**UNIVERSITETET I OSLO**
**Institutt for informatikk**

# Mobile, håndholdte enheter som plattform for semantisk vev-applikasjoner basert på åpne, offentlige data

Masteroppgave

Jens Kilde Mjelva

**2. mai 2011**

**Sammendrag**

Stadig flere offentlige data tilgjengeliggjøres som følge av det politiske fokuset som i økende grad rettes mot denne tematikken. Innovasjonspotensialet og den tilknyttede øknomiske veksten er spesielt vektlagt som argument på hvorfor vi trenger flere åpne, offentlige data. Semantikart er en applikasjon, utviklet i forbindelse med masteroppgaven, hvor dette potensialet utnyttes. Semantikart er utviklet til bruk på mobile, håndholdte enheter. Slike enheters støtte for innebygde verktøy bidrar til nye muligheter for applikasjonsutvikling. Ved å kombinere mobilitet med verktøy for lokalitetsstedfesting tilbys brukerne av Semantikart informasjon om ulike typer ting i geografisk nærhet av sin posisjon, eksempelvis turistattraksjoner eller ladestasjoner for elbil.

Problemene som oppstår i forbindelse med utvikling av applikasjoner basert på åpne, offentlige data, knyttes først og fremst til de ulike måtene slike data er tilgjengeliggjort på. Valget av formater og bruken av identifikatorer og representasjonsformer om de samme tingene varierer fra datasett til datasett. Gjennom semantiske vevteknologier tilbys en egnet måte å overkomme disse utfordringene på. Offentlige data publiseres derfor i økende grad på den semantiske veven, eksempelvis dataene Semantikart nyttiggjør seg av. Disse dataene er spredt blant ulike dataservere som Semantikart kommuniserer med, blant annet gjennom spørringer som henter data fra ulike kategorier innenfor et avgrenset geografisk område rundt applikasjonsbrukerens lokalitet.

**Takk til**

Først og fremst ønsker jeg å takke min veileder, Audun Stolpe, for utfyllende tilbakemeldinger, nyttige innspill og gode idéer. Å arbeide sammen med Audun har vært interessant, morsomt og veldig lærerikt.

Videre ønsker jeg å takke Computas AS og biveileder, David Norheim, som ga meg mulighet til å utvikle en semantisk vevapplikasjon på mobil plattform. Takk til Lotta Nordling og David for teknisk bistand på serversiden og til Odd-Wiking Rahlff og Daniel Johansson for hjelp med interaksjonsdesign.

Takk til Einar Myre for korrekturlesning og til Martin Giese for tilbakemeldinger og betrakninger knyttet til geometri i oppgavens kapittel 6: "Utfordringer".

En takk rettes også til Semicolon-prosjektet, spesielt Audun og de andre involverte fra IFI, for at jeg ble gitt anledning til å skrive en masteroppgave tilknyttet spennende tematikk og teknologi.

Til slutt ønsker jeg å takke min familie, venner og alle andre jeg godt liker som jeg kjenner.

# Innhold

# Figurer

# Tabeller

# Kapittel 1

# Innledning

## 1.1 Bakgrunn

Semicolon er et forskningsprosjekt der Institutt for informatikk ved Universitetet i Oslo (IFI), Computas AS, Det Norske Veritas, Brønnøysundsregistrene, Karde AS og Skattedirektoratet er blant de medvirkende partnerne. Denne masteroppgaven er et resultat av samarbeidet som i denne forbindelse har oppstått mellom Computas AS og IFI.

Økt elektronisk samhandling i offentlig sektor står sentralt som et av Semicolon-prosjektets to hovedmomenter. "Samhandling mellom etater, næringsliv og borgere er en nødvendig forutsetning for å effektivisere offentlig tjenesteproduksjon, samt for å heve kvaliteten i tjenestene. [1]" Prosjektets andre hovedmoment henger sammen med det første og dreier seg om åpne, offentlige data og økt tilgjengeliggjøring av slike. "...Det er veldig mye og veldig forskjellig type informasjon som ligger offentlig tilgjengelig i dag, ikke minst fordi en rekke etater er pålagt å publisere åpne data. [2]" Det argumenteres for at åpne data vil bedre informasjonsflyten mellom offenlige etater, bidra til effektivisering og øke felles dataforståelse. Med økt andel åpne data følger også et stort potensiale for innovasjon. Nye tjenester og applikasjoner kan spinnes rundt åpne data.

Innovasjonspotensialet og veksten som følger av dette er spesielt vektlagt som argument på hvorfor vi trenger flere åpne, offentlige data. Et av målene med denne masteroppgaven er å forsøke å vise hvordan dette potensialet kan utnyttes, og derigjennom viktigheten av slike data. Som en del av oppgaven har derfor en applikasjon basert på åpne, offentlige data blitt laget.

[1] http://semicolon.wiki.ifi.uio.no/Hovedside
[2] http://folk.uio.no/audus/Masterside

Dataene det var aktuelt å involvere i denne applikasjonen, stammer fra ulike kilder som foreligger i forskjellige formater. Det er ofte ikke opplagt hvordan dataene skal forstås. På tvers av datakilder kan data som representerer samme ting beskrives på ulike måter, mens data som svarer til ulike ting kan beskrives på samme måte. Dette er to eksempler på "semantisk heterogenitet".

*Semantisk vev-teknologier* har vist seg velegnede for å eksplisere betydning. Mange har sett fordelene slike teknologier av den grunn gir arbeid med semantisk heterogene data. Stadig flere offentlige data, blant annet de som nevnes i 5.3.3 på side 65, finnes derfor publisert på den semantiske veven som åpne, lenkede data beskrevet med Resource Description Framework og aksesserbare, blant annet, gjennom SPARQL (SPARQL Protocol and RDF Query Language)-endepunkter. Semantisk vev-teknologier står sentralt i Semicolon-prosjektet og er et viktig tema i denne masteroppgaven.

Åpne, offentlige data kan gi opphav til mange ulike semantisk vev-applikasjoner. Tradisjonelt sett er slike utviklet for å brukes i nettlesere på personlige datamaskiner. Andre, mer utradisjonelle plattformer kan gi videre muligheter for semantisk vev-applikasjoner.

Mobile, håndholdte enheter, deriblant mobiltelefoner, er en applikasjons-plattform som stadig vokser i utbredelse. Slike enheters mobilitet kombinert med mulighet for installasjon av tredjepartsprogramvare/applikasjoner og tilgang på innebygde verktøy som GPS og nettilkobling gir nye, spennende muligheter for applikasjonsutvikling. Denne masteroppgaven søker å dra nytte av potensialet som følger med denne plattformens unike muligheter. Den ovennevnte applikasjonen basert på åpne, offentlige data er derfor realisert på en plattform for mobile, håndholdte enheter under navnet "Semantikart".

## 1.2 Problembeskrivelse

Potensialet for nye applikasjoner er, som beskrevet ovenfor, stort både med utgangspunkt i åpne, offentlige data og på en plattform spesielt designet for mobile, håndholdte enheter. Summen av disse to områdenes muligheter for innovativ applikasjonsutvikling er enormt. Dette potensialet, åpne, offentlige data, anvendt på mobil plattform, står derfor i fokus i denne masteroppgaven, hvilket adresseres gjennom problemstillingen: *hvordan kan mobile, håndholdte enheter brukes som plattform for å lage en nyttig semantisk vev-applikasjon basert på åpne, offentlige data?*

## 1.3 Rapportens oppbygning

Denne masteroppgaven berører flere temaer og teknologier. I kapittel 2 beskrives åpne, offentlige data og det tilknyttede politiske bakteppet. Den semantiske veven og lenkede data introduserers i samme kapittel, 2.3 på side 17, mens semantisk vev-teknologier beskrives i større detalj i kapittel 3 på side 23.

Sammenhengen mellom de nevnte temaene analyseres i kapittel 4 på side 47. I dette kapittelet, 4.1 på side 47, beskrives også mobile, håndholdte enheter i større detalj. Analysen i dette kapittelet munner ut i applikasjonen Semantikart, slik beskevet i 4.2 på side 50. Semantikarts oppbygning som system beskrives nærmere i kapittel 5 på side 57 og implementasjonsmessige utfordringer knyttet til systemutviklingen forklares i kapittel 6 på side 71. I kapittel 7 på side 81 oppsummeres oppgaven, før to mulige utvidelser/forbedringer av Semantikart skisseres.

Rapporten inneholder også et tillegg A på side 91, som inneholder informasjon, tilknyttet Semantikart, om installasjon, applikasjonens tekniske oppbygning, kildekode og brukerveiledning.

# Kapittel 2

# Åpne data, lenkede data og den semantiske veven

Dette kapittelet starter med å forklare hva som menes med åpne, offentlige data og deres samfunnsmessige verdi. Videre beskrives den politiske situasjonen og inititativer knyttet til åpne data i Norge og utlandet.

Kapittelet fortsettes med en beskrivelse av problematikken rundt frigjøring av offentlige data og hvilke hindringer som må imøtekommes i den forbindelse. Gjennom tankegangen bak den semantiske veven stilles et sett med redskaper velegnet til arbeid med åpne data til rådighet. Kapittelet avsluttes med en beskrivelse av dette: lenkede data og semantisk vev-teknologier, og hvordan slike kan anvendes til frigjøring av offentlige data.

## 2.1 Politisk bakteppe

### 2.1.1 Åpne, offentlige data

Med offentlige data menes data som er produsert av offentlige etater, institusjoner og kommuner. *Data* forstås som det laveste abstraksjonsnivået informasjon kan baseres på; som en variabel eller et sett av variablers kvalitative eller kvantitative egenskaper. En samling av data innenfor samme domene, levert av samme aktør, refereres til som et *datasett*. At dataene er *åpne*, betyr at de er tilgjengeliggjort for allmennheten og kan leses og brukes til marginalkostnad. Marginalkostnad tilsvarer innen distribusjon, kostnaden det å tilby en ekstra kopi har. Tradisjonelt sett vil dette si kostnaden for papirutskrift og postgang. I denne oppgaven forutsettes data i elektronisk

form. Marginalkostnad tilsvarer her kostnaden for dataoverførsel over nett [39, s. 4].

Det ligger store økonomiske interesser bak det å åpne, *frigjøring*, av offentlige data. Frigjøring av offentlige aktørers data åpner for bruk av dataene til andre formål enn hva som opprinnelig var tiltenkt, såkalt *viderebruk*, samt kombinering av data på tvers av ulike datasett, *sammenstilling*. Viderebruk og sammenstilling av offentlige data vil trolig bedre informasjonsflyten mellom ulike etater og bidra til stor inntjening i form av langt mer effektiv informasjonsutveksling og økt felles forståelse. Blant annet kan dobbeltarbeid unngås ved at åpne data, for eksempel adresseinformasjon, fra en etat viderebrukes av en annen etat som trenger tilgang til slikt. Dette står som alternativ til at den andre etaten må søke fram og produsere tilsvarende data om de samme adressene selv.

Økt tilgjengelighet på offentlige data vil føre til stor verdiskapning i form av innovasjon, nye tjenester og applikasjoner, som kan baseres på de frigjorte datasettene. En undersøkelse gjort på vegne av EU estimerer at innovasjon basert på viderebruk av offentlige data har en markedsverdi på hele 27 milliarder euro for EUs medlemsland + Norge [35]. Tjenester og applikasjoner bygd på åpne, offentlige data kan brukes til alt fra enkle, dagligdagse ting, slik som å informere om hvor nærmeste ladestasjon for elbil finnes tilgjengelig, til problemstillinger av større samfunnsmessig betydning, for eksempel automatiserte snøskredsvarsler.

Sammenstilling av ulike offentlige datasett kan lede til nye kombinasjoner av data. Data som satt sammen kan avsløre spennende sammenhenger og ny kunnskap som kan være vesentlig for å overvåke samspillet mellom miljø og helse, og i enkelte tilfeller bidra til å redde liv. Et gammelt eksempel på dette er doktor John Snows oppdagelse av sammenhengen mellom vannforurensning og kolera på midten av 1800-tallet. Snow kombinerte data om steder der det hadde inntruffet koleraforårsakede dødsfall med data om vannpumpers plassering i London. Gjennom denne sammenstillingen oppdaget han økt forekomst av koleradødsfall i nærheten av vannpumper hvis innhold var kloakkholdig. Denne informasjonen bidro til at det ble bygd kloakksystemer, som igjen førte til en bedring av befolkningens helse. Antallet mulige sammenstillinger av data og tilhørende, nye tjenester som kan skapes på bakgrunn av disse, begrenses kun av fantasien. I nyere tid kan en tenke seg andre kombinasjoner av datasett som resulterer i informasjon med nytteverdi for miljø og helse, for eksempel en sammenstilling av radiomasters plassering med kartdata brukt for å illustrere strålingsfare i ulike nabolag.

Det er ikke bare økonomiske årsaker som gjør åpne, offentlige data interessante. Frigjøring- og mulighet for sammenstilling av slike data skaper

offentlig transparens som igjen kan føre til en styrkning av demokratiet. Privatpersoner, frivillige organisasjoner og medier får innblikk i hvilke data som er tilgjengelige og dermed mulighet til å kontrollere den utøvende makt i større grad. Et eksempel på dette er frigjøring av partifinansieringsdata. Disse beskriver hvor mye pengestøtte hvert politisk parti har mottatt i hver av Norges kommuner; spennende data i seg selv som også kan viderebrukes og sammenstilles med andre data for å danne ny informasjon. Mediene blir også mer transparente når datagrunnlaget som benyttes i en sak finnes offentlig tilgjengelig. Slik kan borgerne kontrollere at dataene saken bygger på faktisk stemmer.

De ovennevnte effektene ved frigjøring av offentlige data kan oppsummeres slik:

- effektivisere offentlig sektor
- skape verdier gjennom nye applikasjoner og tjenester
- belyse ny, viktig kunnskap og sammenhenger
- føre til offentlig transparens

Dette er fordeler som har blitt tydelige for mange de siste årene. I kjølvannet av finanskrisen har spesielt det store potensialet for økonomisk vekst blitt vektlagt. Dette har ført til at åpne data har blitt satt stadig høyere på den politiske agendaen i en rekke land de siste årene.

### 2.1.2 Initiativer i utlandet

EUs direktiv om viderebruk av den offentlige sektors informasjon (direktiv 2003/98/EFa), i det følgende kalt viderebruksdirektivet, ble vedtatt 17. november 2003. Direktivet skal etter sigende bidra til å fremme økt viderebruk av offentlig informasjon, særlig digital. Dette "bør bl.a give de europæiske virksomheder mulighed for at udnytte disses potentiale og bidrage til økononomisk vækst og jobskabelse". Direktivets formål er å sikre en harmonisert praksis av EU-landenes regler om viderebruk av offentlig informasjon [2, s. 35]. Jamfør viderebruksdirektivets artikkel 12, settes implementeringsfristen for EUs medlemsland til 1. juli 2005.

Storbritannia har gått et stort skritt i retning av å oppfylle viderebruksdirektivets mål om økt viderebruk av offentlig informasjon gjennom handlingsplanen "Putting the Frontline First: Smarter Government". Denne handlingsplanen ble publisert i desember 2009 og inneholder en rekke prinsipper for hvordan offentlige data skal håndteres. Blant annet skal offentlige data publiseres i

viderebrukbar, maskinlesbar form gjennom åpne standarder og etter anbefalinger fra World Wide Web Consortium (W3C)[1]. Handlingsplanen beskriver videre at rådata skal representeres som lenkede data og tilgjengeliggjøres på nettstedet www.data.gov.uk. Daværende statsminister, Gordon Brown, utnevnte direktøren for W3C, Tim Berners-Lee, til frontfigur for satsningen [19, s. 7].

Det er ikke kun i Europa frigjøring av offentlige data har blitt satt på dagsordenen. Data.gov ble lansert i mai 2009 og er en offisiell nettside levert av de amerikanske myndighetene. Nettsiden inneholder datasett bygd på rådata fra offentlige virksomheter, verktøy/applikasjoner som benytter seg av dataene, samt metadata som beskriver datasettene: hvilken etat/aktør som står bak dem, siste oppdatering, oppdateringsfrekvens, hva slags data datasettet inneholder, tilgjengelige formater for nedlastning etc. Hvert datasett har en rangeringsmulighet hvor brukerne kan gi dem poeng basert på ulike kriterier, for eksempel tilgjengelighet og hvor nyttige de er. Data.gov er i stadig vekst ved at nye datasett og applikasjoner sluttes til av brukerne. Målet med nettsiden, i følge Data.gov selv, er "å bedre tilgangen til offentlige data og å fremme kreativ bruk av disse utenfor myndighetenes vegger i form av for eksempel nettapplikasjoner. ... Åpenheten som skapes av Data.gov vil styrke nasjonens demokrati og fremme myndighetenes effektivitet ." [41]

Den amerikanske regjeringen fulgte opp lanseringen av Data.gov i desember 2009 da de innførte et direktiv som instruerer offentlige virksomheter om å oppfylle flere krav til transparens. Et av disse kravene var at hver virksomhet skulle publisere minst tre datasett av "høy verdi" i åpent format på nett innen 45 dager. Datasettene skulle registreres på Data.gov og kunne ikke ha vært publisert på nett tidligere. Hva som menes med "høy verdi" er definert nærmere i direktivet [19, s. 8]. I den grad disse målene nås, vil Data.gov være en tungtveiende aktør når det gjelder å sikre viderebruk av data i USA. Dette vil forhåpentligvis lede til verdiskapning i form av utvidet tjenesteproduksjon og styrking av demokratiet.

Frigjøring av data kan også skje på mindre skala enn den nasjonale. Høsten 2008 ble konkurransen "Apps for Democracy" avholdt i Washington D.C. (District of Columbia), USA. Her ble det konkurrert i å utvikle applikasjoner som på spennende måter nyttiggjør seg av åpne data tilgjengeliggjort gjennom D.C.s datakatalog[2] — en samling av datasett tilgjengelige for nedlastning. Konkurransen resulterte i 47 mobil- eller vevapplikasjoner som ble premiert med penger og anerkjennelse. Washington D.C.s kostnad i forbindelse med å avholde konkurransen var på 50 000 $, mens det er estimert at den samlede verdien på applikasjonene konkurransen resulterte i, svarer

[1] Se avsnitt 2.3.1 på side 17 for mer informasjon om W3C.
[2] http://data.octo.dc.gov/

til hele 2 300 000 $ [11].

Et av bidragene i "Apps for Democracy" var nettapplikasjonen "DC Bikes"[3]. Denne applikasjonen sikter mot å gjøre Washington D.C. til en mer sykkelvennlig — og dermed også mer miljøvennlig — by. DC Bikes viser informasjon om sykkelruter, sykkelbutikker, t-banestasjoner og sykkeltyverier på kart, og om sykler som ønskes kjøpt eller solgt i D.C.

Et annet konkurransebidrag var nettapplikasjonen iLive.at. Denne viser informasjon om nærområdet til en valgt adresse i Washington D.C.: demografi, servicetilbud, kriminalstatistikk, rekreasjonsmuligheter etc. Dataene presenteres, avhengig av type, som statistikk i diagrammer, i lister sortert på avstand og som punkter på et kart. Applikasjonen søker å forenkle hverdagen til folk som bor i Washington D.C., ved å tilgjengeliggjøre tilbud og informasjon de kanskje ikke var klar over at fantes fra før, og å hjelpe mennesker som skal flytte til/innad i byen å velge et område som tilfredstiller deres ønsker og behov.

"Apps for Democracy" illustrerer det enorme potensialet for verdiskapning og tjenesteproduksjon som frigjøring av offentlige data kan skape. Konkurransen resulterte i flere mye brukte applikasjoner som gavner lokalsamfunnet i D.C. på ulike måter og har inspirert initiativer til lignende tiltak rundt om i verden.

### 2.1.3 Norske initiativer

Den 19. juni 2003 ble en interdepartemental arbeidsgruppe opprettet av den norske regjeringen. Målet for gruppens arbeid var å utrede prinsipper for kommersiell utnyttelse (tilgang til og prising) av offentlig informasjon og å finne ut hvordan viderebruksdirektivet kunne innlemmes i norsk lov [2, s. 7]. Rapporten "Fra bruk til gjenbruk" ble utgitt 30. august 2004 som et resultat av dette arbeidet. Denne rapporten gir anbefalinger til hvilke endringer som må på plass for å innpasse viderebruksdirektivet i norsk lov.

