI): Generic Syntax*. Obtenido de https://tools.ietf.org/html/rfc3986

Van den Berg, J., Meixner, G., Breiner, K., Pleuss, A., Sauer, S., & Hussmann, H. (2010). Model-driven development of advanced user interfaces. *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (págs. 4429-4432). Chicago: ACM.

van Vliet, H. (2008). *Software Engineering: Principles and Practice, third edition.* Chichester (UK): Wiley.

Vanderdonckt, J. (2008). Model-driven engineering of user interfaces: Promises, successes, failures, and challenges. *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction ROCHI'2008* (págs. 1-10). Bucharest: Matrix ROM.

W3C. (01 de 05 de 2015). *Fresnel - Display Vocabulary for RDF*. Recuperado el 11 de January de 2015, de W3C: http://www.w3.org/2005/04/fresnel-info/manual/#csshooking