Semesteroppgave INFO116

Høst 2015



Innlevering av: 100, 154, 216

Innholdsfortegnelse

# 1. Innledning

Rapporten er en samling og forklaring av vårt arbeid med semesteroppgave i INFO116.

## 1.1. Hvordan vi har jobbet med oppgaven

Vi har gjennom hele prosjektet jobbet sammen alle 3 gruppemedlemmer på alle deler av prosjektet. Dette har vært viktig for oss, for å få en dypere forståelse av hverandres arbeid og spesielt siden alle deler av oppgaven påvirker hverandre. Vi brukte lengst tid på å lese oss opp på Wittengsteins filosofi, de tilhørende artiklene og lage ontologien, da dette legger grunnlaget for resten av oppgaven. Vi har med vår inndeling prøvd å gjenspeile hovedlinjene i Wittgensteins filosofi på en forenklet men spesifikk måte, for å få en logisk inndeling. Da ontologien var fullført begynte vi med å legge inn artiklene, semantisk annotering og SPARQL spørringer parallelt, sammen med vurdering av ontologien. Dette hadde med at da vi gikk dypere inn i annoteringen og spørringene, påvirket det deler av ontologien. Dette resulterte i mindre endringer på ontologien, men gav til gjengjeld en mer korrekt ontologi og annotering, som henger sammen med resten av prosjektet. Vi hadde på forhånd lagt en plan for arbeidet som vi i stor grad klarte å følge, som illustrert i bilde 1.



#### Bilde 1: Tidsplan for prosjektet

# 2. Ontologi

## 2.1. Hva er en ontologi

En ontologi er en formell spesifikasjon av en konseptualisering. Ontologier brukes til å fange opp kunnskap om domener av interesse. En ontologi beskriver begrepene i domenet og også relasjoner som holder mellom disse begrepene. For å lage en ontologi må man bryte ned et konsept i klasser, dele de inn i et hierarki, definere relasjoner og lovlige verdier for disse.

## 2.2. Omfang for ontologi

For å definere omfanget til en ontologi er det nødvendig å finne ut hva man skal kunne bruke den til, hvilke spørsmål ontologien skal kunne besvare og hvilke datakilder som er tilgjengelige for å fylle ut kunnskapsbasen.

### 2.2.1. Bruksområder

Vi ønsker at vår ontologi skal brukes til å semantisk annotere verk av Wittgenstein og andre forfattere slik at en bruker kan finne litteratur etter innhold, forfatter og temaer som artiklene håndterer. Flere forfattere skal derfor kunne legges inn og hjelpe brukeren i å finne relevante artikler for sine søk om filosofiske temaer basert på temaer innen Wittgensteins filosofi.

### 2.2.2. Spørsmål ontologien skal besvare

Ontologien skal kunne brukes til å hente ut artikler som omhandler samme tema eller andre egenskaper ved artikkelen som forfatter, årstall osv. Ontologien skal derfor hjelpe brukeren å finne artikler basert på dens preferanser. Ønskede spørsmål som skal besvares er:

* Hvilke artikler diskuterer tema x?
* Hvilke forfattere og litterære verk diskurerer x og kommenterer artikkelen y?
* Hvilke artikler x er publisert i by y?
* Hvilke litterære verk er publisert etter år x?
* Hvilke forfattere diskuterer tema x?

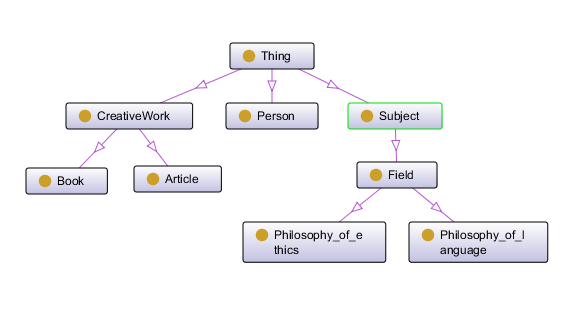
### 2.2.3. Tilgjengelige datakilder

Tilgjengelige datakilder vi har er tildelte artikler utgitt av Wittgenstein og andre forfattere som skriver om hans filosofi, som kan brukes til å fylle ut kunnskapsbasen. Vi har også informasjon og tekster fra The Wittgenstein Archives at the University of Bergen (WAB).