Etter anbefalingene fra "Fra bruk til gjenbruk" og flere andre utredninger ble EUs viderebruksdirektiv implementert i endringen av LOV 2006-05-19 nr. 16: "Lov om rett til innsyn i dokument i offentleg verksemd (offentleglova)" som trådte i kraft 1. januar 2009 [19, s. 5]. Offentlighetsloven lovfester prinsipper som skal skape transparens og åpenhet i offentlig virksomhet og tilrettelegge for viderebruk av offentlig informasjon i Norge. Første paragraf i loven lyder slik: "Formålet med lova er å leggje til rette for at offentleg verksemd er open og gjennomsiktig, for slik å styrkje informasjons- og

[3] http://outsideindc.com/bikes

ytringsfridommen, den demokratiske deltakinga, rettstryggleiken for den enkelte, tilliten til det offentlege og kontrollen frå ålmenta. Lova skal òg leggje til rette for vidarebruk av offentleg informasjon." Videre forbyr reglene om viderebruk i loven diskriminering mellom ulike aktører, lovfester forbud mot enerettsavtaler, innfører rett til å kreve kopi av dokumenter i alle eksisterende formater og språkversjoner og rett til å kreve elektronisk kopi [14, s. 100]. En viktig paragraf med tanke på frigjøring av data er lovens paragraf 9: "Alle kan krevje innsyn i ei samanstilling av opplysningar som er elektronisk lagra i databasane til organet dersom samanstillinga kan gjerast med enkle framgangsmåtar."

Rapporten eNorge 2009 ble levert av Moderniseringsdepartementet i juni 2005, og handler om hvordan regjeringen ønsker å utnytte og realisere mulighetene informasjonsteknologi skaper. I rapporten er det utformet en rekke ønsker for det digitale Norge. Disse inkluderer blant annet: "sikre god og rimelig tilgang til offentlige data som kan benyttes til å skape verdiøkende tjenester" [28, s. 16], "bedre digital samhandling mellom offentlige virksomheter på tvers av forvaltningsnivåer" [28, s. 24] og det konkrete målet om at "all ikke-sensitiv, formell kommunikasjon mellom offentlige virksomheter i hovedsak skal skje elektronisk..." [28, s. 27]

I stortingsmelding nr. 17 (2006-2007) — "Eit informasjonssamfunn for alle" følges ønskene fra tidligere rapporter opp med tiltak 6.6 om viderebruk av offentlig informasjon: "Regjeringa vil føre ein heilskapleg politikk som sikrar effektiv vidarebruk av offentleg informasjon for auka verdiskaping og utvikling av nye tenester. For å leggje til rette for vidarebruk av offentlig informasjon, skal hovudregelen om at innsyn skal vere gratis oppretthaldast, og betaling skal berre takast i særlege tilhøve og då med eit kostnadsbasert prinsipp som hovudregel" [14, s. 104].

Det legges altså tydelige føringer fra høyt politisk hold for å sikre bedre digital samhandling og økt viderebruk av informasjon i det offentlige Norge. Ting har derfor sakte, men sikkert begynt å skje på denne fronten. Torsdag 15. april 2010 ble et nettsted på domenet data.norge.no opprettet. Nettstedet var opprinnelig en blogg driftet og administrert av Fornyings-, administrasjons- og kirkedepartementet (FAD). Det første blogginnlegget, skrevet av fornyingsminister Rigmor Aasrud, beskrev tanken bak siden slik: "...data.norge.no skal være et nettsted som samler informasjon om offentlige data. ...Målsetningen er at dette nettstedet skal være en møteplass for alle som arbeider med viderebruk av offentlige data". Aasrud etterspurte videre hjelp til å utvikle nettstedet i form av at brukerne ble bedt om å delta i debatt rundt hvordan nettstedet skal utformes, hvilke offentlige data som ønskes tilgjengeliggjort og hvilke måter offentlige data skal tilgjengeliggjøres på. Tilbakemeldingene dette resulterte i ble brukt for å

utforme kravspesifikasjonen FAD lagde før de la utvikling av siden ut på anbud [15]. data.norge.no skal altså bli en portal lignende data.gov(.uk) hvor man kan få tilgang til datasett fra offentlige, norske virksomheter og en møteplass hvor erfaringer og meninger om åpne data utveksles.

Fornyings-, administrasjons- og kirkedepartementet vedtok fem fellesføringer for 2011 som departementene skal videreformidle til de underliggende etatene [17]. "Ei fellesføring er eit pålegg til alle departement og etatar om ein aktivitet på eit særskilt område. Ei fellesføring skal vere like viktig å gjennomføre for verksemdene som eit pålegg frå ein statsråd til ei underliggjande verksemd" [16]. Én av de fem fellesføringene for 2011 dreier seg om tilgjengeliggjøring av offentlige data: "Etatene skal gjøre egnede og eksisterende rådata tilgjengelige i maskinlesbare formater. Dette gjelder informasjon som har samfunnsmessig verdi, som kan viderebrukes, som ikke er taushetsbelagte og der kostnadene ved tilgjengeliggjøring antas å være beskjedne (bortfall av inntekter ved salg av data anses som en kostnad). Formater og bruksvilkår må være i overensstemmelse med Referansekatalogen og FADs føringer på nettstedet data.norge.no..." [17]. Alle etater og direktorater er dermed pliktige å tilgjengeliggjøre offentlige data fra januar 2011 etter prinsippene FAD fører gjennom data.norge.no. Fellesføringene skal følges opp innenfor budsjettrammene for 2011 [17]. Mange aktører slipper trolig unna denne fellesføringen som følge av dette, ikke minst siden den kun gjelder frigjøring av informasjon der kostnaden for tilgjengeliggjøring antas å være beskjeden samtidig som bortfall av inntekter ved salg av data anses som en kostnad.

## 2.2 Frigjøring av data

### 2.2.1 Introduksjon

Vi har sett eksempler på stor politisk vilje rettet mot frigjøring av offentlige data i en rekke land, inkludert Norge. Flere rapporter levert av eller på oppdrag for myndighetene har forsøkt å fremme økt transparens i form av viderebruk og frigjøring av offentlige data i Norge. Datafrigjøring er også pålagt gjennom fellesføringene for 2011. Hvordan er så statusen hos det offentlige i dette øyemed? I hvor stor grad har åpne data blitt en realitet?

Data finnes overalt og i "uendelige" former; alt fra personlige data og data på sosiale nettsteder om hvem man anser som sine venner til statlig statistikk og næringslivsdata. Problemet er at veldig mye data ikke er tilgjengeliggjort for bruk av andre en dataeieren selv. Bedrifter er redde for å frigi data i frykt for å gi uønsket hjelp til konkurrerende bedrifter, mens offentlige enheter kan

vegre seg mot å åpne data i frykt for at de skal inneholde sensitiv informasjon. Åpne data er slik sett kanskje et mer politisk komplisert tema enn hva det er teknisk sett. Alt handler om å få dataeierne til å åpne dataene sine.

Et knippe norske offentlige virksomheter har frigjort deler av sine data. Sommeren 2009 lanserte Avinor en tjeneste på nett som tilbyr flydata, statustekster, flyplassnavn og flyselskap i XML-format. Disse dataene er åpne for viderebruk så lenge innholdet ikke benyttes i sammenhenger som bryter norsk lov [3]. 1. desember samme år åpnet Statens kartverk for viderebruk av sine visningstjenester, men ikke de underliggende kartdataene [19, s. 6] som man trenger for å for eksempel kunne vite om en vei er kommunal eller en fylkesvei. Frigjøringen var, til tross for at den var begrenset, et steg i "riktig" retning. Kartverkeksempelet er viktig fordi kart og kartdata står veldig sentralt som datatyper mange vil ønske å viderebruke. Slike data kan sammenstilles med en rekke andre typer data. Dette er svært relevant for blant annet applikasjonen Semantikart, som er en del av denne masteroppgaven. Semantikart visualiserer ulike åpne, offentlige data som punkter på et kart. For eksempel knyttes data om hvor det finnes parkeringshus sammen med en kartposisjon for å vise hvor det finnes parkeringsmuligheter i applikasjonsbrukerens nærområde.

### 2.2.2 Holdninger til datafrigjøring

Allmennheten skal ha tilgang til alle offentlige data og dokumenter i henhold til offentlighetslovens paragraf 9. Et viktig unntak finnes imidlertid: sensitive personopplysninger. Slike opplysninger er underlagt taushetsplikt i henhold til Personopplysningsloven [40, s. 13].

Institutt for informasjons- og medievitenskap ved Universitetet i Bergen adresserte frigjøring av offentlige data i Norge gjennom et kartleggingsprosjekt med utgangspunkt i spørsmålene: "Hvilke datakilder forvalter offentlige virksomheter, og hva hindrer at mer av disse dataene gjøres tilgjengelig for viderebruk?" [19, s. 10]. Kartleggingsprosjektet fant sted i perioden august—desember 2009 og dannet grunnlag for rapporten "Fakta først" som ble publisert i januar 2010.

"Fakta først" oppsummeres med at en stor andel av de undersøkte offentlige virksomhetene har mangelfull eller fravær av informasjon om datakilder på sine nettsider. For disse svikter dermed den grunnleggende forutsetningen for viderebruk av offentlige data: tilgjengeliggjøring og informering. To tredjedeler av de undersøkte virksomhetene sier at deres virksomhet har data med potensial for viderebruk som ikke blir utnyttet [19, s. 3]. Ifølge undersøkelsen er økte kostnader og faren for at eksterne aktører kan misforstå

dataene og spre villedende informasjon og sensitive personopplysninger de største hindrene for at mer data blir frigitt [19, s. 20].

EU-kommisjonen har kommet med en evaluering av innføringen av viderebruksdirektivet. I denne evalueringen vises det blant annet til at offentlige organer har manglende forståelse for viderebrukspotensialet i sine egne data [40, s. 23]. For at frigjøring av offentlige data skal bli en realitet, må mulighetene slik frigjøring åpner for belyses. I Norge har FAD tatt konsekvensene av dette og lyst ut midler både til et oppdrag om å lage "en lettfattelig og engasjerende videosnutt som forklarer konseptet viderebruk av offentlige data" og en "innbydende, lesevennlig, pedagogisk og lettfattelig" veileder om tilgjengeliggjøring av offentlige data [15]. EU-kommisjonens evaluering viste videre at markedet har mangelfull informasjon om hvilke offentlige datasett som er gjort tilgjengelige. Dette vil også trolig bedre seg som følge av økt fokus på emnet.

"Fakta først" og EU-kommisjonens evaluering viser at mange av problemene med frigjøring av offentlige data ser ut til å dreie seg om holdninger hos dataeierne og manglende forståelse for mulighetene viderebruk av offentlige data gir. Siden datafrigjøring settes i søkelyset i stadig større grad, blant annet gjennom Fornyings-, administrasjons- og kirkedepartementet og data.norge.no, vil disse problemene forhåpentligvis bli mindre med tiden. Ved at det opplyses om hvilke datasett som finnes tilgjengelige i Norge på data.norge.no vil markedet få øynene opp for de åpne datasettene, mens informasjon og føringer fra politisk hold kan bidra til holdningsendringer hos de offentlige dataeierne.

### 2.2.3 Hvordan tilgjengeliggjøre offentlige data?

På Oslo Brannvesens nettsider finnes en underside, Statistikk[4], som inneholder data om hvilke typer utrykninger brannvesenet har hatt på årlig basis. Disse dataene er tilgjengeliggjort gjennom et Excel-ark. Dataene beskriver hva slags type utrykninger som er gjort og hvor mange av hver type som har funnet sted for hver av månedene. Dataene er altså aggregert på måned og inneholder ikke mer informasjon om hver enkelt utrykning utover dette.

Oslo Brannvesen har i dette datasettet på forhånd gjort antakelser for hvordan den tilgjengeliggjorte informasjonen skal brukes. Når dataene er aggregert på måned, begrenser dette måtene dataene kan presenteres videre på. Statistikksiden inneholder flere bildefigurer som viser kakediagrammer for hvordan spredningen var blant de ulike typene av utrykninger i 2008 —

[4] http://www.brann-og-redningsetaten.oslo.kommune.no/statistikk/

enda grovere granularitet enn i Excel-fila. Dette var tydelig den tiltenkte måten dataene skulle presenteres på fra brannvesenets side: aggregert for hele året. Med finere granularitet der utrykninger istedenfor var samlet på dager, kunne disse dataene vært kombinert med blant annet værdata. En slik sammenstilling kunne vist hvordan typen utrykninger per dag varierte med værtypen. For eksempel kunne en tenke seg økt forekomst av vannskadeutrykninger på dager der mildvær etterfulgte kuldeperioder.

Dette eksempelet illusterer et viktig poeng ved datafrigjøring: åpne data bør ikke aggregeres, med tanke på en spesiell anvendelse, når dette fører til at annen offentlig informasjon, som kunne vært anvendt av andre, ikke blir tilgjengeliggjort.

NRKs dokumentarprogram "Brennpunkt" lagde i 2008 et innslag som omhandlet hvordan hyppigheten av branner fordelte seg blant verneverdige bygg i forhold til andre bygg i Oslo. Dette fantes det ikke noen direkte statistikk på fra før. Brennpunkt måtte derfor sammenstille informasjon fra to datakilder: Byantikvarens offentlige, gule liste over verneverdige bygninger og brannvesenets liste over branner i Oslo. Hadde disse datakildene delt et felles vokabular og vært tilgjengeliggjort i åpent format på nett, ville det krevd lite arbeid å finne ut av. En enkel spørring som sammenlignet adresseidentifikatorene fra begge kildene kunne raskt levert data om dette. Så enkelt var det imidlertid ikke i virkeligheten. Byantikvarens gule liste har vært tilgjengeliggjort på Oslo kommunes hjemmesider som nedlastbart Excel-ark fra år 2005 [8]. Oslo Brannvesen frigjør foreløpig ikke andre data enn antall forskjellige utrykninger per måned og måtte derfor kontaktes for å gi tilgang til dataene som beskrev hvilke adresser det hadde vært brann på. De nødvendig dataene var altså ikke åpne. De to datakildene hadde dessuten lagret adresseinformasjon på litt ulike måter og brannvesenets data inneholdt flere stavefeil på adressenavn. Brennpunkt måtte derfor lage egne koblinger mellom adressene i de to datakildene og manuelt kontrollere at koblingene var korrekte; ressurs- og tidkrevende arbeid.

Brennpunkteksempelet illustrerer en av mange muligheter for produksjon av nye, interessante data gjennom viderebruk og sammenstilling av offentlige data. I dette eksempelet fantes ett av to nødvendige datasett åpent tilgjengelig. Dersom begge datasettene var åpne, kunne mye tid vært spart på å sammenstille dem. I tillegg til behovet for å øke andelen åpne offentlige data, ser vi også et behov for en enighet om et felles utvekslingsformat som datafrigjøringen kan skje på.

### Utvekslingsformat

Verdensveven, heretter kalt veven, har revolusjonert måten vi deler kunnskap på ved å minske terskelen for å publisere dokumenter og å øke vår tilgang til dem [9, s. 1]. På veven knyttes dokumenter til andre dokumenter ved hjelp av hyperlenker. Søkemotorer indekserer dokumentene og lar oss stille dem enkle spørringer som tilbyr oss tilgang til potensielt relevante treff. Til tross for at disse prinsippene lenge har vært gjeldende for dokumenter, har ikke det samme vært tilfelle for dataene som er frigjort på veven.

Vi kan anta at dataene i første omgang ikke ble lenket opp på samme måte som dokumenter på nettet fordi fokus ble lagt på presentasjonen av (layout, interaksjon og struktur) istedenfor rådataene selv [21, s. 3]. Det finnes enorme mengder offentlige data. Disse dataene er spredd rundt blant tusenvis av forskjellige dataeiere: departementer, etater, offentlige organisasjoner, direktorater etc. Dataene er lagret på forskjellige formater, både fra virksomhet til virksomhet og lokalt mellom systemer innad hos enkelte virksomheter. Dataene som er publisert, finnes tradisjonelt på XML-format, som komma-separerte verdier, innpakket i dokumentfiler, eller i HTML-tabeller. Formatet offentlige data tilgjengeliggjøres på, er viktig fordi det avgjør hvordan disse dataene kan viderebrukes.

Flere dataformater begrenses i sin bruk ved at de er proprietære og eventuelt lukkede. En fil på et *lukket format* krever en spesifikk programvare for å kunne åpnes — programvare som i visse tilfeller kan koste penger. Tilgjengeliggjøring av data bør ikke skje på slike formater siden dette ikke gjør dem tilgjengelige for alle. Data som er frigjort på et lukket format, er ikke åpne data.

*Proprietære formater* er kontrollert og eid av en privatperson eller organisasjon. Slike formater er gjerne beskyttet av patenter hvor eieren beholder eksklusiv kontroll over teknologien. Det unike eierskapet et proprietært format har, bidrar til å øke risikoen for at formatet i framtiden kan bli avleggs. Formatets levedyktighet avhenger av firmaet som eier det sin levedyktighet, og at det ikke plutselig bestemmes at formatet ikke lenger skal satses på. Proprietære formater, selv når de er åpne, bør av denne grunn unngås som utvekslingsformat for åpne data. Åpne, ikke-proprietære formater kan fritt implementeres av hvem som måtte ønske det og har derfor mindre risiko for å bli avleggs.

Dataelementene som finnes i tekstdokumenter er "kamuflert" blant teksten den står sammen med. En maskin er ikke i stand til å hente ut delene av teksten som er viktig for en gitt kontekst uten videre, for eksempel navnene på møtedeltakerne sammen med informasjon om møtetidspunkt og møtested trukket ut fra et dokument som oppsummerer alle møtene en

bedrift har avholdt i løpet av et år. *Maskinlesbare formater* er formater der informasjonen er strukturert på en måte som er tolkbar for maskiner. XML er et eksempel på et slikt format hvor ulike merker (tags) representerer data av en gitt type. Tabulære data er et annet eksempel hvor en gitt kolonne beskriver data med en bestemt egenskap. At åpne data frigjøres på et åpent, maskinlesbart format er et krav for at dataene skal kunne viderebrukes effektivt i de fleste sammenhenger.

### Felles identifikatorer

De ulike maskinlesbare formatene som brukes, varierer fra datasett til datasett. Sammenstilling av ulike datasett øker i kompleksitet når datasettene er lagret på ulike formater. Data i forskjellige datasett, lagret på samme format, inneholder gjerne semantiske heterogeniteter på representasjonsnivå datasettene imellom. Enten i form av navnekollisjoner: samme navn brukes om forskjellige ting, eller ved at ulike identifikatorer og representasjonsformer brukes om de samme tingene i forskjellige datasett.

For å sikre effektiv sammenstilling av data fra heterogene kilder er det et behov for felles etablerte identifikatorer.

### Åpne data — hva er det?

Open Definition har på sin nettside[5] listet opp et sett med kriterier som verk (data) og samlinger (datasett) må oppfylle for å kunne sies å være åpne. Mange datasett frigjøres under en lisens. Med begrepet lisens, slik det beskrives på nettsiden, menes juridiske vilkår knyttet til tilgjengeliggjøringen av data. Blant kravene til Open Definition finner vi blant annet at "verket skal være tilgjengelig i sin helhet og helst kunne lastes ned kostnadsfritt" og at det "skal tilbys i en slik form at det ikke finnes noen teknologiske hindringer". Videre skal eventuelle lisenser ikke diskriminere noen personer, grupper, fagfelter eller bruksområder, mens modifiseringer og avledninger av verk og distribusjon av slike skal tillates under samme vilkår som gjaldt det opprinnelige verket.

Oppsummert vil åpne data si, slik beskrevet innledningsvis i kapittelet, data som er tilgjengeliggjort for allmennheten og kan leses og viderebrukes uten begrensninger til marginalkostnad. Dessuten må dataene være tilgjengeliggjort på et åpent, ikke-proprietært og maskinlesbart utvekslingsformat slik beskrevet i avsnittet "Utvekslingsformat", 2.2.3 på forrige side.

[5] http://www.opendefinition.org/okd/norsk_bokmaal/

## 2.3 Den semantiske veven og lenkede data

> "I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web — the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize."
>
> — *Tim Berners-Lee*

### 2.3.1 Introduksjon

*Den semantiske veven* er en term som først ble benyttet av direktør for *World Wide Web Consortium* (W3C), Tim Berners-Lee. I følge Berners-Lee dreier det hele seg om lenker skapt på en måte som personer og maskiner kan benytte seg av for å utforske en vev av data [6], den semantiske veven.

W3C er en organisasjon som arbeider med å utarbeide standarder til bruk på verdensveven (www). World Wide Web Consortium har også skapt flere standarder til bruk på, og er en sentral aktør innen den semantiske veven. W3C visualiserer den semantiske veven som: "et desentralisert, verdensomspennende informasjonsrom for deling av maskinlesbare data til et minimum av integrasjonskostnader" [34].

W3Cs idéer for en slik vev danner utgangspunkt for min oppgave. Gjennom *lenkede data*, en term som av Berners-Lee er betegnet som "den semantiske veven gjort på rett måte" [25], beskrives retningslinjer for hvordan data kan frigjøres. Ved å følge disse retningslinjene adresseres en del av de ovennevnte teknologiske og strukturelle utfordringene som følger med datafrigjøring. Følgelig baseres min oppgave på data som er tilgjengeliggjort på W3C-måten, som "lenkede data".

### 2.3.2 Lenkede data

Lenkede data baseres på standard vevteknologier, som URI-er og HTTP. *HTTP* (HyperText Transfer Protocol), standardprotokollen for utveksling over veven, forutsettes i denne rapporten som kjent, mens *URI* står for Uniform Resource Identifier og er en enkel og utvidbar måte å identifisere en

ressurs på [5]. Termen ressurs[6] brukes i det følgende om alle identifiserbare ting, inkludert forhold mellom slike. På verdensveven brukes HTTP-URI-er for å kombinere globalt unik navngivning med en enkel, velprøvd måte å sikre tilgang til ressurser på. Ved å bruke HTTP-URIer identifisereres ressurser med en URI, som gjennom oppslag via HTTP-protokollen også kan resulterere i en beskrivelse av ressursen [22, s. 8].

Innen lenkede data supplementeres URI-er og HTTP med *Resource Description Framework (RDF)*, et formelt språk som brukes for å beskrive strukturert informasjon, data. Målene med RDF er å la applikasjoner kunne utveksle data på verdensveven og å åpne for at de kan kombineres med andre data og viderebrukes av andre enn dem de opprinnelig ble skapt for. Dette gjør at data kan sammenstilles og spørres mot, blant annet gjennom RDF-spørrespråket SPARQL, automatisk av maskiner. RDF anses derfor som den grunnleggende representasjonsformen på den semantiske veven [24, s. 19].

Ved hjelp av HTTP-URI-er og RDF utvides verdensveven fra en vev der informasjon er tilpasset lettleselighet for mennesker, i form av HTML-sider, til en semantisk vev der data i tillegg beskrives i maskinlesbar form.

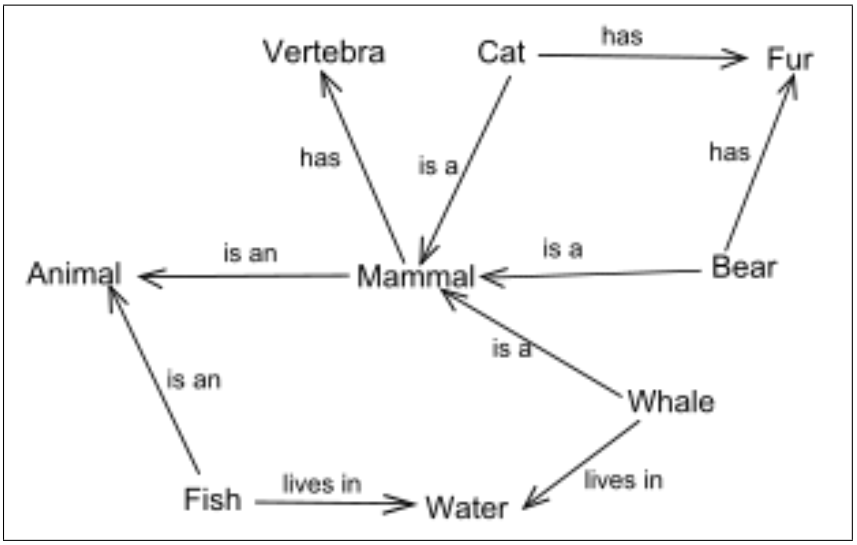
Retningslinjene som termen *lenkede data* refererer til, har opphav i Tim Berners-Lees designnotat, "Linked Data", publisert i 2006. Fire retningslinjer beskrives i notatet [6], som utgjør en oppskrift for hvordan man bør publisere og koble sammen data på veven [9, s. 2]:

1. Bruk URI-er som navn på ting
2. Bruk HTTP-URI-er slik at man kan gjøre oppslag på det som identifiseres gjennom URI-ene (navnene)
3. Når noen slår opp i en URI, tilby nyttig informasjon gjennom standardene (RDF, SPARQL)
4. Inkludér lenker til andre URI-er, slik at flere ting kan oppdages

Teknisk sett, kan vi forstå lenkede data som data publisert på veven på en måte som gjør dem maskinlesbare, med eksplisitt uttrykt betydning og med lenker til og fra andre eksterne datasett [9, s. 2]. Betydningen ekspliseres ved hjelp av såkalte *typede lenker*: lenker som inneholder informasjon om forholdet mellom ressursene lenken knytter sammen [9, s. 1]. Ny, verdifull informasjon kan oppdages når to eksisterende datasett kobles sammen gjennom slike lenker.

En liknende måte å koble sammen data på finner vi igjen i semantiske nettverk. Et semantisk nettverk er en måte å representere kunnskap på

[6] Mer om ressurser i semantisk vev-kontekst i 3.1.1 på side 24.




Figur 2.1: Eksempel på et semantisk nettverk

[36] og en velprøvd datastruktur blant annet innenfor kunstig intelligens. Figur 2.1 illustrerer et semantisk nettverk i form av en graf, hvor datarelasjonene — gitt av kantene i grafen, kan sammenlignes med typede lenker.

### Hvorfor frigjøre offentlige data som lenkede data?

Retningslinjene i Berners-Lees notat adresserer utfordringene som oppstår i forbindelse med publisering av åpne data på veven. Teknologiene som nevnes i notatet: URI-er og W3C-standardene RDF og SPARQL, er alle sentrale innenfor lenkede data og den semantiske veven. Disse danner en verktøykasse med redskaper, semantisk vev-teknologier, som av flere grunner er egnet for publisering av og arbeid med åpne data.

De nevnte teknologiene er alle åpne og ikke-proprietære. Ved frigjøring av data gjennom semantisk vev-teknologier oppfylles dermed kriteriene til åpenhet stilt i 2.2.3 på side 16.

Oppslag kan gjøres på alle ressurser representert av HTTP-URI-er. Slike oppslag gir mer informasjon om ressursene, noe som samsvarer med retningslinje nummer to i Berners-Lees notat. På den semantiske veven er det ingen prinsipiell forskjell mellom data og dokumenter i den forstand at begge er tilgjengelige gjennom samme grensesnitt: HTTP. En URI kan brukes som et navn, samtidig som den leder, gjennom HTTP-oppslag, til en

side som dokumenterer navnet. Slike oppslag kan gi mer informasjon om hva dataene beskriver og vise betyningen til det URI-en refererer til. Et HTTP-oppslag kan vise et vevdokument, presentert på en måte som er lettleselig for mennesker og som inneholder mer informasjon om dataobjektet, når det gjøres i en nettleser, eller returnere en maskinlesbar RDF-representasjon av dataobjektet og dets tilknyttede (gjennom typede lenker) data. Med URI-er brukt til navngivning av lenkede data, inkludert selve lenkene, sikres en utvetydig måte å referere til vilkårlige entiteter på.

Lenkede data kan kombineres med andre lenkede data når de følger samme standard, RDF. RDF, med URI-er som referansemekanisme, sikrer derfor muligheten for sammenstilling av data fra forskjellige datasett. Dette kan gjøres automatisk av maskiner. Sammenknytting av data fra forskjellige kilder i RDF svarer til den siste av Berners-Lees retningslinjer. Informasjon fra ulike datasett som er knyttet til og relevant for hverandre viderebruker de samme URI-ene, mens typede lenker brukes for å beskrive hva slags relasjon data fra et datasett har til data i et annet. Slik kobles dataene ved hjelp av RDF sammen til en vev av data.

Verdien på informasjonen i et datasett øker når den kobles sammen med informasjon fra et annet datasett og slik settes inn i en kontekst. Viderebruk av data gjør URI-ene som viderebrukes mer synlige og forsterker på denne måten, såfremt de brukes "riktig", deres etablerte betydning. Når betydningen av en URI er etablert og kjent, viderebrukes den gjerne av andre. På denne måten skapes praksiser for navngivning av ofte brukte ressurser. URI-ene blir slik etablerte identifikatorer, noe som sikrer mulighet for effektiv sammenstilling av distribuerte datasett.

For å oppsummere, er semantisk vev-teknologier egnet til frigjøring av og arbeid med åpne data av følgende gunner:

- Utnyttelse av vevens allmenne arkitektur
- Bruk av åpne, ikke-proprietære standarder
- Maskinlesbarhet
- URI-oppslag: skaper forståelse, etablerer betydning og identifikatorer
- Etablerte identifikatorer: forenkler sammenstilling
- Støtte for typede lenker: beskriver forhold mellom data, skaper koblinger mellom datasett
- Koblinger mellom datasett: hever verdien på og synligheten til datasettene

### Anvendelser og relevante initiativer

Stadig flere initiativer knyttet til lenkede data og semantisk vev-teknologier dukker opp. Britiske data.gov.uk, som nevnt i 2.1.2, er et godt eksempel på dette. Flere andre omfattende prosjekter med store ambisjoner er underveis.

EU-prosjektet "PlanetData" er et eksempel på et slikt. I følge PlanetDatas nettside[7], er et av prosjektets hovedmål å skape løsninger til stor-skala-databehandlingsformål gjennom ulike forskningsdisipliner. Slik databehandling inkluderer RDF-data. PlanetData skal i den forbindelse levere et bibliotek med verktøy, som følger prinsippene bak lenkede data og fundamentale vevstandarder (URI-er, HTTP og RDF), egnet for publisering av data i åpne, desentraliserte miljøer [30].

Et annet EU-prosjekt "Linking Open Data 2" (LOD2), ble startet i 2010 og har en varighet ut 2014. På prosjekts hjemmeside[8] beskrives blant annet følgende mål:

- lage verktøy og metodologier klare til bruk for å administrere og eksponere veldig store mengder informasjon på veven
- etablere et testmiljø for høykvalitetsontologier fra kilder som Wikipedia og OpenStreetMap med støtte for flere anvendelser og språk
- skape algoritmer basert på maskinlæring for automatisk lenking av data på veven
- lage adaptive verktøy egnet for søk på, til å bla gjennom og til administrering av lenkede data

Den store satsningen på lenkede data i slike prosjekter har illustrert nytteverdien av Berners-Lees prinsipper anvendt til frigjøring av offentlige data. Flere og flere datasett finnes derfor tilgjengelig gjennom semantisk vev-teknologier som RDF og SPARQL; blant annet de som nevnes i 5.3.3 på side 65. Mobilapplikasjonen Semantikart baserer seg på data tilgjengeliggjort på denne måten. Semantisk vev-teknologier er derfor et svært sentralt tema for denne rapporten og gjennomgås i større detalj i neste kapittel.

### Fem-stjerners data

Berners-Lee kom med en oppdatering av sitt designnotat i 2010 [6]. I denne oppdateringen ble et fem-stjerners system, som kan brukes for å rangere

[7] http://planet-data.eu/about
[8] http://lod2.eu/Welcome.html

graden av viderebrukbarhet i åpne data, lansert:

⋆
Tilgjengeliggjort på nettet (uansett format), men med en fri lisens

⋆⋆
Som (1), men på et maskinlesbart format (f.eks. Excel-format istedenfor et bilde av en tabell)

⋆⋆⋆
Som (2), men på et ikke-proprietært format (f.eks. kommaseparerte verdier, CSV, istedenfor Excel)

⋆⋆⋆⋆
Som (3), pluss: bruk av åpne standarder (RDF og SPARQL) for å identifisere ting, slik at tingene blir lenkbare

⋆⋆⋆⋆⋆
Som (4), pluss: dataene lenkes til andres data for å skape kontekst

Data bør åpnes på en måte som svarer til flest mulige antall stjerner i dette systemet, men én-stjernes publisering er bedre enn ingen publisering. Graden av stjerneoppnåelse ved datafrigjøring bør sees i sammenheng med tilgjengelige ressurser og kunnskap hos aktøren som frigjør data. Tre-stjerners data bør være overkommelig å levere for de fleste. Slike data kan, relativt enkelt, konverteres til RDF og dermed heves til fire- eller fem-stjerners data — data som er litt mer krevende, og som det forutsettes kunnskap om semantiske teknologier for å levere.

# Kapittel 3

# Semantisk vev-teknologier

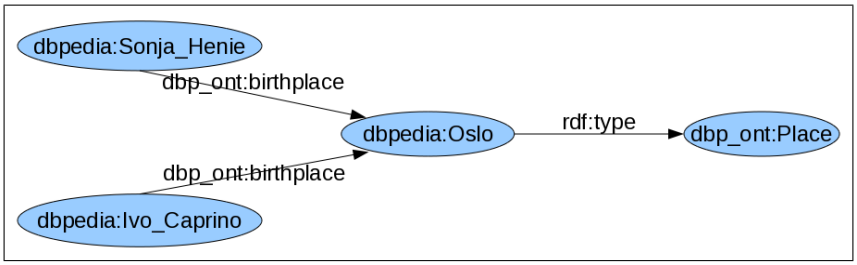
World Wide Web Consortium har de siste årene utarbeidet og publisert en rekke standarder som støtter utveksling av semantisk rik informasjon [24, s. 14]. Disse standardene, slike som RDF, OWL og SPARQL, utgjør viktige teknologier for den semantiske veven.

Dette kapittelet tar for seg det viktigste av semantisk vev-teknologier, med spesielt fokus på delene som er mest sentrale for, og som benyttes av Semantikart. Videre beskrives blant annet hvordan data kan konverteres til RDF-format og mulighetene for samspill mellom RESTfulle systemer og den semantiske veven.

## 3.1 RDF

Den semantiske veven antar en maskinlesbar måte å representere kunnskap på. RDF tilbyr dette.

RDF er en abstrakt datamodell, som beskriver relasjoner mellom data, og som kan realiseres i en av mange tekstformater, såkalte *RDF-serialiseringer*. I RDFs datamodell utgjør relasjonene en rettet graf: et sett av noder med rettede kanter mellom seg [24, s. 20]. RDF har en formell semantikk som avgjør, matematisk, hvilke slutninger en maskin kan trekke utifra en slik graf. Basert på denne semantikken kan en RDF-graf oversettes til et logisk uttrykk med samme betydning [27].




Figur 3.1: RDF-graf

### 3.1.1 RDFs grafmodell

RDF-grafer er bygd opp av en mengde data satt sammen som tripler. Et trippel representerer en relasjon i grafen og består av et subjekt, et predikat og et objekt. Subjektene og objektene utgjør nodene i grafen, mens predikatene svarer til kantene mellom dem. De samme bestanddelene i ett trippel kan inngå som bestanddeler i andre tripler og en node som har rolle som subjekt i ett trippel kan opptre som objekt i et annet. Hver kant i en RDF-graf går altså mellom et subjekt og et objekt. Antallet tripler en graf inneholder er derfor likt antallet kanter i grafen. Figur 3.1 illustrerer en RDF-graf som har tre kanter og som følgelig består av tre tripler.

Subjektet utgjør ressursen som beskrives. Ressurser representeres med URI-er og svarer til alle ting som har en tydelig identitet i en gitt kontekst: steder, mennesker, bøker, hendelser, abstrakte konsepter, forhold mellom ting etc. Predikatet utgjør en ressurs som representerer forholdet mellom subjektet og objektet. Med andre ord står subjektet i relasjonen angitt av predikatet til objektet.

Predikatene utgjør de typede lenkene mellom dataene. Slike lenker kan blant annet beskrive forholdet mellom et subjekt og objekt innenfor samme datasett, eks:

```
Subjekt:
http://dbpedia.org/resource/Folketeateret

Predikat:
http://dbpedia.org/property/architect

Objekt:
http://dbpedia.org/resource/Christian_Morgenstierne
```

Eller knytte sammen data fra forskjellige kilder, eks:

```
Subjekt:
http://opendata.computas.no/data/norad/bistand/1062

Predikat:
http://opendata.computas.no/data/norad/bistandsmottaker

Objekt:
http://www4.wiwiss.fu-berlin.de/factbook/resource/Namibia
```

Et objekt kan enten være en ressurs uttrykt med en URI, eller en såkalt literal. Literaler er konstante verdier representert av tekststrenger [27], slik som `"37"` eller `"Audrey Horne"`. Literaler kan tilknyttes en datatype, og kalles da *typede literaler*, f.eks. `"2011-01-19"^^xsd:date`, hvilket lar dens betydning tolkes uavhengig av kontekst. I eksempelet tolkes strengen som en dato. En literal som ikke har typen angitt, en *utypet literal*, kan tilknyttes et språkmerke "`@`" som angir hvilket språk tekstrengen er på: `"Oslo Opera House"@en`. Literaler kan ikke opptre som annet enn objekter i RDF [27].

### 3.1.2 RDF-serialiseringer

RDF-grafer er lette å lese for mennesker, men har en for billedlig form til å være egnet for maskinprosessering. RDF-serialiseringer representerer RDF mer konkret, tekstlig, som filer og er derfor det som brukes for å representere RDF for maskiner.

RDF/XML er en slik serialisering. I denne serialiseringen pakkes RDF-triplene inn med XML-syntaks. XML er en veletablert standard som anvendes av mange applikasjoner og som er støttet med egne biblioteker i mange programmeringsspråk. RDF/XML er valgt som den offisielle RDF-serialiseringen for å sikre at det finnes én RDF-syntaks som støttes av alle RDF-verktøy [23, s. 74]. En ulempe med RDF/XML er at den er vanskelig å tyde for mennesker siden semantikken overskygges litt av XML-syntaksen med merker, elementer og attributter. Se figur 3.2 på neste side.

XML har en datamodell med trestruktur, noe som skiller den fra RDFs grafstruktur. Trestrukturer har en hierarkisk oppbygning bestående av en rotnode med en eller flere barnenoder som igjen kan ha sine egne barnenoder osv. Denne strukturen overføres til RDF/XML hvor triplene må stå i et hierarki. Et RDF/XML-dokument innledes med en rotnode som beskriver at dokumentet inneholder RDF-data. Videre skrives subjekter under rotnoden med predikater under disse igjen og deretter objekter under predikatene. Tripler der en ressurs, som er objekt i et trippel $T$, opptrer som subjekt,

```
<?xml version="1.0"?>
<rdf:RDF xmlns:dbpedia="http://dbpedia.org/resource/"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dbp-ont="http://dbpedia.org/ontology/">
 <rdf:Description rdf:about=
                      "http://dbpedia.org/resource/Sonja_Henie">
   <dbp-ont:birthplace>
     <dbp-ont:Place rdf:about=
                           "http://dbpedia.org/resource/Oslo" />
   </dbp-ont:birthplace>
 </rdf:Description>
 <rdf:Description rdf:about=
                      "http://dbpedia.org/resource/Ivo_Caprino">
   <dbp-ont:birthplace rdf:resource=
                           "http://dbpedia.org/resource/Oslo" />
 </rdf:Description>
</rdf:RDF>
```

Figur 3.2: Triplene fra figur 1 serialisert som RDF/XML

kan enten nøstes under noden som svarer til objektet i $T$, eller settes som en ny node direkte under rotnoden. Ressurser beskrives i RDF/XML gjennom elementer av typen `rdf:Description`, hvor verdien til `rdf:about`-attributtet representerer ressursens URI. Når en ressurs forekommer i flere tripler kan hver av dem enkodes med et eget element av `rdf:Description`, noe som kan føre til at flere slike elementer refererer til det samme subjektet [24, s. 28]. En og samme RDF-graf kan dermed omformes til flere ulike RDF/XML-trær som alle beskriver de samme triplene. Det finnes med andre ord mange måter å beskrive samme RDF-graf på i RDF/XML.

Turtle (Terse RDF Triple Language) er en RDF-serialiseringsform som er et subset av Tim Berners-Lees *Notation 3* (N3) -serialisering. Turtle er mindre komplisert enn og velges derfor her framfor N3 som støtter noen avanserte utrykk utenfor fokuset til denne rapporten.

Turtle er laget spesielt for RDF og er av den grunn ikke bundet til å representere RDF-grafer som trær. Dette bidrar til at Turtle har en veldig enkel, lettleselig syntaks. I Turtle listes tripler opp i rekkefølgen: subjekt, predikat, objekt. Flere tripler adskilles med punktum.

Turtle er, til tross for at den kan sies å være mer lettleselig for mennesker, ikke like utbredt som RDF/XML [24]. Dette har trolig sammenheng med XMLs historisk overlegne verktøystøtte. Turtle har inspirert trippelsyntaksen til W3Cs SPARQL-spørrespråk (se avsnitt 3.4 på side 34) som er sentral

```
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix dbp-ont: <http://dbpedia.org/ontology/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

dbpedia:Sonja_Henie dbp-ont:birthplace dbpedia:Oslo .
dbpedia:Ivo_Caprino dbp-ont:birthplace dbpedia:Oslo .
dbpedia:Oslo        rdf:type           dbp-ont:Place .
```

Figur 3.3: Triplene fra figur 1 serialisert med Turtle-syntaks

```
dbpedia:Oslo rdf:type dbpedia-ont:Place ;
             dcterms:subject cat:Capitals_in_Europe,
                             cat:Counties_of_Norway .
```

Figur 3.4: Syntaktisk forkortning i Turtle

for denne rapporten. Dette, kombinert med dens lettleslighet gjør at Turtle velges som serialiseringsform for å illustrere RDF-tripler i det følgende.

Turtle-dokumentene kan forkortes syntaktisk ved at en ressurs som er subjekt i flere tripler kun nevnes én gang. De ulike parene av predikater og objekter knyttet til ressursen skrives da etter subjektet adskilt av semikolon <;>. Det holder også å nevne tripler bestående av samme par med subjekter og predikater kun én gang. De ulike objektene tilhørende disse parene skilles i så fall med komma <,>. Figur 3.4 illustrerer syntaktisk forkortning i Turtle.