## 2.3. Ontologi

### 2.3.1. Illustrasjon av ontologi

Klasser blir tolket som sett som inneholder individer, som representerer objekter i domenet og beskriver konsepter i domenet. For vår ontologi har vi valgt å definere superklassene “Person”, “CreativeWork” og “Subject”.



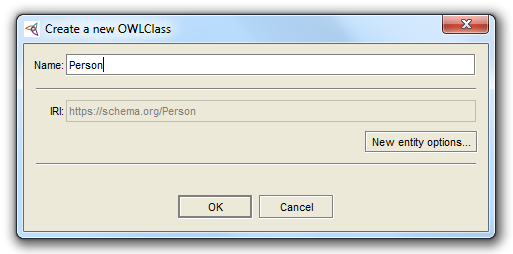
#### Bilde 2: Klassehierarki

### 2.3.2. Klasser, individer, relasjoner og attributter

Klassene i vår ontologi er disjunkte, som vil si at en instanse av Person ikke kan være en instanse av "CreativeWork" da disse er to forskjellige ting. Dette er for å definere at to klasser ikke kan være like, uavhengig av hvordan klassene blir tolket, og fordi OWL klasser som standard går ut i fra at to klasser kan overlappe.

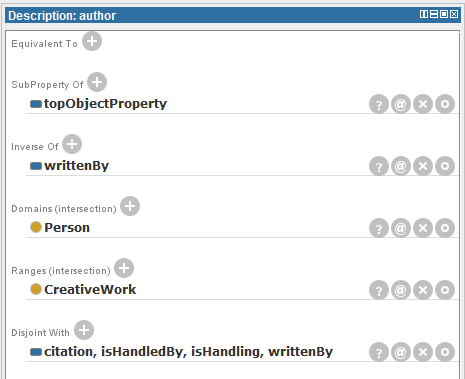
#### Person

Klassen Person er knyttet til schema.org sitt skjema “Person”. Vi velger å benytte denne fordi det er en veletablert og mye brukt ontologi. Vi knyttet klassen til schema.org i Protege ved å endre entitetens URL, og har gitt den tilsvarende navn som den har på schema.org som vist på bilde 3.



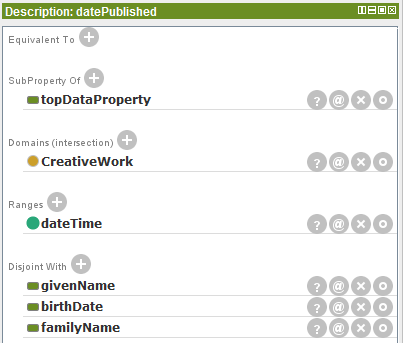
#### Bilde 3: Eksempel på hvordan vi har lagt til skjema fra schema.org i ontologien

Person har de tilhørende relasjonene “author”, og relasjonen “writtenBy”, som er den inverse relasjonen av “author”. “Author” har domene definert som “Person”, og range “CreativeWork”, som lovlige verdier for relasjonen. “Author” kan dermed kun brukes mellom en person og et litterært verk.

#### Bilde 4, 5: Lovlige verdier for relasjonen author og grafisk fremstilling av relasjon

“Person” har også flere data properties; “givenName”, “familyName” og birthDate”, for å knytte navn og fødselsdato til individene. Attributtet “givenName”, “familyName” er definert med datatypen “string”, og “birthDate” er definert med datatypen “dateTime”.



#### Bilde 6: Lovlige verdier for data property “datePublished”

Vi har lagt til syv individer under personer på bakgrunn av hvem som snakkes om og har skrevet de litterære verkene vi har benyttet.

**

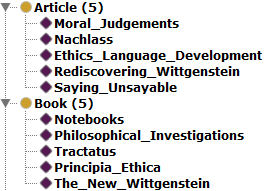
#### Bilde 7: Individer Person

På forfattere hvor vi kun har funnet årstall for fødsel, har vi antatt at dato for fødsel er 1 januar, dette gjelder også for litterære verk. Vi mener det er årstall som er mest essensielt i spørringer, og at det derfor ikke ha stor innvirkning på resultatene av spørringene.

#### Litterature

Superklassen "CreativeWork" har subklassene “Article” og “Book”, som også er knyttet fra schema.orgs skjema “CreativeWork” på samme måte som “Person”. Til "CreativeWork" har vi lagt til data properties “placePublished”, “datePublised” og “name”. Dette er for å definere når og hvor et litterært verk er utgitt. “PlacePublished” og “name” er definert som en “string”, og “placePublished” er definert med “dateTime”.