### 3.1.3 URI-er

Ressursene identifiseres og skilles fra hverandre i RDF ved hjelp av Uniform Resource Identifiers, URI-er [27], som er en etablert praksis for navngiving.

Det er flere egenskaper ved URI-er som gjør dem attraktive som navn på RDF-ressurser. Én slik egenskap, også nevnt i 2.3.2, er at et oppslag på en URI kan, og bør etter Berners-Lees retningslinjer, åpne et dokument på veven som beskriver den URI-navngitte ressursen. Dette forutsetter at URI-en beskriver en vev-ressurs, med andre ord: URI-en er en URL. Muligheten for slike oppslag er viktig for å etablere en felles forståelse for hva URI-en identifiserer og bør skje gjennom HTTP [34].

En annen fordel ved URI-er er at de er bygd opp av domener som eieren selv kontrollerer. Dermed er det lett å knytte en ressurs til dens eier. Dette

forutsetter selfølgelig at ressursene ikke navngis med domener eieren ikke har kontroll over.

Uniform Resource Locators (URL-er) er et subsett av URI-er og brukes som adresser til dokumenter på veven. Informasjon i RDF som omhandler vev-ressurser kan derfor være identifisert med en URL. Ting som ikke identifiseres som vev-ressurser kan også være beskrevet i RDF, men disse er da identifisert med en annen type URI enn en URL. Eksempler på slike ting er e-post-adresser og bøker. Alle URI-er følger konstruksjonsskjemaet nedenfor. Derfor er selv URI-er som ikke identifiserer ressurser på veven oppbygd etter samme lest som URL-er [24, s. 22].

```
skjema:[//autoritet]bane[?spørring][#fragment]
```

**Skjema** klassifiserer URI-typen, for eksempel *http, mailto* eller *file*. **Autoritet** angir vanligvis et domenenavn som for eksempel *www.example.com*. **Bane** organiseres hierarkisk med "/" som separator. **Spørring** er valgfritt å ta med og brukes typisk for å angi parametere til f.eks. vev-tjenester. **Fragment** brukes ofte for å angi deler av en ressurs [24, s. 23].

En vanlig misforståelse om URI-navngivningens egenskaper er at en URI er alene om å identifisere én bestemt ting. Slik unik identifiserering er ikke mulig hverken med URI-er eller noe annet symbolsystem. De samme URI-ene kan brukes for å navngi ulike ting og ulike URI-er kan brukes til navngivning av den samme tingen. URI-ers oppbygning med domener fører imidlertid til at sannsynlighten for at samme navn brukes om forskjellige ting er langt mindre enn den er for mange andre symbolsystemer. Dermed reduseres forekomsten av semantiske heterogeniteter mellom datasett når URI-er brukes til navngivning. Dette et argument som taler i favør av bruken av URI-er til identifisering av ressurser i åpne, offentlige datasett.

```
kommune:Nes eksempel:iFylke fylke:Buskerud .
kommune:Nes eksempel:iFylke fylke:Akershus .
```

Disse to triplene illustrerer at de samme URI-ene i teorien kan brukes for å navngi ulike ting. Ett av triplene utrykker at en ressurs, *kommune:Nes*, ligger i Buskerud fylke, mens det andre trippelet utrykker at den samme ressursen ligger i Akershus fylke. Dermed brukes samme URI for å navgi det som åpenbart er to forskjellige kommuner: Nes i Buskerud og Nes i Akershus.

### 3.1.4 Vokabularer

Det er i RDF vanlig å benytte kjente *vokabularer* for navngivning. Vokabularer brukes i RDF om samlinger av navn med en klart definert betydning [24, s. 33]. Disse navnene er ressurser representert av URI-er som består blant annet av vokabulareierens/-skaperens domenenavn.

Det finnes vokabularer innenfor en rekke områder som beskriver ofte anvendte ressurser. RDF er selv et eksempel på et slikt vokabular og inneholder blant annet ressursen: http://www.w3.org/1999/02/22-rdf-syntax-ns#type, navngitt med W3Cs domene med mer. Denne ressuren har en allmenn oppfattet betydning à la: i et trippel hvor denne ressursen opptrer som predikat, er subjektressursen av typen (klassen) angitt av objektressursen. Denne ressursen er så mye brukt at den i Turtle-syntaksen har en egen syntaktisk forkortelse: tegnet a. Andre velkjente vokabularer er blant andre "Friend of a friend"[1] (FOAF) som inneholder ressurser til bruk for å beskrive mennesker og relasjoner mellom dem og "Description of a Project"[2] (DOAP) som kan brukes for å beskrive prosjekter.

Ved gjenbruk av navn fra eksisterende vokabularer unngås det å skape nye ressurser som representerer forhold og ting allerede beskrevet av andre. Dette forutsetter konsistens i form av at gjenbruken begrenses til en måte som semantisk overholder navnenes tiltenkte betydning. Slik gjenbruk skaper felles forståelse for ofte brukte ressurser. Dataene i et fremmed datasett forstås raskere av både mennesker og maskiner dersom de er ressurser som fra før er kjente.

### 3.1.5 Navnerom

Et *navnerom* er delen av en URI som utgjør et bestemt vokabular, et felles prefiks navnene i vokabularet består av. FOAFs navnerom er: http://xmlns.com/foaf/0.1/. De ulike navnene i FOAF er URI-er som alle starter med dette navnerommet. For eksempel: http://xmlns.com/foaf/0.1/maker eller http://xmlns.com/foaf/0.1/homepage.

Navnerommene kan enten avgrenses med en skråstrek, "/", slik som i FOAF, eller med et firkanttegn, "#", slik som i DOAP; http://usefulinc.com/ns/doap#programming-language og http://usefulinc.com/ns/doap#license. Skråstreknavnerom og firkanttegnnavnerom har litt ulike bruksområder [34], men disse forskjellene er utenfor rapportens omfang og tas derfor ikke med.

[1] http://www.foaf-project.org/
[2] http://trac.usefulinc.com/doap

URI-er inneholder ofte mange tegn. Navnerom forkortes derfor gjerne slik at vi slipper å skrive deres URI fult ut for hver ressurs de er inneholdt i. Slik forkortning skjer ved at navnerommene først deklareres med et prefiks som angir hva forkortelsen skal refereres til som, og en URI som fult ut angir navnerommet. Ressurser kan så refereres til ved å angi forkortelsen for navnerommet den tilhører, etterfulgt av "`:`" og dens navn; for eksempel forkortelsen `foaf:maker` for `http://xmlns.com/foaf/0.1/maker`.

### 3.1.6 Blanke noder

En eksistensielt kvalifiserbar variabel beskriver at det finnes en ting, uten at det sies noe om tingens identitet. I RDF brukes såkalte *blanke noder* for å representere ekstensielt kvalifiserebare variabler. `(_:blank1 :b :c)` betyr at "det finnes en ting, blank1, slik at blank1 er b-relatert til c." Når vi i RDF ønsker å beskrive noe vi vet at finnes, hvis identitet ikke er viktig/kjent, bruker vi en blank node for å referere til denne tingen, for eksempel en geografisk posisjon bestemt av en lengdegrad og en breddegrad. Det er ikke hensiktsmessig å navngi alle posisjoner for de utallige kombinasjonene av lengde- og breddegrader. Blanke noder er mer egnet for å angi slike. Blanke noder brukes også når vi ønsker å gruppere utsagn og for å representere forhold mellom tre eller flere ressurser, slik som at "x kjøper y av z".

Blanke noder navngis ikke med URI-er siden de representerer noe hvis identitet ikke viktig. Allikevel kan blanke noder refereres til med navn i RDF-serialiseringer. Disse navnene er kun unike innenfor konteksten til det aktulle RDF-dokumentet og er ikke refererbare utenfor dette [23, s. 84]. Slik intern navngivning er nødvendig for å eksplisitt skille blanke noder fra hverandre og for å sikre at den samme blanke noden skal kunne være subjekt eller objekt i vilkårlig mange tripler.

I Turtle-syntaksen representeres navngitte, blanke noder med en understrek "`_`" på plassen før kolon "`:`" som vanligvis opptas av et navneromprefiks etterfulgt av et variabelnavn (se avsnitt 3.1.5 på forrige side). Eksempel som sier at *Ivo Caprino* og *Sonja Henie* begge har en, den samme, fødebyen:

```
dbpedia:Sonja_Henie dbp-ont:birthplace _:id1 .
dbpedia:Ivo_Caprino dbp-ont:birthplace _:id1 .
```

Blanke noder kan også refereres til implisitt uten navn. I Turtle beskrives slike noder med hakeparenteser "`[]`", der predikater og objekter tilhørende den blanke noden angis mellom parentesene. Disse nodene kan ikke refereres til andre steder, siden de er uten navn. Eksempel som uttrykker at *Rasletind*

ligger på et punkt (blank node) som har en breddegrad *geo:lat* og lengdegrad *geo:long*:

```
fjell:Rasletind geo:location
          [ geo:lat "61.4"^^xsd:float ;
            geo:long "8.7"^^xsd:float . ] .
```

Det er kun nodene i RDFs grafmodell som kan være blanke, ikke kantene. En blank node kan altså ikke oppta predikatets plass i et trippel.

### 3.1.7 Fletting av RDF-grafer

Som nevnt innledningsvis om RDF, ligger en datamodell der tripler utgjør rettede grafer til grunn for dette rammeverket. Hvis to eller flere RDF-grafer, bestående av et vilkårlig antall tripler, flettes sammen, dannes en ny rettet graf. Fletting av rettede grafer er en enkel prosess. Siden navngivning av blanke noder kun er unik innenfor RDF-dokumentet nodene er beskrevet i, må imidlertid spesielle hensyn tas når to RDF-dokumenter og deres underliggende grafmodeller skal flettes sammen.

Gitt at vi har to Turtle-dokumenter, der dokument $A$ inneholder triplene `dbpedia:Grotten geo:pos _:id1` og `_:id1 geo:lat "59.918"`, mens dokument $B$ består av triplene `borgerkanalen:12352 geo:pos _:id1` og `_:id1 geo:lat "60.54"`. Hvis grafene som representerer disse triplene slås sammen uten å ta hensyn til de blanke nodene, får vi et resultat der de to stedene, Grotten og Borgerkanalsaken, ligger på samme geografiske posisjon (geo:pos) som har to ulik breddegrader (geo:lat), "59,918" og "60.54": `_id1 geo:lat "60.54", "59.918" .` Dette som følge av at de blanke nodene i hvert av dokumentene er navngitt likt.

Ved fletting av RDF-grafer må derfor alle blanke noder navngis på nytt slik at ingen av de blanke nodene i en av de opprinnelige grafene får samme navn som noen av de blanke nodene i noen annen graf den skal flettes sammen med. Sammenflettingen av dokumentene $A$ og $B$ vil med hensyn til dette resultere i følgende tripler:

```
dbpedia:Grotten geo:pos _:id1 .
borgerkanalen:12352 geo:pos _:id2 .
_:id1 geo:lat "59.918 .
_:id2 geo:lat "60.54" .
```

## 3.2 RDFS, OWL og resonnering

RDF brukes for å beskrive egenskaper ved- og forhold mellom entiteter; slik som at en bygård ble bygget et bestemt år: `byantikvaren:23322 byantikvaren:byggeaar "1900"`. *RDF Schema* (RDFS) er en utvidelse av RDF, som i tillegg tilbyr muligheter for å beskrive terminologisk kunnskap i form av klasse- og egenskaphierarkier og semantiske avhengigheter mellom slike [24, s. 67]. RDFS er selv et RDF-vokabular og alle RDFS-dokumenter er dermed også RDF-dokumenter [24, s.46].

Det er vanligvis lett for mennesker å forstå den tiltenkte meningen til ressurser i RDF utifra navnet, eksempelvis at "byantikvaren:byggeaar" refererer til årstallet en bygning ble bygget på. For en maskin gir ikke slike ressurser i seg selv noen mening, "byantikvaren:byggeaar" er for maskiner kun en tekststreng og kunne like gjerne vært brukt som predikat til et objektliteral med verdien "hakkespett" som "1900". I RDFS kan denne typen mening, semantikken, eksplisitteres gjennom definerte terminologier. RDFS-terminologien:

```
dbpedia:Architect rdfs:subClassOf dbpedia:Person .
dbpedia:birthPlace rdfs:domain dbpedia:Person ;
                   rdfs:range dbpedia:Place .
```

beskriver deler av DBPedia. Det første trippelet sier at "Architect" er en subklasse av person, mens de to andre triplene sier at predikatet "birthplace" brukes for å beskrive et forhold der subjektet, predikatets domene, er av typen "Person" og objektet, predikatets verdiområde, av typen "Place". Dermed gjøres det mulig, også for maskiner, å avlede at ressursen "dbpedia:Lillehammer" er av typen "dbpedia:Place" utifra ovennevnte tripler i kombinasjon med trippelet `dbpedia:Nils_Slaatto dbpedia-o:birthPlace dbpedia:Lillehammer`, eller at `dbpedia:Nils_Slaato` er en "dbpedia:Person" basert på `dbpedia:Nils_Slaato a dbpedia-o:Architect`.

Prosessen med å utlede nye data basert på eksisterende data kalles *resonnering*. Innen RDF tilsvarer dette at nye tripler utledes fra allerede tilgjengelige tripler. Resonnering foregår ved hjelp av logiske slutninger basert på mønstre, eksempelvis typepropagering: "Slaatto er en arkitekt og en arkitekt er en person, derfor er Slaatto en person".

*Web Ontology Language (OWL)* er en standard, anbefalt av W3C, som brukes for å modellere ontologier [24, s. 111]. En ontologi er en formell beskrivelse av forholdet mellom begreper. DBPedia-terminologien beskrevet ovenfor i RDFS er dermed en ontologi. Med OWL-ontologer kan mer avansert informasjon enn den som lar seg utrykke i RDFS, beskrives; slik som at "en

geografisk posisjon har akkurat én tilknyttet lengdegrad og én breddegrad".

I likhet med RDF(S) har OWL en formell semantikk som beskriver de mulige slutningene som kan gjøres i språket. Gitt en ontologi i OWL eller RDFS, kan den korresponderende formelle semantikken anvendes for å avlede ny informasjon basert på ontologien.

Den mest brukte syntaksen i OWL-dokumenter er den samme som brukes i RDF. OWL-dokumenter som følger denne syntaksen er dermed også gyldige RDF-dokumenter [24, s.112]. Dette gjør at OWL-dokumenter enkelt kan knyttes til andre RDF-dokumenter gjennom lenker — bli en del av den semantiske veven, og resonneres mot i kombinasjon med slike.

## 3.3 Konvertering av data til RDF-formater

Blant etatene som i dag tilgjengeliggjør sine data, er det få som leverer disse beskrevet med RDF. Data finnes på mange forskjellige formater: i databaseform, som filer på XML- eller .csv-format, eller i tekstdokumenter lokalt eller på veven. For at disse dataene, uavhengig av opprinnelig format, skal kunne benyttes av semantisk vev-teknologier, må de først konverteres til en RDF-serialisering.

Vanskelighetsgraden i å konvertere data til RDF varierer veldig fra format til format. Konvertering kan skje programmatisk ved å lese inn en fil på et gitt format i et programmeringsspråk, tolke dataene og produsere en ny fil med de samme dataene som RDF. Data som finnes i et XML-format fra før, kan konverteres til RDF ved bruk av EXtensible Stylesheet Language (XSLT) der ulike merker, elementer og attributter i originalfilen spesifiseres til å erstattes med andre slike i en ny målfil på f.eks RDF/XML-format.

Data på vevsider er gjerne pakket inn som tekst i ulike HTML-etiketter, slik som HTML-merker. Data som hører til samme datasett på veven er ofte spredt utover mange forskjellige dokumenter med forskjellige URL-er, eksempelvis data om ulike fjelltopper, der http://peakbook.org/peak/5/Glittertinden.html og http://peakbook.org/peak/115/Nautgardstinden.html beskriver to forskjellige instanser av slike som hver har en mengde data, som navn og høyde, tilknyttet seg i HTML-tabeller. For å konvertere disse dataene til RDF kan man traversere de ulike dokumentene og for hver av dem "skrape" de aktuelle dataene fra HTML-koden og lagre disse i nye RDF-serialiserte filer — tidkrevende, tungvint arbeide.

Det er ikke bare filformatene som er forskjellige fra datasett til datasett. Måten dataene er strukturert på varierer også veldig, uavhengig av filenes

| Informasjon | Vernestatus | Bydelsnavn |
|---|---|---|
| Uthus | Bevaringsverdig | Nordstrand |
| Våningshus | Foreslått fredet | Sagene |
| | Bevaringsverdig | Sagene |

Tabell 3.1: Utdrag fra Byantikvarens gule liste over verneverdige bygg i Oslo

format. Derfor må det gjøres spesielle tilpasninger før konvertering av nær sagt alle datasett. Dette gjør det vanskelig å lage programmer som tar data av én bestemt filtype og returnerer en fil med disse konvertert til en RDF-serialisering.

Det kreves en viss domeneforståelse for de opprinnelige dataene slik at de tolkes riktig og konverteres til godt strukturerte RDF-tripler — tripler der blant annet URI-er fra veletablerte vokabularer brukes når det er hensiktsmessig, slik som `foaf:name` som predikat for å beskrive navnet til en person. Følgende steg kan benyttes for å konvertere data strukturert i tabellform til RDF-tripler:

- En egnet ressurs skapes, eventuelt viderebrukes, for å beskrive innholdet i hver av tabellens rader. Disse radene svarer til subjekter i RDF. I de fleste tilfeller vil det være naturlig å skape identifikatorene til ressursene som beskrives i datasettet, subjektene, med egendefinerte URI-er som kan knyttes til et egeneid domene, for eksempel: `www.mittdomene.no/id/personer/ansatte/01`.
- En kolonne i tabellen svarer gjerne til et RDF-predikat. For å representere disse brukes ressurser fra velkjente vokabularer dersom dette er egnet. Ellers lages nye URI-er.
- Tabellens celleverdier svarer til objekter i RDF. Disse kan representeres som en literal dersom cellen beskriver en konstant verdi, eller som en ny eller gjenbrukt ressurs, typisk hvis cellen beskriver en ting som refereres til andre steder i tabellen.

Dataene fra tabell 3.1 kan med disse stegene konverteres til RDF (Turtle-serialisert) som vist i figur 3.5 på neste side.

## 3.4 SPARQL

SPARQL Protocol and RDF Query Language (SPARQL) er et spørrespråk og en protokoll for RDF. Spørrespråket SPARQL brukes for å stille spørringer

```
@prefix id: <http://www.byantikvaren.oslo.kommune.no/id/> .
@prefix ont: <http://www.byantikvaren.oslo.kommune.no/ont/> .
@prefix dbpedia:  <http://dbpedia.org/resource/> .

id:1 ont:informasjon "Uthus" ;
     ont:vernestatus "Bevaringsverdig" ;
     ont:bydel        dbpedia:Norstrand .

id:2 ont:informasjon "Byvilla i mur" ;
     ont:vernestatus "Foreslått fredet" ;
     ont:bydel        dbpedia:Sagene .

id:3 ont:vernestatus "Bevaringsverdig" ;
     ont:bydel        dbpedia:Sagene .
```

Figur 3.5: Dataene fra tabell 3.1 på forrige side konvertert til RDF, Turtle-syntaks.

mot RDF-basert informasjon, mens SPARQL som RDF-protokoll er en måte å overføre SPARQL-spørringer fra spørreklienter til spørreprosessorer på [10]. Et *SPARQL-endepunkt* er en slik spørreprosessor: en tjeneste som tar i mot, prosesserer og returnerer svar på SPARQL-spørringer gjennom HTTP. SPARQL består også av et presentasjonsformat for spørringsresultater i XML [24, s. 262].

Det er først og fremst spørredelen av SPARQL som er sentral for oppgaven og som derfor vektlegges i denne rapporten. Det finnes også andre spørrespråk tilgjengelige for bruk mot RDF [23, s. 192]. SPARQL er det eneste av disse som beskrives her da det er anbefalt av W3C og støttes av endepunktene Semantikart knytter seg til, 5.3.2 på side 64.

### 3.4.1 SELECT-spørringer

SELECT-spørringer er den vanligste typen spørring i SPARQL. Nøkkelordet SELECT brukes for å identifisere variabler som skal opptre i spørringens resultat. SELECT-spørringer inneholder også en WHERE-del som beskriver et *grafmønster*. Disse mønstrene består av RDF-tripler hvor hver av trippelbestanddelene kan byttes ut med variabler [31]. I SPARQL matches grafmønstrene mot RDF-grafene det spørres mot, *spørregrafen*, ved hjelp av unifikasjon. Unifikasjon er en algoritme som brukes for å avgjøre hvilke tilordninger som må gjøres for at to uttrykk som kan inneholde variabler skal matche [26, s. 68]. En SELECT-spørring har en løsning for hver slik

```
PREFIX ont: <http://www.byantikvaren.oslo.kommune.no/ontologi/>
SELECT ?id ?vstatus
WHERE {
  ?id ont:vernestatus ?vstatus .
}
```

Figur 3.6: Enkel SPARQL-SELECT-spørring

| id | vstatus |
|---|---|
| <http://www.byantikvaren.oslo.kommune.no/id/1> | "Bevaringsverdig" |
| <http://www.byantikvaren.oslo.kommune.no/id/2> | "Foreslått fredet" |
| <http://www.byantikvaren.oslo.kommune.no/id/3> | "Bevaringsverdig" |

Tabell 3.2: Svar på spørringen fra figur 3.6 stilt til RDF-grafen representert i figur 3.5 på forrige side

mulige, unike tilordning av hver av variablene fra SELECT-delen til en ressurs eller literal fra spørregrafen slik at grafmønsteret matcher et subsett av spørregrafen.

Figur 3.6 viser et eksempel på en enkel SPARQL-SELECT-spørring. Denne spørringen identifiserer først prefikset "ont". Prefikser angis i SPARQL etter nøkkelordet "PREFIX" slik at vi slipper å skrive URI-er fult ut for hver ressurs prefiksene inngår i. Variablene ?*id* og ?*vstatus* identifiseres i SELECT-delen. Disse kan så brukes i WHERE-delen som trippelbestanddeler i grafmønsteret. Triplene i grafmønsteret skrives etter samme syntaks som i Turtle og variablene identifiseres med spørsmålstegn etterfulgt av ønsket variabelnavn. I eksempelspørringen består grafmønsteret av ett trippel hvor ?*id* opptar subjektets posisjon og ?*vstatus* objektets. Resultatet av en spørring er en liste med løsninger som korresponderer til måten spørregrafen matcher grafmønsteret på [31]. Listen inneholder et antall løsninger som hver svarer til et unikt subsett av spørregrafen som matcher grafmønsteret. De samme variablene kan brukes flere steder i mønsteret, men må være tilordnet den samme tingen for hver løsning. Hvis det ikke finnes noe subsett av spørregrafen som matcher grafmønsteret, er løsningslisten tom. Spørringen fra figur 3.6 stilt til en RDF-graf lik den i figur 3.5 på forrige side, vil resultere i en løsningssekvens med tre resultater, ett for hver person som er identifisert og har et navn. Se tabell 3.2. ?*id* tilordnes subjektene i hvert trippel, *ont* : *vernestatus* tilordnes predikatet *ont* : *vernestatus* og ?*vstatus* tilordnes literal-verdien som står som objekt til dette predikatet.

```
PREFIX ont: <http://www.byantikvaren.oslo.kommune.no/ont/> .

SELECT ?status ?informasjon
WHERE {
  ?id ont:vernestatus ?status .
  OPTIONAL { ?id ont:informasjon ?informasjon . }
}
```

Figur 3.7: SPARQL-SELECT-spørring med OPTIONAL-undermønster

| status | informasjon |
|---|---|
| "Bevaringsverdig" | "Uthus" |
| "Foreslått fredet" | "Byvilla i mur" |
| "Bevaringsverdig" | |

Tabell 3.3: Svar på spørringen fra figur 3.7 stilt til RDF-grafen representert i figur 3.5 på side 35

### 3.4.2 Undermønstre, OPTIONAL og UNION

Undermønstre kan lages blant grafmønstrene i SPARQL. Omfanget til et mønster i SPARQL avgrenses med krøllparenteser, "`{}`". Grafmønsteret i en spørring er avgrenset av de ytterste krøllparentesene i spørringens WHERE-del, mens eventuelle undermønstre beskrives innenfor disse.

OPTIONAL er et nøkkelord som brukes for å lage en type undermønster i en spørring. Et OPTIONAL-undermønster definerer strukturen til variabler som det ikke er påkrevd at skal være bundet til noe for å inngå som en del av en løsning. Hver løsning på spørringen må tilfredstille det som står i grafmønsteret, utenfor eventuelle OPTIONAL-undermønstere, mens det som står innenfor slike ikke nødvendigvis må tilfredstilles. I spørringen i figur 3.7 bindes bare variabelen $?informasjon$, i en subgraf av spørregrafen som tilfredstiller mønsteret utenfor OPTIONAL, dersom den tilfredstiller OPTIONAL-mønsteret. Tabell 3.3 viser resultatet av denne spørringen anvendt på RDF-grafen i figur 3.5 på side 35. Hvis et resultat inneholder en løsning der en variabel ikke er bundet (OPTIONAL-undermønsteret er ikke tilfredstilt) presenteres denne variabelen i løsningen som tom, slik som i den siste cellen i tabell 3.3.

Det finnes andre undermønstre enn OPTIONAL. Nøkkelordet UNION brukes for å spesifisere flere alternative mønstre [24, s. 268] innenfor et grafmønster der en løsning må tilfredstille minst ett av dem. Figur 3.8 på neste side viser et mønster med to alternativer som leverer løsninger der en

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?navn
{ ?person   a   foaf:Person .
  { ?person foaf:name      ?navn . } UNION
  { ?person foaf:givenName ?navn . }
}
```

Figur 3.8: SPARQL-SELECT-spørring med UNION-undermønstre

variabel `?navn` bindes til objektet som står til et predikat `foaf:name` og/eller et `foaf:givenName` på en ressurs av typen `foaf:Person`.

### 3.4.3 Filtrering

Resultatet av en spørring kan avgrenses ved filtrering. FILTER er et nøkkelord som brukes i SPARQL for å sette et kriterium som spesifiserte variabler må oppfylle for å kunne være en del av en løsning. Et FILTER-kriterium gjelder kun innenfor mønsteret det inngår i og filtrerer vekk løsninger som ellers tilfredstiller dette mønsteret dersom de ikke overholder kriteriet. Et mønster kan inneholde vilkårlig mange kriterier. Eksempler på noen filtreringskriterier:

- Begrense literaler til å være på et bestemt språk. *LANG* er en av flere unære spesialoperatorer til bruk på literaler i RDF [24, s.273-74].

  ```
  ...
  { ?sted dc:description ?beskrivelse .
    FILTER (LANG(?beskrivelse) = "nn")
  }
  ```

- Kreve at en literal er lik en tekststreng eller URI. Filtre kan kombineres med logisk konjunksjon (og), "&&", disjunksjon (eller), "||" , og negasjon (ikke), "!".

  ```
  ...
  { ?sted     a     ?t ;
            dc:title ?navn .
  FILTER ((?navn = "Meteorologisk institutt") &&
  (?t = <http://linkedgeodata.org/vocabulary#surveillance>))
  ```

- Begrense verdien på en literal av en numerisk type til å ligge innenfor et bestemt intervall.

```
PREFIX  dc: <http://purl.org/dc/terms/>
SELECT  ?title
WHERE {
 GRAPH <http://opendata.computas.no/dataset/byggesak_oslo> {
  ?sak dc:title ?title ;
 }
}
```

Figur 3.9: SPARQL-SELECT-spørring på navngitt graf

```
...
{ ?person eks:alder ?alder .
  FILTER (?alder >= 18 )
}
```

### 3.4.4 RDF-datasett og navngitte grafer

Mange RDF-datalagre består av flere RDF-grafer som hver inneholder en mengde tripler. For eksempel inneholder Computas Linked Open Dataserver, se 5.3.2 på side 65, én RDF-graf for hvert datasett, eksempelvis byggsaker og ladestasjoner for elbil. En SPARQL-spørring utføres på et *RDF-datasett* som består av alle grafene inneholdt på et datalager. Et RDF-datasett inneholder én standardgraf uten navn og null eller flere såkalte *navngitte grafer* [31, avsnitt. 8]. En navngitt graf er identifisert med en egen URI, et navn.

Størrelsen på RDF-datasettet en SPARQL-spørring skal matches mot kan begrenses ved å angi hvilke (navngitte) grafer spørringen skal hente data fra. På denne måten blir ikke spørringen matchet mot alle RDF-grafene i settet, men kun de, eller den, av dem som navngis i spørringen. Dette kan føre til en reduksjon av spørretiden siden grafmønsteret matches mot en tilsvarende mindre spørregraf.

Nøkkelordet GRAPH brukes i en SPARQL-spørring for å angi hvilken graf et mønster skal matches mot. Figur 3.9 viser en spørring som ber om tittelen på data fra grafen angitt av URI-en <http://opendata.computas.no/dataset/byggesak_oslo>.

### 3.4.5 Andre typer spørringer

Det finnes også andre typer SPARQL-spørringer enn SELECT. Disse er ikke like sentrale for masteroppgaven, og beskrives derfor kun kort i denne

rapporten.

- ASK brukes for å sjekke om et grafmønster er representert i spørregrafen. Det eneste som returneres i en slik spørring er "Yes" eller "No", avhengig av om mønsteret er representert eller ikke.

  ```
  PREFIX dc: <http://purl.org/dc/terms/>
  ASK { ?x dc:title "Aker Brygge" }
  ```

  Denne spørringen returnerer "yes" dersom det finnes minst ett trippel i grafen der predikatet er dc:title og objektet er literalen "Aker Brygge". Hvis ikke returneres "no".

- CONSTRUCT brukes for å returnere resultater formatert som et nytt RDF-dokument. CONSTRUCT-spørringer stilles med et mønster som beskriver hvordan RDF-en som skal returneres skal se ut [24, s. 277].

  ```
  PREFIX dbp-ont:  <http://dbpedia.org/ontology/>
  CONSTRUCT { ?place dbp-ont:isBirthPlaceOf ?person . }
  WHERE { ?person dbp-ont:birthplace ?place . }
  ```

  Spørringen ovenfor, dersom anvendt på en RDF-graf som inneholdt triplene fra figur 3.1 på side 24, ville returnert følgende tripler:

  ```
  @prefix dbpedia:  <http://dbpedia.org/resource/> .
  @prefix dbp-ont:  <http://dbpedia.org/ontology/> .