“CreativeWork” har totalt 10 individer fordelt likt på artikler og bøker ut fra tekstene vi har semantisk annotert.



#### Bilde 8: Individer av “CreativeWork”

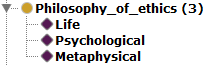
#### 

#### 

#### Subject

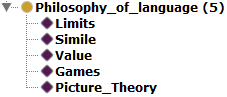
Superklassen “Subject” har underklassen “Field”, som igjen har subklassene “Philosophy\_of\_ethics” og “Philosophy\_of\_language” som er hovedtemaene Wittgenstein omtaler i sin filosofi. Vi har valgt å skille mellom “Philosophy\_of\_ethics” og “Philosophy\_of\_language”, ettersom hans filosofi basert på de utgitte artiklene omhandler overordnet disse temaene. Det er til tider en overlapp mellom de to temaene, da hans filosofi om språk gjennomsyrer store deler av hans filosofi, men vi har valgt å skille på det for å kunne finne artikler som snakker mer spesifikt om etikk som tema og ikke har hovedfokus på språk.

Under “Philosophy\_of\_ethics” har vi opprettet individene “metaphysical”, “psychological” og “life” fordi det er de tre hovedskillene som snakkes om innen Wittgensteins filosofi i artiklene.



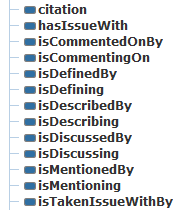
#### Bilde 9: Individer under “Philosophy\_of\_ethics”

“Philosophy\_of\_language” har individene “Games”, “Limits”, “Meaning”, “Picture\_Theory”, “Simile” og “Value”. Vi har valgt å lage flere individer av dette temaet fordi det er et svært stort tema og Wittgenstein går i detalj på alle disse temaene. Det er derfor interessant å kunne skille på dette i ontologien for å kunne finne mer spesifikt hva han mener om de ulike undertemaene. “Value” er en samleklasse for alt som handler om verdi, som er temaet som snakkes mest om. “Value” blir omtalt som sense, nonsense, absolute- og relative value i artiklene, noe som byttes på alt etter hvem som har skrevet artikkelen. Vi har derfor valgt å kalle den “Value”, som omhandler dem alle.



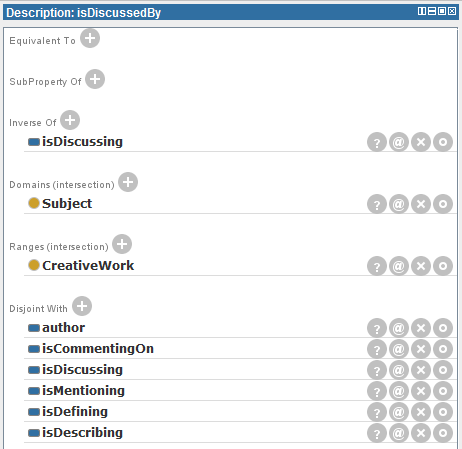
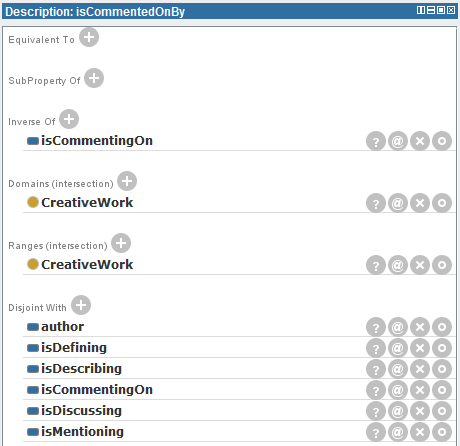
#### Bilde 10: Individer “Philosophy\_of\_language”

“Subject” benytter også en data property “name”, for å gi navn på de litterære verkene. For å relatere artikler til de ulike temaene i fields har vi opprettet relasjonene “citation”, “isCommentingOn”, “isDefining”, “isDescribing”, “isDiscussing”, “hasIssueWith” og “isMentioning” og de inverse relasjonene “citedBy”, “isCommentedOnBy”, “isDefinedBy”, “isDescribedBy”, “isDiscussedBy”, “isTakenIssueWithBy” og “isMentionedBy”.



#### Bilde 11: Relasjoner knyttet til “Subject”

Dette er for å skille på hvordan eller hva som diskuteres i tekstene og kunne søke mer spesifikt på dette. Noen artikler nevner kun enkelte undertemaer, mens andre diskuterer temaer i dybden. Det vil derfor ved søk være lettere å finne det man ønsker ved å skille på hvordan et tema i litterært verk er håndtert. “isCommentingOn” er for eksempel ment for å brukes når en artikkel håndterer en annen artikkel i sin tekst, “isCommentingOn” kan dermed ligne på “isDiscussing”, men vi har bevisst valgt å skille på disse da “isDiscussing” kun er knyttet til temaer og ikke artikler som “isCommentingOn” er. Dette gjenspeiles også i hvordan domenet og rekkevidden er definert for, som vist på bildene under:



#### Bilde 12, 13: Domene og range for “isCommentedBy” og “isDiscussedBy”

# 3. SPARQL

SPARQL (**S**PARQL Protocol And RDF Query Language) er et spørrespråk som kan hente ut og manipulere semantiske data lagret som RDF tripler fra kilder og databaser. Våre SPARQL-spørringer som er presentert under benytter samme prefikser:

PREFIX rdf: <<http://www.w3.org/1999/02/22-rdf-syntax-ns#>>

PREFIX owl: <<http://www.w3.org/2002/07/owl#>>

PREFIX xsd: <<http://www.w3.org/2001/XMLSchema#>>

PREFIX rdfs: <<http://www.w3.org/2000/01/rdf-schema#>>

PREFIX : <[http://www.semanticweb.org/vsj010/ontologies/2015/10/untitled-ontology-4#](http://www.semanticweb.org/vsj010/ontologies/2015/10/wittgenstein-philosophy#)>

Alle spørringene baserer seg på og prøver å besvare spørsmålene vi ønsker at ontologien skal besvare, og har vært utgangspunkt for spørringene vi har laget.

## 3.1. Spørring 1

I den første spørringen ønsker vi å finne navn på forfatter og litterære verk som både omtaler et tema og kommenterer en artikkel. Dette er et eksempel på hvordan man kan finne en artikkel ved å søke på hvordan den er håndtert, hva slags tema den snakker på og om den kommenterer en spesifikk artikkel i kombinasjon. Dette kan gi brukeren relativt spesifikke søk på artikler. I eksempelet under søkes det spesifikt etter forfattere og artikler som diskuterer verdi og kommenterer artikkelen “Lecture on Ethics”:

SELECT ?fname ?lname ?litterature

WHERE {

?x :isDiscussing :Value ;

:isCommentingOn :Nachlass ;

:name ?litterature ;

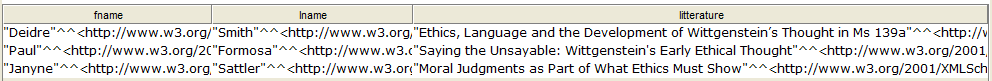
:writtenBy ?y .

?y :givenName ?fname ;

:familyName ?lname .

}

Resultatet blir som vist i bilde 14:



#### Bilde 14: Resultat for spørring 1

## 3.2. Spørring 2

Spørring 2 illustrerer hvordan man kan søke opp en spesifikk forfatter på navn, som har kommentert på en spesifikk artikkel. I eksempelet søker vi etter en forfatter med navn Smith med filtrering på etternavnet og har kommetert artikkelen “Lecture on Ethics”:

SELECT ?fname ?lname ?litterature

WHERE {

?x :writtenBy ?author ;

:name ?litterature ;

:isCommentingOn :Nachlass .

?author :givenName ?fname ;

:familyName ?lname .

FILTER regex(?lname, "smith", "i")

}

Resultatet blir som vist i bilde 15:



#### Bilde 15: Resultat for spørring 2

## 3.3. Spørring 3

Denne spørringen vil returnere en liste over alle verkene som har publiseringsdato fra en angitt dato og sortert fra tidligst til senest publisert. I vårt eksempel søker vi etter artikler som er publisert etter 01.01.2000.

SELECT ?article ?published

WHERE {

?x :datePublished ?published ;

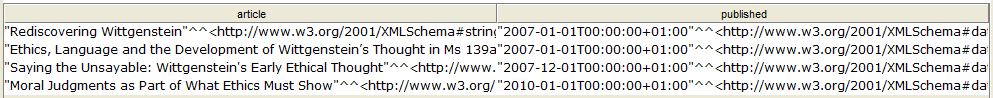
:name ?article .