  dbpedia:Oslo dbp_ont:isBirthplaceOf dbpedia:Sonja_Henie ,
                                      dbpedia:Ivo_Caprino .
  ```

- DESCRIBE returnerer en RDF-graf med tilgjengelig informasjon om ressurser som matcher spørringen. Klienten trenger ikke på forhånd vite hvordan spørregrafen er konstruert for å stille slike spørringer mot den [23, s. 224].

## 3.5 Den semantiske veven og RESTfullhet

*Representational State Transfer*, REST, er en type arkitektur som er egnet til bruk i distribuerte systemer. Systemer som anvender seg av REST,

*RESTfulle systemer*, består av klienter og servere. Klienter sender forepørsler som serverne behandler og returnerer svar på.

RESTfulle systemer kommuniserer vanligvis, men ikke nødvendigvis, gjennom HTTP-protokollen [12, s. 100]. Gjennom HTTP tilbys verktøy som URI-er, Internet media types (*MIME-typer*), forespørsel- og svarkoder etc. Datakommunikasjon innenfor den semantiske veven skjer ofte via HTTP på RESTfulle systemer, slik som på Computas Linked Open Data Server som Semantikart kommuniserer med. RESTfulle systemer, avgrenset til en vevkontekst, er dermed sentrale for denne masteroppgaven. Termen RESTfulle systemer benyttes derfor i det følgende om de systemer som benytter seg av REST via HTTP.

I tradisjonell mellomprogramvare (middleware) og andre distribuerte systemer, blant annet de som benytter seg av Simple Object Access Protocol (SOAP), skjules ofte distribusjonen av fjerntliggende prosedyrekall, Remote Procedure Calls (RPC). Ved hjelp av slike kall kan distribuerte systemer håndteres som om de var ett lokalt system.

I REST er identifisering av ressurser det sentrale. REST bygger på en idé om desentralisering: tilgang til fjerntliggende ressurser gjøres eksplisitt gjennom identifikatorer uten at det forsøkes å skjule distribusjonen [45, s. 2]. Klienter og tjenere utveksler representasjoner av ressursene ved hjelp av klientforespørsler via HTTP mot ressursenes identifikatorer. Oppslag på ressurser i REST forutsetter derfor ikke noe programmeringsspråk med en tilhørende API, slik som tilfellet er med eksempelvis databaser. I REST er det ingen prinsipiell forskjell på hvordan forespørsler gjøres avhengig av hva slags type ressursene er av. Om ressursen er et datasett, tekst, lyd eller et dokument forandrer ikke forespørselen. Oppslag i RESTfulle systemer skjer dermed på en universell og generell måte.

RPC-baserte- og RESTfulle systemer har ulike bruksområder. I RPC fokuseres det på å skjule distribusjonen så langt det lar seg gjøre og med dette skape følelsen av at det jobbes mot ett homogent system. RPC er derfor populært blant annet innenfor enterprise-IT hvor det ofte er et eksplisitt mål om å skape ett integrert system. Dette er en forskjell fra REST, hvor fokus er arbeid mot heterogene, uforutsigbare miljøer [45, s. 2]. Verdensveven er et eksempel på et slikt miljø. For systemer som opererer her, er det nødvendig å raskt kunne koble seg til og viderebruke eksisterende tjenester, såkalte vevtjenester. En *vevtjeneste* er et sett med teknologier som anvendes for å tilby et grensesnitt på veven for interaksjon mellom maskiner. RESTfulle vevtjenester øker stadig i popularitet og REST finnes nå i bruk av store aktører som Google, Amazon, Flicker og Yahoo. Flere av disse aktørenes REST-API-er (REST-vevtjenester) er mer brukt enn de samme aktørenes SOAP-API-er (SOAP-vevtjenester) [45, s. 2][1, s. 1].

| | |
|---|---|
| GET | Returnerer liste med URI-er og eventuelle andre data om instansene av denne typen |
| PUT | Bytter ut hele samlingen med en annen samling |
| POST | Lager en ny instans i samlingen. Den nye instansens URI lages automatisk og returneres |
| DELETE | Sletter hele samlingen |

Tabell 3.4: RESTfulle vevtjenesters (implementert ved hjelp av HTTP-forespørsler) effekt på en URI som navngir en en samling instanser av samme type, f.eks http://opendata.computas.no/data/norad/bistand

Hver komponent, som er tilgjengelig for brukeren, modelleres i RESTfulle systemer som en ressurs identifisert av en derefererbar URI. En derefererbar URI kan åpnes i en nettleser og der tilby mer informasjon om ressursen gjennom dokumenter som beskriver ressursens nåværende eller tiltenkte tilstand. En slik URI er altså en URL. URI-ene i RESTfulle systemer skapes vanligvis etter et skjema der kombinasjonen av protokoll, domene og applikasjonssti opptrer som et felles navnerom for alle ressursene inneholdt i systemet. Type og eventuell instans-ID brukes så for å identifisere en bestemt ressurs innenfor dette navnerommet [4]. protokoll://domene/applikasjonssti/type/instans-ID Ved hjelp av slike skjemaer identifiseres ulike instanser av data med samme type, f.eks. ulike instanser av bistand fra Norge til et annet land: http://opendata.computas.no/data/norad/bistand/152 og http://opendata.computas.no/data/norad/bistand/153. Ved å fjerne instans-ID-en fra URI-en kan man referere til en samling instanser av en bestemt type: http://opendata.computas.no/data/norad/bistand.

Sentralt for RESTfulle vevtjenester er et sett av operasjoner som brukes for å endre eller lese systemets tilstand, uavhengig av måten dataene er lagret på og måten som brukes for å hente dem ut av systemet [4]. Dette settet består av operasjonene Create, Read, Update, Delete, (*CRUD*), som brukes henholdsvis for å lage, lese, oppdatere og slette en ressurs i et RESTfullt system. Tilsvarende operasjoner finnes støttet i HTTP-protokollen som HTTP-forespørsler. Create tilsvarer HTTP-POST, Read - HTTP-GET, Update - HTTP-PUT, mens Delete svarer til HTTP-DELETE. Tabellene 3.4 og 3.5 på neste side viser hvordan ulike HTTP-forespørsler vanligvis brukes for å implementere RESTfulle vevtjenester.

Måten ressursen presenteres på som svar til en HTTP-forspørsel, avgjøres i REST ved *innholdsforhandling*. En innholdsforhandling baseres på hva slags MIME-type(r) som spesifiseres i en forespørsels *HTTP-aksepthode* og hvilken klienttype som står for forespørselen. En ressurs returneres i det høyest prioriterte formatet, MIME-typen, som serveren støtter. Dersom

| | |
|---|---|
| GET | Henter en representasjon av ressursen |
| PUT | Skriver over den valgte ressursen dersom den fantes fra før. Hvis ikke, lages den |
| POST | Instansen behandles som en samling selv. Nye data kan lages under denne. |
| DELETE | Sletter instansen fra samlingen den inngår i |

Tabell 3.5: RESTfulle vevtjenesters (implementert ved hjelp av HTTP-forespørsler) effekt på en URI som navngir en en instans av en spesifikk type, f.eks http://opendata.computas.no/data/norad/bistand/153

ingen MIME-type er angitt, eller de som angis ikke støttes av serveren, returneres en representasjon av ressursen valgt på bakgrunn av klienttypen. For eksempel kan et XML-dokument returneres som et pent formatert vevdokument til en nettleser, mens XML-koden returneres rått når klienten er en Java-applikasjon.

Hva slags MIME-typer som skal støttes i en HTTP-forspørsel kan angis for hver av CRUD-operatorene. Slik kan for eksempel en HTTP-GET-forespørsel returnere en ressurs representert som RDF/XML, JSON eller HTML, avhengig av hva slags MIME-type som settes i hodet. En HTTP-POST-forespørsel kan legge til en ny ressurs i systemet basert på data i en fil på et format angitt i HTTP-aksepthodet, f.eks Turtle. Det forutsettes at systemene som mottar forespørselene har implementert støtte for disse MIME-typene. Et RESTfullt system på veven får gjennom HTTP en slags API der funksjonen av et kall gis av tuppelet: `{RessursURI, HTTP-forespørseltype, HTTP-hode}` [4]. Kommunikasjonen i RESTfulle systemer gjøres gjennom slike kall.

REST og semantisk vev-teknologier baserer seg begge på vevens allmenne arkitektur. Måten ressurser identifiseres på i REST, med URI-er, tilsvarer ressursnavngivningen som foregår på den semantiske veven. Med HTTP kan ressurser enkelt legges til, endres og fjernes i RESTfulle systemer. Dette er ressurser som kan beskrives med RDF, med tripler som lenker til andre ressurser og beskriver tilknyttede literaler.

RESTfulle tjenester kan enkelt gjenbrukes og sammenstilles når alle ressursene identifiseres på en konsistent måte: gjennom URI-er som støtter HTTP-operasjoner [45, s. 2]. Ressursene i RESTfulle systemer inneholder lenker til andre ressurser i tråd med Berners-Lees retningslinjer slik beskrevet i 2.3.2. Applikasjoner som kobles opp mot slike systemer kan gjennom lenkene traversere dataene — både internt innen et system og eksternt, mellom andre systemer, som en vev av data.

Videre sikrer systemer som baseres på REST, når de brukes til publisering

av lenkede data beskrevet med RDF, mulighet for selvdokumenterende navngivning av ressurser. Dette skjer gjennom HTTP-URI-er, i tråd med Berners-Lees retningslinje nummer to. REST er av disse grunnene en type arkitektur som passer godt sammen med semantisk vev-teknologier og dermed publisering av åpne, offentlige data.

### 3.5.1 REST + SPARQL

RESTfulle systemer, brukt til publisering av åpne data beskrevet med RDF, kan manipuleres og leses innholdsmessig ved hjelp av vevtjenester over HTTP. Ressurser leses, legges til, endres og fjernes.

Med HTTP-GET kan man i RESTfulle systemer hente ut et sett bestående av alle instansene av en gitt type, eller en bestemt instans angitt av en URI. For å begrense settet med instanser som systemet returnerer basert på gitte kriterier, kan en vevtjeneste som gjør dette lages. Et eksempel på en slik funksjon, som er sentral for Semantikart, er å få returnert alle ressursene av en bestemt type som ligger i nærheten av et gitt geografisk punkt. En måte å gjøre dette på, er å lage en RESTfull vevtjeneste som en HTTP-GET-forespørsel, der et geografisk koordinat og en avstand sendes med som parameter. Denne funksjonen returnerer alle ressurser som er knyttet til et geografisk punkt beliggende maksimalt den gitte avstanden fra koordinatet.

Dersom serveren inneholder et SPARQL-endepunkt som opererer på de samme dataene som det RESTfulle systemet, kan en forespørsel som begrenser datamengden på ovennevnte måte alternativt stilles gjennom dette. En SPARQL-spørring der det lages et filterkriterium for avstand sendes da gjennom HTTP-GET til endepunktet, hvoretter et svar i valgt representasjonsformat returneres.

SPARQL og RESTfulle vevtjenester utfyller hverandre som mekanismer for manipulasjon og lesing av data på en server. De samme URI-ene brukes for å identifisere de samme dataene. Slik kan for eksempel nye RDF-serialiserte data legges til ved hjelp av en RESTfull vevtjeneste som senere spørres mot med SPARQL.

Funksjonaliteten som tilbys gjennom RESTfulle vevtjenester over HTTP og et SPARQL-endepunkt, kan på samme server overlappe. HTTP-GET på typenivå, f.eks på URI-en http://opendata.computas.no/data/tellus/produkt/ i et RESTfullt system kan svare til den samme mengden ressurser som returneres av en SPARQL-SELECT som ber om alle tripler fra en navngitt graf angitt av samme URI.

RESTfulle vevtjenester som leverer fra seg de samme dataene som en

SPARQL-spørring kan altså konstrueres. Problemet er bare at slike tjenester må være konstruert på forhånd for å kunne benyttes. Det må gjøres antakelser for hvordan de tilgjengelige dataene skal kunne viderebrukes. Dette hemmer dataenes åpenhet. Gjennom SPARQL-endepunkter kan klienter stille spørringer som aldri før har vært stilt. Slik kan tidligere ukjente sammenstillinger av data skapes. SPARQL-endepunkter tilbyr dermed utvidet funksjonalitet for uthenting av data i forhold til de RESTfulle tjenestene — funksjonalitet som ikke legger begrensninger på mulig viderebruk av data.

En fordel med RESTfulle tjenester kontra SPARQL-spørringer, er at de raskt og enkelt kan gi informasjon om en ressurs. For å få all informasjon om en ressurs' tilknyttede egenskaper holder det å gjøre et oppslag på denne ressursens URI. Med SPARQL må man gå veien om spørringer.

For å sikre mulighet for at klientene kan gjenbruke akkurat de dataene de måtte ønske, mulighet for HTTP-oppslag på URI-er og lettvekts CRUD-operasjoner på datamengden i en server med åpne data, bør den derfor både være RESTfull og tilby et SPARQL-endepunkt.

# Kapittel 4

# Mobile, håndholdte enheter som plattform for semantisk vev-applikasjoner

I dette kapittelet analyseres problemet masteroppgaven søker å belyse: "hvordan mobile, håndholdte enheter kan brukes som plattform for å lage en nyttig semantisk vevapplikasjon basert på åpne, offentlige data." Innledningsvis i kapittelet forklares hva som menes med mobile, håndholdte enheter og hvilke ulike plattformer som finnes tilgjengelige for applikasjonsutvikling mot slike.

Videre beskrives Semantikart, applikasjonen som har blitt utviklet for å illustrere hvordan mobile enheter og semantisk vev-teknologi kan kombineres på en interessant måte. Deretter beskrives en nyttig og aktuell type funksjonalitet som med hell kan knyttes til slike applikasjoner: stedsbasert innrapportering av feil og mangler i det offentlige. Til slutt forklares det hva som gjør mobile enheter spesielt egnet som plattform for semantisk vev-applikasjoner som Semantikart.

## 4.1 Mobile, håndholdte enheter

Mange mobiltelefoner leveres i dag med god maskinvareytelse, mulighet for Internettilgang og operativsystemer som lar brukeren installere egne applikasjoner på dem. Slike telefoner kalles for *smarttelefoner*. Det anslås at mobiltelefonen vil overta datamaskinens posisjon som den vanligste enheten å aksessere Internett med løpet av 2013 [18]. Mobiltelefoner er imidlertid ikke de eneste mobile enhetene som tilbyr støtte for nettilgang og installasjon av egne applikasjoner. Flere mobile medieavspillere tilbyr tilnærmet samme

funksjonalitet som smarttelefoner, bortsett fra telefoni. I det siste har det også vært en stor framvekst av såkalte nettbrett: mobile enheter spesielt konstruert for å konsumere informasjon tilgjengelig på nettet. Nettbrett er utstyrt med større skjermer enn hva som er tilfellet for smarttelefoner. Grensene mellom de ulike typene enheter er vage. Det finnes nettbrett med støtte for telefoni, mobiltelefoner med så store skjermer at de ikke lenger får plass i en vanlig bukselomme og telefoner der medieavspillingsmulighetene er lagt mer vekt på enn telefonistøtten. I det følgende brukes samlebetegnelsen: *mobil, håndholdt enhet (MHE)* om en enhet som har nettilgang, mulighet for installasjon av tredjepartsapplikasjoner og som er mobil i den forstand at den enkelt kan medbringes "over alt" og er skapt for å opereres fra håndholdt posisjon, uten å måtte støttes mot noen annen flate.

Tradisjonelt sett har applikasjonsbruk og nettilgang foregått ved bruk av stasjonære- og i senere tid bærbare datamaskiner. En bærbar datamaskin har en mobilitet som begrenses av at den er for stor til å kunne medbringes i en bukselomme og en bruk som er tilpasset at maskinen plasseres på en flate for å kunne gjøres effektivt. Med MHE-er kan applikasjoner og Internett brukes effektivt håndholdt sittende, liggende, stående eller gående, utendørs så vel som innendørs.

MHE-ers andel av innebygde verktøy gir videre muligheter for tjenester. MHE-er kan inneholde verktøy som avgjør enhetens geografiske posisjon (GPS), innebygde kameraer, kompass og tilkoblingsmuligheter til trådløse nettverk etc. Det finnes mange typer tjenester som drar nytte av et eller flere av slike innebygde verktøy i kombinasjon med MHE-ers mobilitet, for eksempel applikasjoner som lar brukeren ta bilde av en ting for så å gjøre et sammenlignende bildesøk over nett som avgjør hva slags ting det er snakk om, eller spillapplikasjoner som bruker bilder fra brukerens kamera til grafikkelementer.

Verktøy for lokalitetsbestemmelse (GPS) i kombinasjon med MHE-ers mobilitet gjør slike enheter til en spesielt egnet plattform for tjenester basert på en kontekst gitt av enhetenes posisjon. For eksempel kan applikasjoner som viser informasjon om vilkårlige ting i nærheten av brukerens posisjon (matbutikker, kinoer, parkeringshus etc.), eller hjelper brukeren med å finne veien fra sin posisjon til et gitt punkt, lages med utgangspunkt i dette.

### 4.1.1 Applikasjonsplattformer for MHE-er

Mobile plattformer, slike som Android, Apple IOS, Windows Mobile og Symbian, benytter seg av ulike programmeringsspråk og forskjellige rammeverk som en applikasjon må støtte for å kunne kjøre på hver av

disse. Derfor kan vanligvis ikke en applikasjon utviklet til én av plattformene også kjøres på en av de andre. Enkelte plattformer er også proprietære. Slike plattformer kan kun brukes på en enhet som er laget av en bestemt fabrikant. Et eksempel på dette er Apples IOS der kun enheter laget av Apple (Iphone, Ipod og Ipad) er støttet. Den eneste måten en applikasjon kan gjøres tilgjengelig for bruk på alle MHE-plattformene, er som en applikasjon laget for verdensveven, en vevapplikasjon. MHE-er har mulighet til å kjøre slike applikasjoner gjennom innebygde nettlesere.

De ulike mobilplattformene stiller alle egne metoder og verktøy til rådighet for utvikling av applikasjoner mot deres plattform. Egne API-er, grafiske verktøy for brukergrensesnittsutforming og ulike publiseringsløsninger finnes for hver av dem. Det søkes å gjøre utviklingen mest mulig behagelig for utviklerne slik at flest mulig applikasjoner skal utvikles til en gitt plattform; både fordi selskapene som eier plattformen vanligvis tar en prosentvis andel av salgssummen et kjøp av en applikasjon til deres plattform gir, og fordi jo flere (populære) applikasjoner som finnes tilgjengelig på en plattform, jo større oppslutning får denne plattformen. I alternativet, utvikling av en vevapplikasjon, må det tas høyde for at enhetene som kjører applikasjonen kan være svært ulikt utstyrt, både med tanke på innebygd funksjonalitet, maskinvareytelse og skjermstørrelse. På enkelte av plattformene varierer de ulike enhetenes egenskaper innen dette området veldig, men på disse kan utviklerne, gjennom plattformens biblioteker, enkelt tilpasse funksjonaliteten i applikasjonen basert på enheten som kjører det sine egenskaper. Plattformens programmeringsbiblioteker inneholder metoder som avgjør hva slags maskinvare som finnes tilgjengelig og lar utviklerne tilpasse applikasjonens funksjonalitet etter dette. Slik tilpassing er ikke like lett tilgjengelig ved utvikling av vevapplikasjoner.

Tilgangen på MHE-ers innebygde verktøy, slik som kamera og GPS, kan gjøres gjennom kall til den respektive utviklingsplattformens API-er for dette. Applikasjoner utviklet til veven mangler tilgang til slike innebygde verktøy. Applikasjoner spesielt utviklet for en plattform, såkalte *hjemmehørende applikasjoner*, har dermed et fortrinn i forhold til vevapplikasjoner på MHE-er. Hjemmehørende applikasjoner har også tilgang til og mulighet for å lagre data og seg selv i MHE-ens lagringssystem. Slik kan hjemmehørende applikasjoner startes, lagre data og benyttes uten at MHE-en må være tilkoblet Internett. Hjemmehørende applikasjoners hastighet begrenses derfor kun av maskinvareytelsen, såfremt den ikke er avhengig av å hente data fra nettet. Vevapplikasjoner avhenger derimot av Internettilkobling både for å kunne åpnes og benyttes. Dermed har slike en hastighet og responstid som er begrenset av Internetthastigheten og en bruk som avhenger av nettilgang.

MHE-ers fysiske oppbygning med knapper og funksjonalitet knyttet til

disse varierer fra plattform til plattform. For eksempel har Android-baserte enheter egne meny- og tilbakeknapper, mens enheter som kjører IOS ikke har dette. En bruker av en Android-applikasjon vil derfor forvente at et trykk på meny-knappen åpner en meny, fordi dette er vanlig oppførsel og enkelt å implementere i en hjemmehørende Android-applikasjon. Slik funksjonalitet er det vanskelig å implementere for vevapplikasjoner fordi funksjonalitet i slike ikke kan bindes til en enhets fysiske knapper. Når en vevapplikasjon kjøres på Android, er det nettleserapplikasjonens funksjonalitet disse knappene er koblet mot, ikke vevapplikasjonens.

Siden hjemmehørende plattformer er forskjellige, blant annet med tanke på bruk av innebygde fysiske knapper, må en vevapplikasjon som skal støtte alle plattformene forsøke å generalisere dette. Følgelig må den lages uten støtte for slike knapper. Dette kan være med å videre svekke slike applikasjoners posisjon i forhold til applikasjoner spesielt utviklet til én plattform, for eksempel Android-plattformen.

De alternative måtene å utvikle applikasjoner på til MHE-er, enten som en type hjemmehørende applikasjon eller som en vevapplikasjon, har begge fordeler og ulemper. Siden det finnes mange ulike plattformer for hjemmehørende applikasjoner og applikasjoner ikke fungerer på tvers av disse, kan det være fristende å lage en applikasjon som kan kjøres uavhengig av plattform, gjennom en nettleser. Vevapplikasjoner har imidlertid visse mangler i forhold til hjemmehørende applikasjoner slik beskrevet ovenfor. Hjemmehørende applikasjoner benytter seg av rammeverk som er skreddersydd for bruk på MHE-er. Slike applikasjoner utnytter dermed MHE-ers egenskaper og innebygde verktøy bedre og på en enklere måte, og gir i større grad en følelse av å være utviklet for bruk på en MHE enn hva som er tilfelle for en vevapplikasjon.

## 4.2 Semantikart — en applikasjon basert på åpne, offentlige data og semantisk vevteknologi på mobile, håndholdte enheter

Det finnes mange typer åpne, offentlige data. En applikasjon som nyttiggjør seg av dem alle, er det neppe mulig å implementere. Det måtte derfor gjøres en avgrensning på hvilke åpne data systemet som ble utviklet i forbindelse med masteroppgaven, skulle benytte seg av. Dette resulterte i delproblemstillingen: *hvilke offentlige data skal systemet baseres på?* En annen problemstilling som henger sammen med denne, er: *hvordan lage en interessant tjeneste på bakgrunn av dataene som velges?*

Som nevnt i 2.3.1 på side 17, ligger W3Cs ideer for den semantiske veven, hvor RDF brukes som modell for å beskrive data, som en forutsetning for denne masteroppgaven. De åpne, offentlige dataene oppgaven baseres på, er derfor begrenset til slike publisert med RDF. Dette er en begrensning som ikke utelukker åpne, offentlige data uten RDF, fordi disse kan heves til RDF-beskrevne data jamfør Berners-Lees 5-stjerners data, 2.3.2 på side 21.

Mange data er tilknyttet en geografisk posisjon, såkalte *geolokaliserte data*. En populær måte å visualisere slike på er som punkter på et kart. Kartdata lar seg kombinere med alle data som kan knyttes til en geografisk posisjon. Kartvisualiseringer gir data en fysisk tilknytning som brukere kan dra nytte av med tanke på å navigere seg frem til dataenes plassering. Data visualisert på kart, er data som kombinereres med noe som er kjent for brukeren fra før. Dette gir brukeren en måte å plassere dataene i kontekst på. Kart er også et visuelt element som kan ha stor detaljrikdom og bedre brukeropplevelsen fordi de oppleves som fine å se på.

Det finnes flere tjenester som lar utviklere tilby kartfunksjonalitet i sine applikasjoner. Google Maps er det mest populære av slike. Her tilbys en tjeneste for å vise detaljerte kart over hele verden, med støtte for å navigere i, zoome inn og ut på, og plotte eller tegne figurer og mønstre på disse. Google Maps inneholder API-er som kan benyttes for å tilby kartfunksjonalitetet i vevapplikasjoner såvel som i hjemmehørende applikasjoner på plattformer som Android og Apple IOS.

Kartvisualisering av data, inkludert brukerens posisjon, på mobile, håndholdte enheter, gir brukere mulighet til å oppdage ting i geografisk nærhet til sin posisjon og å finne veien fram til disse. Avstanden mellom brukeren og et geolokalisert dataobjekt kan regnes ut og være med på å tilby mer informasjon i en visualisering.

MHE-ers mobilitet kombinert med innebygde verktøy for lokalitetsbestemmelse, gjør som nevnt slike enheter spesielt egnet som plattform for tjenester basert på en kontekst gitt av enhetens posisjon. Mulighet for denne typen datavisualisering ble derfor valgt som en funksjonalitet som systemet skulle støtte fordi det gir en generell måte å presentere stedsbaserte data på som brukeren kan nyttiggjøre seg av og som forholdsvis enkelt kan implementeres ved hjelp av ulike kartfunksjonalitetstjenester, slik som Google Maps. Denne visualiseringsmåten gir også en videre naturlig avgrensning av hvilke åpne, offentlige data systemet skal bygges rundt: geolokaliserte data.

Kartvisualiseringen er så sentral for systemet at den er med på å forme det som har blitt applikasjonens navn: *Semantikart*. En annen viktig komponent, som utgjør den første delen av navnet, "Semantik", er den semantiske veven — sentralt som følge av semantisk vev-tilnærmingen til åpne data som

oppgaven forutsetter.

At Semantikart føles som den er skreddersydd for en MHE og har mulighet for å enkelt nyttiggjøre seg av slike enheters innebygde verktøy, som GPS og kamera, var utslagsgivende for valget av utviklingsplattform. Semantikart er derfor utviklet som en hjemmehørende applikasjon. Plattformen som er valgt er ikke-proprietær, har åpen kildekode og er den flest smarttelefoner som selges i dag leveres med [38]: Android-plattformen.

Brukergenererte data kan berike eksisterende informasjon. I applikasjoner som visualiserer data, vil muligheter for brukeren til å legge inn egne data, kommentere og/eller rangere eksisterende data gi tjenesten en ekstra dimensjon. Muligheter for å lage slike kommentarer/infopunkter, eller nye Borgerkanalsaker, er derfor implementert i MHE-applikasjonen Semantikart.

## 4.3 Stedsbasert innrapportering av feil eller mangler i det offentlige — Borgerkanalen

Hull i veien, mangelfull snørydding eller skumle istapper som henger ned fra hustak er eksempler på problemer som til stadighet dukker opp i det offentlige rom. For å få bukt med slike problemer må den myndighet som står som ansvarlig for å rette opp i et slikt problem på et gitt sted kontaktes. Det å finne ut hvem det er riktig å kontakte i slike tilfeller kan by på vanskeligheter. Få mennesker går rundt med full oversikt over hva eller hvem som er ansvarlig myndighet for enhver type sak på ethvert sted de befinner seg. Med en automatisert måte å finne og varsle rett myndighet på i forbindelse med dette, kan hastigheten og frekvensen av innrapportering, videre saksgang og utbedring av feil eller mangler i det offentlige økes.

Flere tjenester som søker å tilby funksjonalitet for denne typen automatisert innrapportering har dukket opp, eller er under utvikling. Britiske FixMyStreet[1], som i dag er hyppig brukt, var en av de første slike tjenestene og har inspirert mange andre lignende initiativer. Her hjemme fikk vi et slikt i forbindelse med engangs-prosjektstipendordningen Nettskap 2.0, som ble opprettet i regi av FAD våren 2010. Målet med Nettskap var å stimulere utvikling av tjenester med brukergenerert innhold, gjerne basert på åpne data. Ordningen var inspirert av lignende initiativ fra andre land, slik som Apps for Democracy i USA (se 2.1.2 på side 7) og resulterte i 16 ulike prosjekter, med et totalt støttebeløp på 2,5 millioner kroner [13]. Et av prosjektene som ble tildelt midler var *Borgerkanalen*, et prosjekt startet av biveileder for denne masteroppgaven, David Norheim, og firmaet han representerer, Computas

[1] http://www.fixmystreet.com/

AS.

Borgerkanalen er en tjeneste, lignende og inspirert av FixMyStreet.com, der brukere skal kunne opprette såkalte borgerkanalsaker basert på informasjon om sakens kategoritilknytning og lokalitet. Borgerkanalen utvikles på en plattform for MHE-er (IOS) og nyttiggjør seg av verktøyene stilt til rådighet av denne. I motsetning til FixMyStreet, som ikke er spesielt utviklet for MHE-er, lar Borgerkanalen enheten den benyttes på sin posisjon brukes som sted for opprettelse av en sak. Posisjonen avgjøres ved hjelp av enhetens verktøy for posisjonering. En MHEs eventuelle innebygde kamera kan videre benyttes for å tilby en lettvinn mulighet å legge ved saken bildedokumentasjon på. En borgerkanalsak rapporteres automatisk av applikasjonen til rett myndighet basert på posisjon og en brukervalgt kategori.

Semantiske vevteknologier er et av kompetanse- og satningsområdene til Computas AS og også sentralt for Borgerkanalprosjektet. Alle borgerkanalsakene beskrives derfor med RDF og lagres på Computas Linked Open Data Server (se 5.3.2 på side 65). Lesing og manipulasjon av data skjer gjennom et SPARQL-endepunkt på eller med HTTP-forespørsler til denne RESTfulle serveren.

Å få Borgerkanalen tilgjengelig på andre plattformer enn IOS som selve Borgerkanalen-applikasjonen skal utvikles til, er ønskelig. Derfor er det implementert støtte for borgerkanalfunksjonalitet i Semantikart. Semantikart kommuniserer med Computas Linked Open Data Server og har gjennom denne tilgang til å lese og opprette nye borgerkanalsaker. Denne kommunikasjonen beskrives nærmere i 5.2.1 på side 58.

## 4.4 Den semantiske veven på mobile, håndholdte enheter

Det finnes egne RDF-biblioteker for arbeid med semantisk vev-teknologier i enkelte programmeringspråk, slik som Jena og Redland. Disse bibliotekene lar utviklere benytte seg av forhåndsdefinerte metoder som tilbyr nyttig funksjonalitet mot den semantiske veven, f.eks. enkel oppkobling mot SPARQL-endepunkter, behandling av dataobjekter som RDF-tripler i programmeringsspråket og resonnering over slike. Støtten for bruk av slike RDF-biblioteker forsvinner imidlertid ved programmering mot enkelte hjemlige MHE-plattformer. Blant annet kan ikke Jena-bibliotekene brukes i et Android-prosjekt til tross for at Android støtter Java som utviklingsspråk og Jena er laget for Java da den virtuelle maskinen Android benytter seg av,

Dalvik, ikke støtter hele Javas klassebibliotek, blant annet flere biblioteker Jena avhenger av. Derfor må programmering mot den semantiske veven implementeres "på nytt" av utvikleren ved bruk av disse plattformene. For eksempel må en egen metode lages for oppkobling mot SPARQL-endepunkter.

SPARQL-endepunkter benytter seg av HTTP for datakommunikasjon (se 3.4 på side 34). MHE-er har verktøy som muliggjør tilkobling til Internett. Siden HTTP er standardprotokollen for kommunikasjon over nettet, støtter følgelig MHE-er bruk av denne. Dermed kan en MHE kommunisere forholdsvis enkelt med SPARQL-endepunkter, uten bruk av innebygd støtte for dette gjennom biblioteker som Jena eller Redland. Dette gjelder også kommunikasjon gjennom MHE mot RESTfulle systemer, siden HTTP er kjernen her (se 3.5 på side 40). RESTfulle systemer er derfor godt egnet i samspill med applikasjoner på mobile, håndholdte enheter. Biblioteker som er rettet mot HTTP-kommunikasjon finnes støttet i Android og på de fleste andre programmeringsplattformer.

En spesifikk ting som svarer til et subjekt på den semantiske veven, er ofte tilknyttet en bestemt type eller kategori. En vanlig konvensjon er å bruke ressursen "rdf:type" som predikat i et trippel der en slik ting står som subjekt og hvor objektet svarer til typen subjektet er av, for eksempel: `byggesak:201000015 rdf:type computas:Byggesak .` SPARQL-spørringer som henter data av en bestemt type kan så konstrueres. Ved å tilpasse spørringer basert på brukernes valg eller søkeord, kan applikasjoner som henter inn data av en eller flere brukerbestemte typer lages. Dette kan utnyttes i kartapplikasjoner i form av mulighet for å vise data som har en bestemt typetilhørighet på kart, for eksempel alle hoteller, eller parkeringshus. Denne funksjonaliteten implementeres av Semantikart for å kategorisere ulike typer data som brukeren kan velge å få vist eller ikke.

Mange typer beskrives som undertyper av noe annet, f.eks: `dbp-owl:Stadium rdfs:subClassOf dbp-owl:Building .` Når en ressurs har en slik underkategori som sin type, f.eks: `dbpedia:Ullevaal_Stadion rdf:type dbpedia-owl:Stadium`, kan denne ressursen gjennom resonnering (transitivitet) avledes til også å være av typen `dbp-owl:Building`. Dermed kan vi med resonnering inkludere alle data med typen `dbp-owl:Stadium` når vi snakker om `dbp-owl:Building`. Uten resonnering vil kun dataene eksplisitt angitt med typen `dbp-owl:Building` være inkludert.

Dersom det er ønsket å la resonnering skje i en applikasjon, må dette implementeres av uvikleren selv ved manglende støtte fra RDF-biblioteker. Dette må gjøres gjennom logikk på en egnet datastruktur, noe som kan være krevende. Siden Semantikart er utviklet på Android, som mangler støtte for Jena og dermed innebygde muligheter for resonnering,

støttes i utgangspunktet ikke transitivitet i denne applikasjonen. MHE-er har begrenset maskinvareytelse, mens resonnering gjerne er en relativt ytelsesintensiv operasjon. Derfor er det trolig i de fleste tilfeller ikke hensiktsmessig å la resonnering skje på slike enheter uansett.

SPARQL tilbyr funksjonalitet for avgrensning av antallet matcher en spørring skal gi. Dette kan komme til nytte ved spørringer fra en MHE hvor ytelse og minne kan være begrenset. MHE-ens ytelse kan være såpass lav at den ikke takler å motta mye data på en gang eller visualisere mye data som grafiske elementer på for eksempel kart. SPARQL-spørringer som begrenses med nøkkelordet LIMIT, kan i slike sammenhenger brukes for å sikre at spørringene ikke returner uønsket mye data og dermed sikre mindre påkrevd minnebruk.

Siden antallet åpne, geolokaliserte data er enormt, må Semantikart gjøre visse begrensninger på hvilke slike den skal vise. Hvis ikke, sprenges kapasiteten til enheten den kjøres på. En måte å gjøre dette på er, som nevnt ovenfor, med LIMIT i SPARQL. LIMIT gir et avgrenset antall matcher, men mengden matcher som returneres er vilkårlig. Semantikart vil i første omgang tenkes brukt for å vise data som er i geografisk nærhet til brukeren. Fordi det er mulig at alle de vilkårlig utvalgte geolokaliserte dataene er plassert fjernt fra brukerens posisjon, er det lite hensiktsmessig å avgrense dataene på denne måten. Mer naturlig er derfor en begrensning i form av et FILTER i SPARQL, der datamengden avgrenses med utgangspunkt i dataenes plassering. På denne måten kan man be om å kun få returnert matcher med data som ligger innen en viss avstand fra brukerens posisjon. Et slikt FILTER kan kombineres med LIMIT for å, om nødvendig, begrense antallet treff ytterligere. Problemstillingen rundt hvordan slik innhenting av åpne data som har en geografisk nærhet til brukerens posisjon kan gjøres i SPARQL, adresseres i 6.1 på side 71.

# Kapittel 5

# Oppbygging av systemet

Dette kapittelet starter med en systembeskrivelse som skisserer systemets funksjonalitet i grove trekk. Videre følger en overordnet systemarkitektur som viser hvilke komponenter systemet består av. Herunder redegjøres det for kommunikasjonen som foregår mellom systemdelene. Kapittelet avsluttes med en grundigere gjennomgang av systemets ulike deler. Komponentene som utgjør applikasjonen Semantikart, som har blitt utviklet i mastergradsarbeidet, vektlegges spesielt her.

## 5.1 Systembeskrivelse

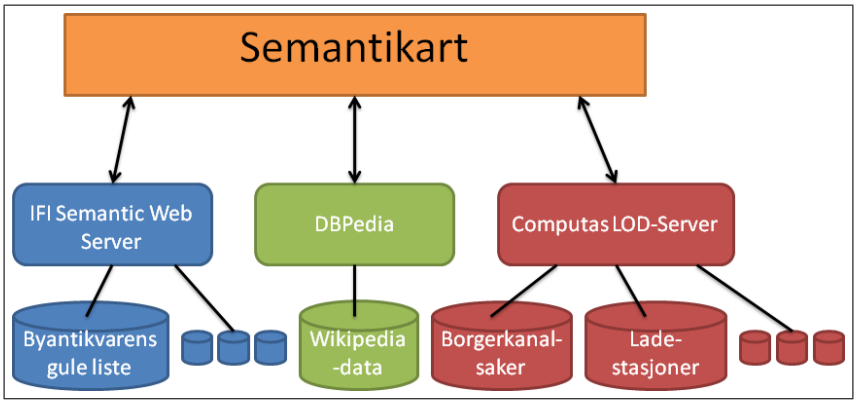
Semantikart er en mobilapplikasjon utviklet for Android-plattformen og tilgjengeliggjort[1] gjennom distribusjonsportalen Android Market. Applikasjonen er nedlastbar, vederlagsfritt, for alle Android-baserte MHE-er.

Semantikart tilbyr brukeren data visualisert som punkter på et kart eller i en sorterbar stedsliste. Dataene stammer fra en rekke offentlig tilgjengelige datakilder, blant annet Oslo Byantikvars gule liste over verneverdige bygg (se 5.3.3 på side 66) og en kilde som inneholder data for alle EL-billadestasjoner i Norge (se 5.3.3 på side 69). Felles for de tilgjengelige dataene er at de alle kan knyttes til en geografisk posisjon.

Brukeren kan selv velge hvilke datatyper han ønsker å ha tilgjengelig i applikasjonen og hvor stort geografisk område han ønsker å hente inn data fra.

Det er også mulig for brukeren å legge til egne data gjennom applikasjonen.

[1] https://market.android.com/details?id=no.uio.ifi.at.jenskm.semantikart




Figur 5.1: Overordnet systemarkitektur

Dette gjøres ved å legge til en ny "borgerkanalsak". En borgerkanalsak er en innrapportering om en feil eller mangel som brukeren ønsker at skal tas hånd om av riktig myndighet. Les mer om Borgerkanalen i 4.3 på side 52.

En brukerveiledning som beskriver Semantikarts funksjonalitet i større detalj finnes i masterrapportens vedlegg, A.3 på side 100.

## 5.2 Systemarkitektur

Semantikart henter data fra flere ulike dataservere. Disse serverene inneholder et eller flere datasett fra en eller flere kilder. Det overordnede systemet består altså av tre typer deler: mobilapplikasjonen Semantikart, som utgjør systemets klient og er utviklet som del av mastergradsarbeidet, dataserverene klienten benytter seg av, samt datasettene som er lagret på serverne.

Figur 5.1 viser koblingene mellom applikasjonen Semantikart og ulike dataservere som hver inneholder data fra én eller flere forskjellige datakilder.

### 5.2.1 Kommunikasjon mellom systemdelene

Kommunikasjonen i systemet foregår mellom applikasjonen og serverne via HTTP. De to viktigste tilfellene av slik kommunikasjon skjer når stedsressursene skal hentes inn i applikasjonen og når en ny borgerkanalsak lages i applikasjonen.

I det første tilfellet, når dataene skal lastes inn, er vi i en situasjon der data skal hentes fra én eller flere servere med hvert sitt SPARQL-endepunkt.

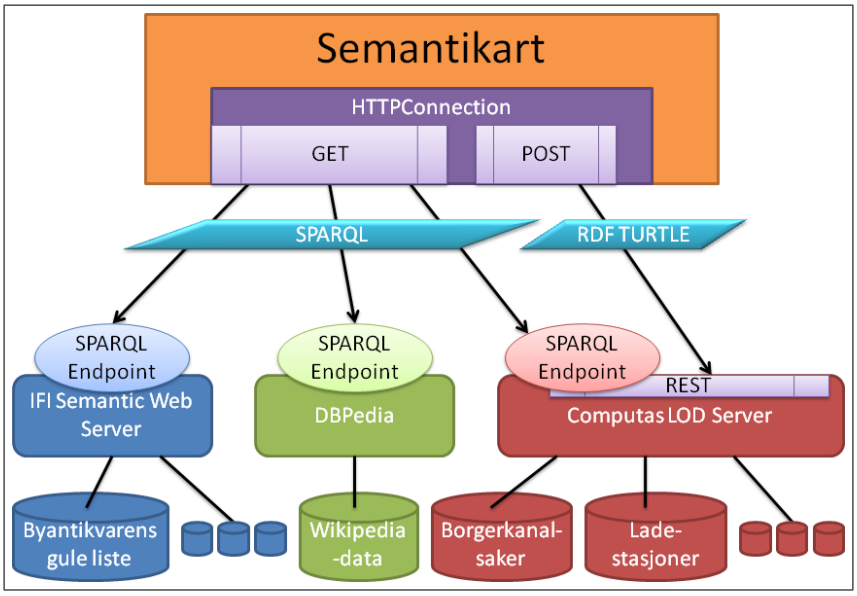
Denne datainnhentingen foregår ved at det lages en HTTP-kobling for hver av datatypene det skal hentes data fra. Koblingene åpnes til det endepunktet som inneholder den aktuelle datatypen. Dette skjer ved at deres adresse settes til å være endepunktets URL konkatenert med en URL-enkodet[2] representasjon av en SPARQL-spørring. SPARQL-spørringene vil typisk være av typen "gi meg alle data om X, hvor X er av datatype D". Dette blir en HTTP-GET-forespørsel til serveren hvor et svar returnes i form av data på RDF/XML-format. Disse dataene tolkes og lagres så av applikasjonen.

I det andre tilfellet, når en ny borgerkanalsak lages i applikasjonen, skapes det nye data som så lagres på en server slik at de blir gjort synlige for andre brukere og ikke går tapt når applikasjonen avsluttes. Slik lagring er det tilrettelagt for på Computas LOD-server. Denne serveren kan klienter lagre data på gjennom HTTP-POST. HTTP-POST kan programmeres til å utføre nesten enhver type handling på serversiden, men i RESTfulle systemer — slik som Computas LOD er — brukes den vanligvis til å lage eller utvide ressurser [43]. Se 3.5 på side 40 for mer informasjon om RESTfulle systemer og HTTP-forespørsler.

I en HTTP-POST-forespørsel til Computas LOD Server lages en HTTP-kobling med et spesifisert HTTP-aksepthode (se 3.5 på side 43) som angir hva slags MIME-type som skal være input til koblingen. Computas har programmert serveren til å kunne ta imot data på visse formater og konvertere disse til RDF representert på serveren. En av disse støttede MIME-typene er RDF-serialiseringen Turtle. Dette formatet spesifiseres som aksepthode i applikasjonens kobling til LOD-serveren. En Turtle-formatert tekststreng sendes så fra applikasjonen når koblingen er åpnet. Denne Turtle-strengen konverteres av serveren og trippelmengden strengen representerer legges inn i serverens database. Slik får endepunktet nye data å tilby.

Det er også et tredje tilfelle av kommunikasjon som skjer via HTTP fra applikasjonen. Noen av stedsressursene har et bilde tilknyttet seg. Dette er bilder som er knyttet til det ressursen beskriver. For eksempel et bilde av "Vigelandsparken" knyttet til ressursen som representerer stedet med samme navn, eller et bilde av et hull i veien ved "Akersgata" knyttet til en Borgerkanalsak. Det ville være plasskrevende å laste inn alle slike bilder i applikasjonens internminne. Derfor lagres ikke selve bildene i applikasjonen, men derimot URI-er som representer adresser til hvor bildene finnes lagret på nett. Når et bilde som er knyttet til en stedsressurs skal vises i applikasjonen, lastes det aktuelle bildet inn gjennom en HTTP-GET fra en slik URI. På

[2] I URI-er finnes det flere tegn som har reservert betydning. En tekststreng som inneholder slike tegn vil tolkes på en annen måte enn den tiltenkte dersom den benyttes i sin opprinnelige form som URI. For å unngå dette, kan tekstrengen URL-enkodes. Dette vil si at de reserverte tegnene byttes ut med en sammensetning av symboler som i URI-er entydig representerer disse tegnene.




Figur 5.2: Spørringer sendes over HTTP-GET til endepunktene. RDF-tripler i TURTLE-format sendes over HTTP-POST til RESTful tjeneste på Computas LOD-Server hvor de lagres som data.

denne måten kan bilder presenteres effektivt til brukeren, forutsatt at nett-tilkoblingen er tilfredstillende, uten at de alle må lagres internt og dermed oppta minneplass.

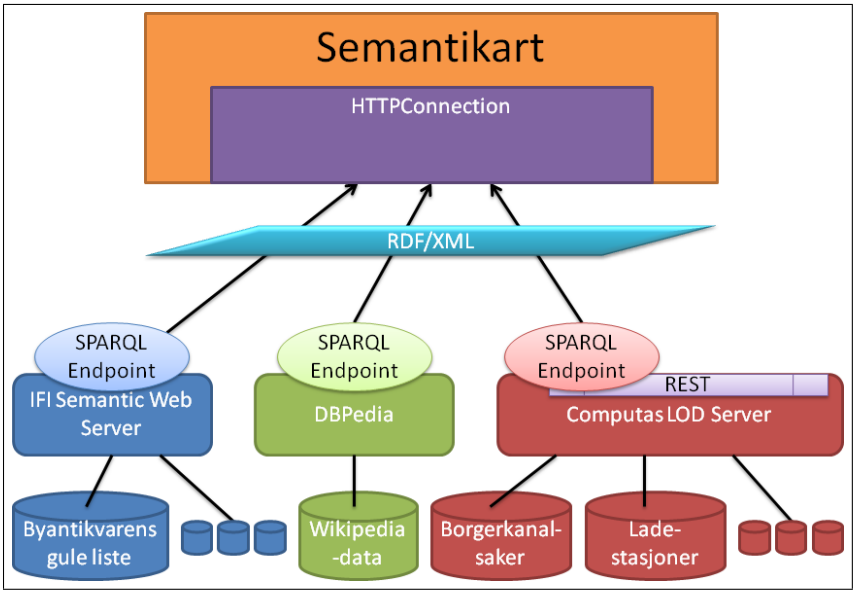
## 5.3 Beskrivelse av systemdelene

### 5.3.1 Applikasjonen

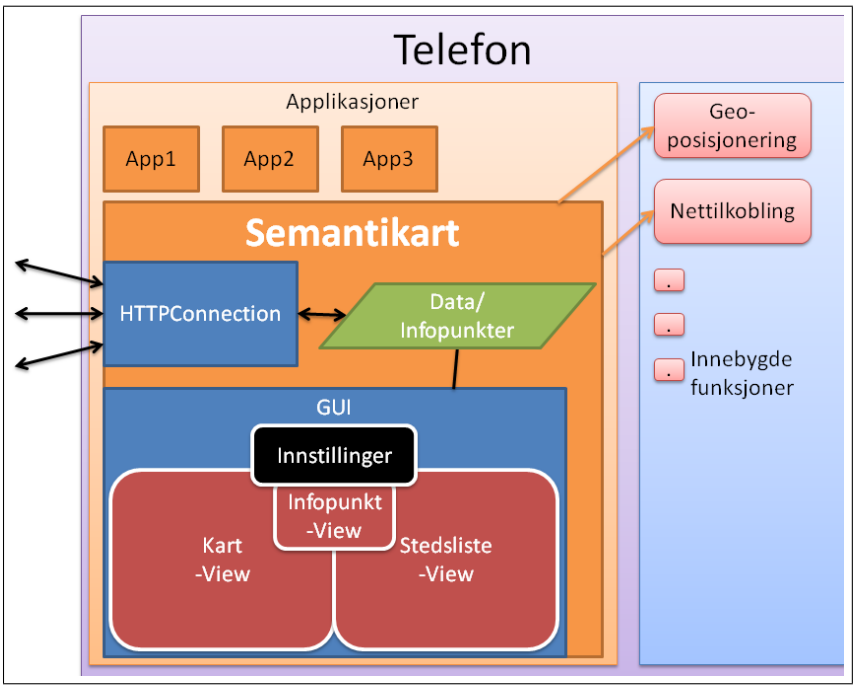
Den viktigste delen av systemet er selve applikasjonen, Semantikart. Denne består av flere ulike interne komponenter og er i tillegg avhengig av å bruke verktøy som finnes tilgjengelige på den mobile, håndholdte enheten den kjører på.

#### Data

Applikasjonens data blir hentet fra serverne eller skapt av brukeren og må representeres i en egnet struktur. Det finnes to hovedtyper av data i applikasjonen: steder og infopunkter. Et *sted* er noe som har en geografisk




Figur 5.3: Svaret på spørringene mottas av applikasjonen gjennom HTTP fra endepunktene på RDF/XML-format.




Figur 5.4: Figuren viser Semantikarts interne struktur i grove trekk, applikasjonens plass blant andre mobilapplikasjoner på telefonen, samt dens tilgang til noen av telefonens innebygde verktøy.

lokasjon og dermed kan kobles til et enkelt koordinatpunkt. Et *infopunkt* beskriver noe som finnes på et sted, en ressurs. Infopunktet er av en *datatype* gitt av datasettet ressursen er hentet fra, eller eventuelt et subsett av et slikt — noe som er tilfelle i datasettet TellUs, ytterligere beskrevet i 5.3.3 på side 68. Videre har hvert infopunkt en tittel som svarer til navnet på ressursen den beskriver og eventuelt en beskrivelsestekst og/eller adresse til et tilknyttet bilde.

Det skilles mellom steder og infopunkter i Semantikart fordi det finnes steder som inneholder flere forskjellige ting, flere infopunkter. For eksempel finnes infopunktene "Frogner Hovedgård" og "Oslo Bymuseum", begge fra datakilden DBPedia[3], på samme sted. Oslo Bymuseum ligger i lokalet Frogner Hovedgård. Derfor er disse ressursene beskrevet med samme geografiske koordinatpar.

Den samme tingen kan dessuten være beskrevet i flere forskjellige datasett. I slike tilfeller vil også ulike infopunkter være knyttet til samme sted. De fleste steder vil dog kun være tilknyttet ett eneste infopunkt.

### Brukergrensesnitt

Applikasjonens brukergrensesnitt tilbyr to ulike innsynsvinkler til dataene: et kartperspektiv og en stedsliste.

Kartet viser dataene som punkter der de hører hjemme geografisk. Steder vises med ulike symboler på kart og i stedslisten avhengig av hva slags datatyper infopunktene den inneholder har. Eksempelvis vil et sted som kun inneholder infopunkter av én datatype vises med en knappenål som har et design som indikerer hvilken datatype den representerer. Knappenålspissen stikker ned i kartet på det geografiske stedet, koordinatet, som stedet er knyttet til. Kartfunksjonaliteten systemet benytter leveres av Google Maps, ytterligere beskrevet i 4.2.

Stedslisten tilbyr brukeren en liste over infopunktene som i tillegg til å vise infopunktets navn viser avstanden fra brukerens posisjon til stedet infopunktet er tilknyttet. Denne listen er sorterbar etter ulike kriterier: alfabetisk fra a—å eller å—a, eller etter avstand fra bruker; stigende eller synkende.

Videre finnes det i Semantikart noen andre mindre perspektiver, blant annet et detaljert infopunktperspektiv som viser alle dataene om et infopunkt. Dette perspektivet vil vises når man velger et sted på kartet eller i stedslisten. Dersom stedet inneholder flere infopunkter, får brukeren frem en liste over

[3] Se 5.3.2 på side 65

disse hvor hun kan velge en av dem for å få fram det valgte infopunktets detaljerte perspektiv.

### HTTP-kobling

Applikasjonen inneholder en komponent som er ansvarlig for å kommunisere med serverene. Dette skjer, som beskrevet, ved hjelp av HTTP. Denne komponenten er aktiv når data skal hentes inn i programmet og når data skal sendes tilbake til server. I det første tilfellet konstrueres spørringer som ber om alle dataene for hver av datatypene brukeren vil se innenfor en bestemt avstand fra brukerens posisjon. I det andre tilfellet skapes en tekststreng på RDF-serialiseringen Turtle som representerer ressursen som skal lagres på serveren.

Applikasjonen er avhengig av å tolke dataene den mottar fra serveren i RDF/XML-format. Disse dataene må analyseres og ressursene de beskriver lagres i applikasjonens datastruktur. Hver gang en ny ressurs blir hentet fra en av serverene, må dette oppdateres grafisk i applikasjonen. "HTTP-Connection"-komponenten samarbeider derfor tett med applikasjonens brukergrensesnitt. For at dette skal kunne foregå uten at GUI-en blir stående og vente på datakommunikasjonskomponenten og omvendt, har disse fått hver sin tråd å jobbe på.

### Semantikarts bruk av telefonens innebygde verktøy

Semantikart er en av flere applikasjoner som kan være installert på en mobil, håndholdt enhet. Slike enheter leveres med en rekke avanserte funksjoner og verktøy, blant annet kamera, nettilkobling og blåtann. Applikasjoner kan få tilgang til enhetenes innebygde verktøy dersom dette er nødvendig for deler av deres funksjonalitet. Semantikart har behov for funksjonalitet fra flere slike innebygde verktøy.

En funksjon mange mobile, håndholdte enheter har, er muligheten til å avgjøre dens geografiske posisjon med en viss presisjon. Semantikart er helt avhengig av slik funksjonalitet for å kunne vise brukerens posisjon på kart og å presentere dataressurser som har en geografisk tilknytning med nærhet til denne posisjonen. Stadfesting av enhetens posisjon skjer vanligvis på én av to følgende måter:

1. Enheten har innebygd *Global Posistion System* (GPS) som kan brukes til å finne enhetens posisjon. Dette er den mest presise av de to metodene, men også den som krever mest strøm.

2. *Trådløse nettverk* eller mobile nettverk brukes til å bestemme posisjonen ved hjelp av trianguleringer.

Semantikart er avhengig av internettilgang for å kunne kommunisere og utveksle data med servere. Nettilgang kan på MHE-er sikres gjennom mobilnettet eller trådløst nettverk (WiFi). Støtte for mobilt nettverk finnes tilgjengelig gjennom nesten alle MHE-er i forskjellige utgaver med ulik hastighet. Mobile nettverk er utbredt og tilgjengelige de fleste steder i Norge. Imidlertid kan det være kostbart å laste ned mye data over mobile nettverk med enkelte telefonabbonementer.

De fleste nyere mobiltelefoner leveres også med mulighet til å koble seg til trådløse nettverk (WiFi). Nettverkshastigheten på slike trådløse nett varierer med abonnementstyper og leverandører, men er i de fleste tilfeller langt raskere enn på de mobile nettverkene. Ulempen med WiFi er at de sluker en del batterilevetid og at de kun er tilgjengelige innenfor mindre områder.

Når man med Semantikart lager en ny Borgerkanalsak, har man mulighet til å knytte et bilde til disse. Slike bilder kan legges til enten ved å velge et bilde fra telefonens harddisk eller ved å ta et nytt bilde med MHE-ens kamera. Å ta bilde med kamera avhenger av at telefonen applikasjonen kjører på har et innebygd kamera. Dersom den har dette, vil applikasjonen kunne starte en tjeneste som åpner mobilens bildetakningstjeneste og returnerer et bilde til applikasjonen når et slikt blir tatt. Semantikart krever tilgang til å hente data fra telefonens harddisk samt fra telefonens bildetakningsverktøy på telefoner som støtter dette.

### 5.3.2 Servere og deres endepunkter

Systemet benytter seg av tre ulike servere. Disse serverene har alle hvert sitt SPARQL-endepunkt som Semantikart og andre klienter kan koble seg opp mot for å spørre mot serverenes datamengde. De forskjellige serverene har endepunkter med litt ulike back-end-løsninger. De ulike back-end-løsningene støtter alle SPARQL-protokollen og SPARQL RDF-spørrespråket og tilbyr derfor essensielt den samme funksjonaliteten.

#### Institutt for informatikks Semantic Web Server

Institutt for informatikk ved Universitetet i Oslo har satt opp en egen server for åpne data og lagring av RDF-tripler: IFI Semantic Web Server (SWN). Denne serveren er satt opp i forbindelse med Semicolon-prosjektet

og inneholder blant annet data fra Kommunekatalogen, Byantikvarens gule liste og Partifinansieringen.

Serveren har et endepunkt som er eksternt tilgjengelig på adressen: http://sws.ifi.uio.no/data. Endepunktet bruker HTTP-motoren Joseki som back-end.

#### Computas Linked Open Data Server

Computas AS satser på semantisk teknologi og har laget en server, "Computas Linked Open Data Server", som trolig inneholder den største samlingen av RDF-data tilgjengeliggjort i Norge. Serveren er bygget opp med Oracle 11g Enterprise Edition med Joseki som SPARQL-endpoint. Endepunktet er tilgjengelig på: http://opendata.computas.no:7011/joseki/

Computas LOD Server er RESTfull. Data kan aksesseres og legges til med HTTP-kall på formater angitt i HTTP-hodet fra ulike klienter. I 5.2.1 på side 58 beskrives hvordan kommunikasjonen mellom Semantikart og det RESTfulle systemet foregår.

#### DBpedia

DBpedia kan sees på som semantisk vev-implementasjonen av det brukergenerert-innholdsbaserte nettleksikonet Wikipedia. DBpedia tar strukturert informasjon fra Wikipedia-artikler og beskriver disse med RDF.

Det finnes et offentlig SPARQL-endepunkt som baserer seg på DBpedias data. Endepunktet bruker OpenLink Virtuoso som back-end og finnes tilgjengelig på adressen: http://dbpedia.org/sparql.

### 5.3.3 Datakilder

Alle datakildene var opprinnelig tilgjengeliggjort i formater uten bruk av RDF og måtte dermed konverteres til et RDF-format før de kunne legges inn på serverene. De ulike kildene varierer i hvor mye, og hva slags data som beskrives. Triplene som beskriver data av forskjellige kilder har derfor ulike predikater og struktur. Felles for dem og minimumskravet for at de skal kunne vises i applikasjonen, er at de må inneholde et koordinat som angir bredde- og lengdegrad, og noe som kan brukes som tittel. Eksempelinstanser for hvert av disse datasettene vises nedenfor, men kun med de triplene som er interessante for- og som vil brukes i Semantikart

Tillegget, A.2.4 på side 95, inneholder beskrivelser av spørringer som henter ut nødvendig informasjon for hver av datakildene.

### Borgerkanalen

Dette er datasettet som beskriver innrapporterte saker til Borgerkanalen. Datasettet er tilgjengelig på Computas LOD Servers endepunkt. Eksempel på borgerkanalsakinstans:

```
<http://opendata.computas.no/data/borgerkanalen/sak/1139483192>
      a  <http://opendata.computas.no/voc/borgerkanalen/Sak> ;
      <http://purl.org/dc/terms/description>
              "Farlige istapper henger ned fra taket";
      <http://purl.org/dc/terms/title>
              "Skumle istapper" ;
      <http://purl.org/dc/terms/location>
              [] <http://www.w3.org/2003/01/geo/wgs84_pos#long>
      "10.76105" ;
     <http://www.w3.org/2003/01/geo/wgs84_pos#lat>
     "59.928977" .
```

### Byantikvarens gule liste

Dette datasettet inneholder data om verneverdige bygg i Oslo kommune. Datasettet er en RDF-konvertering av listen som månedlig legges ut på Oslo byantikvars nettside[4]. Eksempelinstans:

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix byantikvar: <http://www.byantikvaren.oslo.kommune.no/> .

[] byantikvar:byggeaar "ca. 1870-1900"^^xsd:string ;
   byantikvar:sted "Seilduksgata"^^xsd:string ;
   byantikvar:vernedato "20060428"^^xsd:string ;
   byantikvar:vernestatus "Fredet, Kml § 20"^^xsd:string ;
   <http://www.euref.eu/etrs89/east>  598531 ;
   <http://www.euref.eu/etrs89/north> 6644460 ;
   <http://www.statkart.no/GAB-registeret.ipsgatenummer> 28 .
```