FILTER (?published > "2000-01-01T00:00:00+01:00"^^xsd:dateTime)

}

ORDER BY ASC(?published)

Resultatet blir som vist i bilde 16:



#### Bilde 16: Resultat for spørring 3

## 3.4. Spørring 4

Spørring nummer fire vil returnere de artiklene som har både blitt publisert på en spesifikk plass og etter en spesifikk dato. Eksempelet vårt gir oss de artiklene som ble publisert i Bergen og har en publiseringsdato som er fra 01.01.2000.

SELECT ?name ?date ?place

WHERE {

?x :placePublished ?place ;

:name ?name ;

:datePublished ?date .

FILTER (regex(?place, "bergen", "i") && (?date > "2000-01-01T00:00:00+01:00"^^xsd:dateTime))

}

ORDER BY ASC(?name)

Resultatet blir som vist i bilde 17:



*Bilde 17: Resultat for spørring 3*

## 3.5. Spørring 5

Spørring 5 er en construct spørring som lager nye tripler. Vi har i denne sammenheng laget nye tripler for å finne ut hvilke forfattere som håndterer et gitt tema. I vårt eksempel lager vi tripler for forfattere som diskuterer value. Spørringen går dermed via hvilke artikler forfatteren har skrevet og om artikkelen diskuterer value, for å så gi hvilke forfattere som diskuterer value.

CONSTRUCT {

?author :isDiscussing :Value

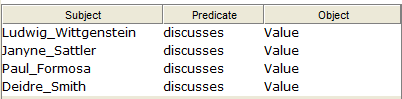
} WHERE {

?author :author ?creativeWork .

?creativeWork :isDiscussing :Value .

}

Spørringen gir da 4 nye tripler og får resultatet som vist i bilde 18:



#### Bilde 18: Resultat for spørring 5

# 4. Semantisk annotering

I prosjektet har vi semantisk annotert fire HTML-dokumenter. I annoteringen brukte vi både RDFa og JSON-LD formatene. Vi valgte å kun annotere artiklenes tekstinnhold, ikke hele siden med menyer og annet, da de inneholder mye kildekode som er irrelevant for vår ontologi og vil dermed bedre vise hvordan vi semantisk har annotert artiklene.

## 4.1. RDFa

Vi har benyttet schema.org og vår egen ontologi i annoteringen av sidene. De skilles mellom ved å bruke ulike prefikser, “schema:” for schema.org og “wit:” for vår ontologi. Dette er vist i bilde 19, og er plassert i bodyen til artikkelens HTML side. Sammen med prefiksene i bodyen defineres hva slags objekt som triplene refererer til. De annoteres dermed med “typeOf” og “resource” som vist i eksempelet under:



#### Bilde 19: Visning av bruk av prefikser og definerering av objekt som annoteres

I artiklene er informasjon som forfatter, dato og sted publisert presentert. Disse er annotert ved å bruke <span> elementer, med tilhørende literaler og verdi. For literater som ikke har samme formatering som den vi benytter i ontologien, legges disse til med rett datatype i “content”, som vist i eksempelet under med “datePublished”. “LocationCreated” er i dette eksempelet med rett verdi og har derfor ikke benyttet “content” i span taggen.



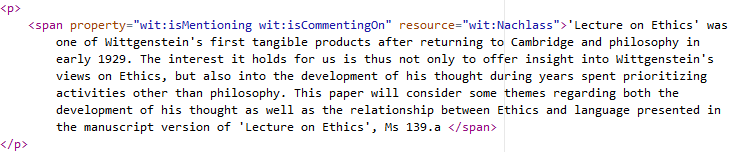
#### Bilde 20: Eksempel på annotering

Forfattere er i artiklene definert som egne objekter, da de er av annen type enn artikkelen.



#### Bilde 21: Eksempel på annotering av forfatter

Ved annotering av hva artiklene omhandler og hvilke relasjoner som de er knyttet til er også span-tagger benyttet eller skrevet rett inn i avsnitt-taggene <p>. “Property” henviser til hvilken relasjon eller literal som benyttes og “resource” til hvilket individ det omhandler. I eksempelet under nevnes og kommenteres artikkelen “Lecture\_of\_Ethics” som er individet som heter Nachlass i vår ontologi:



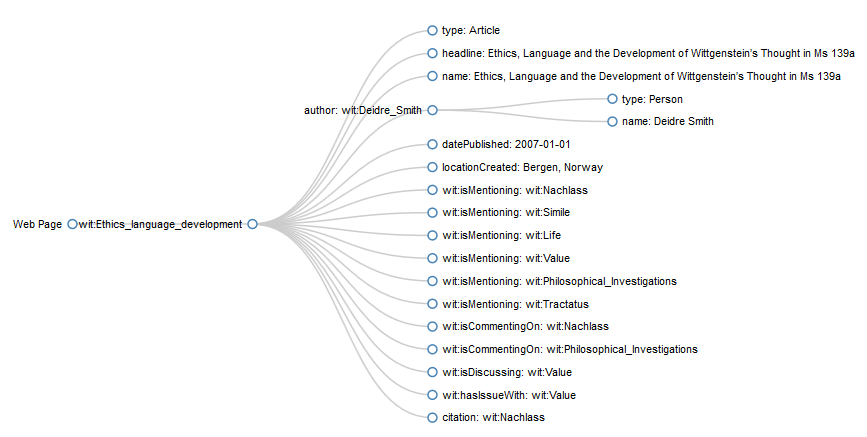
#### Bilde 22: Eksempel på annotering av artikkelsen kobling til ulike verk

Ved annotering til ulike temaer under “subject” er de annotert på tilsvarende måte, som vist i bilde under:

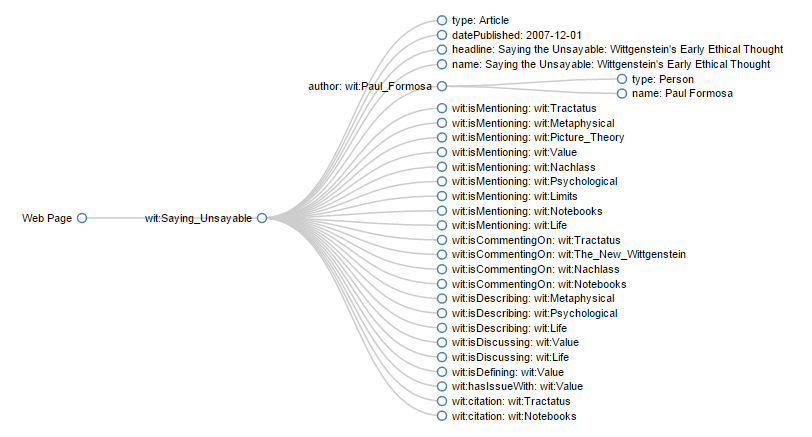


#### Bilde 23: Eksempel på annotering av artikkelsen kobling til et “subject”

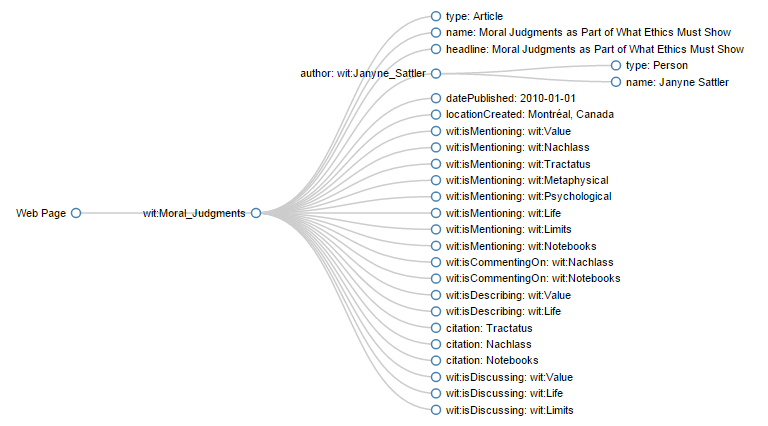
Alle HTML sidene er annotert på samme måte, og får følgende grafer ved visualisering av RDFa annoteringen:



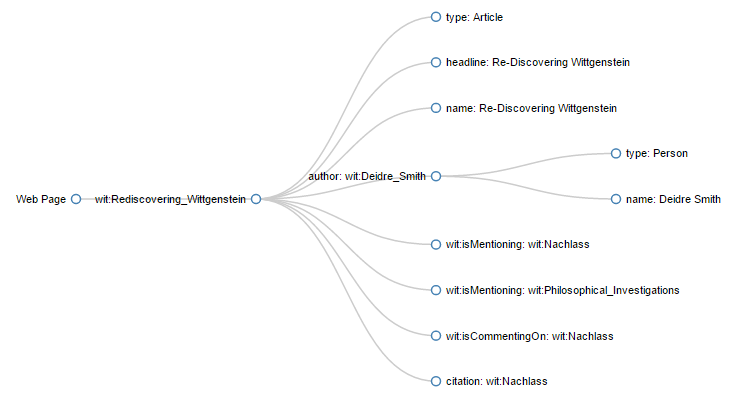
#### Bilde 24: Visualisering av RDFa. Ethics, Language and Development av Deidre Smith



#### Bilde 25: Visualisering av RDFa annoteringen av Saying the Unsayable av Paul Formosa



*Bilde 27: Visualisering av RDFa annoteringen. Rediscovering Wittgenstein av Janyne Sattler*



#### Bilde 26: Visualisering av RDFa. Rediscovering Wittgenstein av Deidre Smith

## 4.2. JSON-LD

I tillegg til RDFa-annoteringen har vi også laget JSON-LD filer (Javascript Object Notation for Linked Data) som er koblet sammen med HTML sidene som vist i bilde 28.



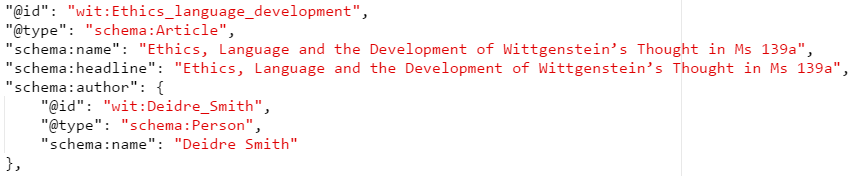
#### Bilde 28: Visning av kobling mellom HTML og JSON-LD sider

Filene gir mer kontekst for RDF-annoteringen i HTML-dokumentene ved å gi en sammenheng mellom object properties og konsepter i en ontologi. Første bildet under viser hvordan man først etablerer en “context” til de objektene og relasjonene som blir brukt.



#### Bilde 29: Context i JSON-LD annotering

Etter “context” ble fastsatt la vi til de individene som er relevante for vår ontologi og RDFa-annotering i de angitte sidene, og i tillegg etableres hvilke “type” individene er.



#### Bilde 30: Annotering av Ethics, Language and Development

## 4.3. Hva de annoterte sidene kan brukes til

Den semantiske annoteringen av HTML-dokumentene gjør at maskiner og for eksempel søkemotorer bedre kan forstå konteksten til informasjonen i de respektive dokumentene. Det at informasjonen blir maskinlesbar gir nye muligheter for bruk av datakildene. Tredjeparts programvare kan bruke den semantiske informasjonen til å automatisk forstå den fulle konteksten til forskjellige dokumenter ved å relatere informasjon fra dokumentene og dermed bestemme hvordan og hvorfor dokumenter kan bli kombinert og/eller brukt. Dette gir nye muligheter for kombinering og bruk av informasjon på nett og gir mer verdi til informasjonen som allerede eksisterer. Ved bruk av SPARQL og annen programvare kan informasjonen presenteres og benyttes på nye og interessante måter av andre og en selv. Med vår semantiske annotering av artiklene kan de dermed benyttes på nye og mer informative måter enn tidligere.

# 5. Oppsummering

Vi har i dette prosjektet laget en ontologi og brukt denne til å semantisk annotere HTML-sider med RDFa og JSON-LD og laget SPARQL-spørringer for å vise hvordan eventuelle web-apper og annen tredjeparts programvare kan bruke ontologien til å hente ut informasjon. I ontologien har vi prøvd å ta hensyn til at andre artikler som omhandler Wittgensteins filosofi, men som ikke ble inkludert i dette prosjektet, kan legges til ontologien slik at dens bruksområde ikke snevres inn til de fire artiklene prosjektet benytter.

# 6. Vedlegg

## Filer til ontologi og SPARQL

* ?????.owl
* sparql.txt

## Filer til semantisk annotering

JSON-LD filer

* ethics\_language\_development\_json.txt
* moral\_judgments\_json.txt
* rediscovering\_wittenstein\_json.txt
* saying\_unsayable\_json.txt

HTML filer

* ethics\_language\_development.txt
* moral\_judgments.txt
* rediscovering\_wittenstein.txt
* saying\_unsayable.txt

# 7. Kilder

semantic web boka

pizza dritten