[4] http://www.byantikvaren.oslo.kommune.no/gul_liste/article42663-934.html

### Byggesaker

Dette datasettet omhandler byggesaker fra Oslo Kommune. Computas LOD Server tilgjengeliggjør disse dataene gjennom sitt endepunkt. Eksempelinstans:

```
@prefix : <http://opendata.computas.no/voc/oslo.kommune.no/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix geo: <http://www.geonames.org/ontology#> .
@prefix wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix dct: <http://purl.org/dc/terms/> .

<http://opendata.computas.no/data/oslo.kommune.no/byggesak/215>
      a       :Byggesak ;
      :bydel  "8 - NORDRE AKER" ;
      :gnrbnr "44/230" ;
      :saksnr "201000015" ;
      :streetname "Bukken bruses vei" ;
      :streetnumber "3" ;
      dct:description """Saksdato: 04-01-2010,
        Sist endret: 18-05-2010, Ansvarlig: TDS/Kjetil Enger" ;
      dct:title """201000015 Bukken
        Bruses vei 3 - Seksjonering""" ;
      geo:location <http://linkedgeodata.org/triplify/way461> .

<http://linkedgeodata.org/triplify/way461>
      dct:title "Bukken Bruses vei, Majorstuen,
                 Oslo, 0864, Norway" ;
      wgs84_pos:lat "59.9446193790289" ;
      wgs84_pos:long "10.7235395418179" .
```

### DBpedia

I dette datasettet finnes strukturerte data fra Wikipedia, slik beskrevet i avsnittet om endepunktet settet ligger på, 5.3.2 på side 65. Eksempelinstans:

```
@prefix geo:   <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dbpedia: <http://dbpedia.org/ontology/> .
@prefix : <http://dbpedia.org/resource/> .

:Havnelageret rdfs:label "Havnelageret"@no ;
```

```
            rdfs:label "Havnelageret"@en ;
            geo:lat     59.90760803222656 ;
      geo:long    10.74666690826416 ;
            dbpedia:abstract "Havnelageret er et..." ;
      dbpedia:thumbnail
                  <http://upload.wikimedia.org/wikipedia/
                  commons/thumb/c/cc/Havnelageret_Oslo.
                  jpg/200px-Havnelageret_Oslo.jpg> .
```

### TellUs

Reiselivsinformasjon fra Norge er det sentrale i dette datasettet tilgjengelig gjennom Computas' SPARQL-endepunkt. Overnattingsmuligheter, bespisning og arrangementer finnes beskrevet her for hele Norge. Datasettet inkluderer også prisinformasjon, en type informasjon som ikke involveres i Semantikart.

"dc:subject"-predikatet brukes i dette datasettet for å angi intern kategoritilhørighet, med koder som sier om en ressurs er eksempelvis et spisested eller en butikk. Semantikart utnytter dette ved å stille flere ulike spørringer til dette datasettet som ber om data av ulike kategorier. Flere av datatypene i Semantikart kommer dermed fra samme datasett: dette. Eksempel på instans i datasettet:

```
@prefix id: <http://opendata.computas.no/data/tellus/produkt/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tellus: <http://opendata.computas.no/voc/tellus/> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#> .
@prefix dbpedia-owl: <http://dbpedia.org/ontology/> .
@prefix cat: <http://opendata.computas.no/data/tellus-categories/>

id:82
  a tellus:Product ;
  dct:title "Bergen Ymca Hostel" ;
  dct:description   """Bergen Ymca Hostel overnatter..."
  dct:subject cat:1 ;
  dct:subject cat:1_809 ;
  dct:subject cat:1_262 ;
  dct:location [
     a dbpedia-owl:Place ;
     a geo:Point ;
     rdfs:label "Nedre Korskirkeallmenning 4" ;
```

```
    geo:long "5.326746" ;
    geo:lat "60.394493" ;
  ] ;
```

### Ladestasjoner

Datasettet er opprinnelig tilgjengeliggjort på en nettside[5] og finnes tilgjengelig i RDF-format på Computas' SPARQL-endepunkt. Datasettet inneholder ladestasjoner for elbiler i Norge.

```
@prefix geo: <http://www.geonames.org/ontology#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix : <http://opendata.computas.no/voc/ladestasjoner.no/> .

[] a :Ladestasjon ;
    dct:title "Akershus festning" ;
    geo:lat "10.73508" ;
    geo:long "59.90929" .
```

[5] http://www.ladestasjoner.no/2009/10/na-kan-du-ogsa-laste-ned-ladestasjoner.html

# Kapittel 6

# Utfordringer

I dette kapittelet beskrives noen av de implementasjonsmessige utfordringer som det var nødvendig å adressere i forbindelse med utviklingen av Semantikart. Utfordringene som beskrives her er knyttet til geometri og posisjonsberegning og henger til en viss grad sammen.

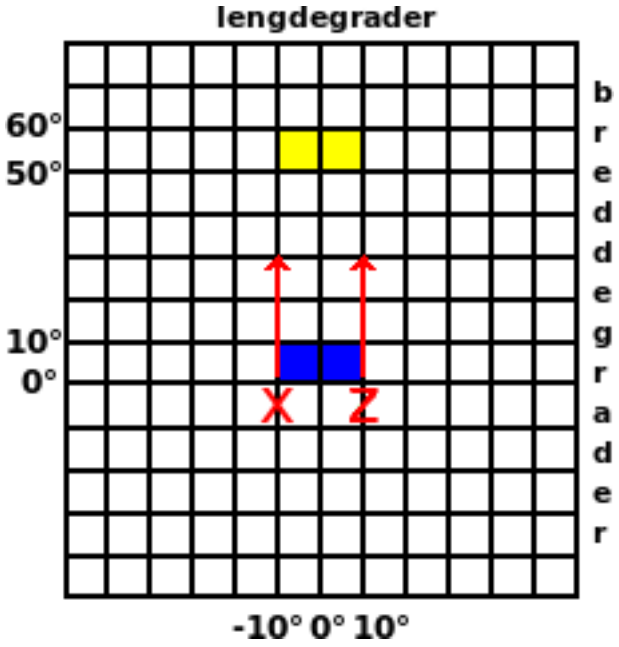
## 6.1 Geografisk avgrensning av data med SPARQL

En problemstilling som oppstår i forbindelse med innhenting av geolokaliserte data med SPARQL, er hvordan det geografiske området en spørring henter sine data fra kan avgrenses.

Brukerens og ressursenes posisjoner representeres alle som koordinatpar. Disse koordinatparene, bestående av lengde- og breddegrader, er derfor det som må brukes for å bestemme avstanden mellom posisjonene. Koordinatparene er, for dataene som skal hentes fra serverene, representert ved RDF-tripler som beskriver ressursens breddegrad og dens lengdegrad. Informasjon om brukerens bredde- og lengdegrad er tilgjengelig gjennom MHE-ens innebygde verktøy for geoposisjonering.

Avstanden fra et koordinat $(x, y)$ til et koordinat $(z, y)$ der $x$ og $z$ er lengdegrader og $y$ er en breddegrad, vil på en flate med rette akser være den samme uansett hvilken breddegrad $y$ svarer til. Se figur 6.1 på neste side. Siden jorda ikke er flat, men tilnærmet kuleformet[1], har den ikke

[1] At jordas form er kuleformet er ikke helt presist. Den er mer lignende en ellipsoide. Eller enda mer presist: har form som en flatklemt sfæroide. Feilmarginen av å anta kuleform når det kommer til å beregne avstanden mellom to koordinater er liten, generelt mindre enn 0,3 % [42]. Å anta kuleform er presist nok for de fleste anvendelser, inkludert Semantikart.

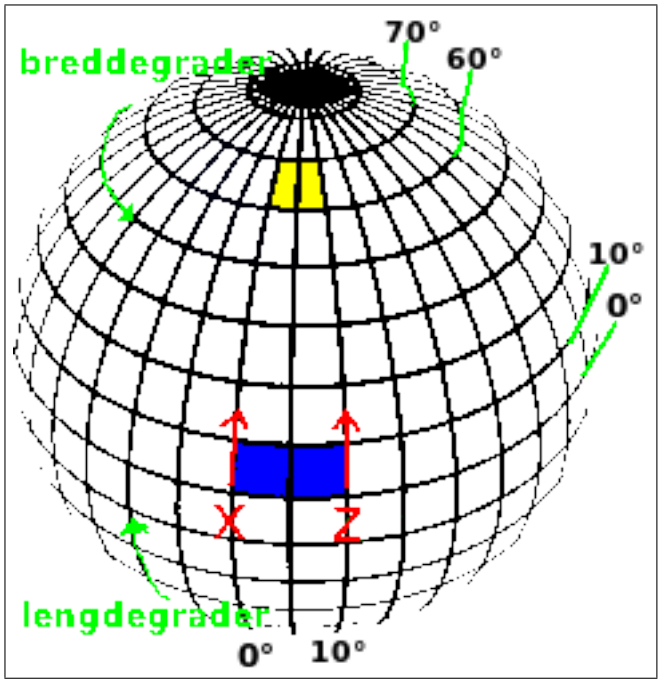



Figur 6.1: På en flate med rette akser vil avstanden mellom lengdegradene x og z være den samme uansett hvor på breddegradaksen vi befinner oss. Både høyden og bredden til det gulmarkerte området er derfor like stor som høyden og bredden til det blåmarkerte.

rette akser, men derimot buede. Avstanden mellom koordinatene $(x, y)$ og $(z, y)$ der $x$ og $z$ er lengdegrader og $y$ er en breddegrad, vil derfor på jorda variere med verdien til $y$. Avstanden mellom de to punktene vil dermed være lengst dersom $y$ er breddegraden til ekvator (0°) der jorda er tjukkest. Jo nærmere $y$ befinner seg en av jordas poler (±90°), jo mindre vil avstanden mellom $(x, y)$ og $(z, y)$ bli. Se figur 6.2 på neste side. Dersom $x = 0°$, $z = 10°$ og $y = 0°$, vil avstanden mellom $(x, y)$ og $(z, y)$ være på ca. 1111 kilometer. Hvis $y$ istedenfor er 80°, blir avstanden mellom dem omlag 193 kilometer. Én breddegrads høyde i meter er den tilnærmet den samme uansett hvilken lengdegrad den skjærer gjennom, mens én lengdegrads bredde i meter varierer med hvilken breddegrad den skjærer gjennom. For å regne ut avstanden mellom to geografiske koordinater på en kuleformet klode, finnes det standard algoritmer, eksempelvis "haversine-formelen" [42].

En spørring som ber om data fra ressurser som ligger maksimalt tre kilometer unna brukerens posisjon kan skisseres slik:

```
PREFIX wgs: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX dc:  <http://purl.org/dc/terms/>
```




Figur 6.2: Avstanden mellom lengdegradene x og z blir gradvis mindre når avstanden i antall breddegrader fra ekvator øker. Derfor blir også bredden til det gulmarkerte området mindre enn bredden til det blåmarkerte området.

```
PREFIX ex:  <http://example.com/SPARQL/functions#>

SELECT lengdegrad, breddegrad
WHERE {
      ?x dc:location ?location .
      ?location wgs:long ?lengdegrad ;
                wgs:lat ?breddegrad .

FILTER(ex:distance(?lengdegrad, ?breddegrad, 10.75, 59.95)
       <= 3000) .
}
```

For å få en spørring til kun å hente data som innfrir kravet om å ligge innenfor tre kilometers avstand fra brukeren, må spørringen inneholde et FILTER-kriterium som filtrerer vekk dataene som ikke tilfredstiller dette kravet. Et filtreringskriterium som sikrer dette er skissert ovenfor ved funksjonen "ex:distance". Ex:distance tar to sett med bredde- og lengdegrader og returnerer, ved hjelp av "haversine-formelen", avstanden mellom punktene disse settene representerer i meter.

SPARQL er et spørrespråk med et i utgangspunktet begrenset funksjonsbibliotek. SPARQL-grammatikken identifiserer noen operatorer, f.eks. $langMatches(A, B)$ og $isLiteral(A)$. Dessuten støttes noen av funksjonene definert i "XQuery 1.0 and XPath 2.0 Functions and Operators"[2] [31], slik som aritmetiske operatorer ($*$, $/$, $+$ og $-$). Funksjonalitet for blant annet kvadratrot, sinus og cosinus mangler dog. Dermed er det vanskelig å konstruere filtere som for eksempel bruker "haversine-formelen" i SPARQL. Imidlertid finnes det muligheter for å benytte seg av funksjoner fra eksterne funksjonsbiblioteker. Slike funksjoner er definert utenfor SPARQL og gjøres tilgjengelige for bruk i en SPARQL-spørring ved å angi en URI til funksjonens serveradresse. Ex:distance er en slik ekstern funksjon som er inneholdt på serveren example.com. Problemet med bruk av eksterne funksjoner er at de ikke nødvendigvis støttes av alle SPARQL-endepunkter.

Spørringene som ber om data i nærheten av brukeren, kan konstrueres på en måte som ikke benytter seg av eksterne funksjoner. For å få til dette må det settes filtreringskriterier i spørringen som avgrenser et område med koordinatgrader for hvor det skal hentes data fra. Dette vil være et område som spenner seg $l$ lengdegrader unna brukerposisjonens lengdegrad mot både øst og vest og $b$ breddegrader mot nord og sør. Slik blir området spørringen ber om data fra, heretter kalt geografisk avgrensningsområde, avgrenset av en figur med brukerens posisjon i midten, "en slags firkantet figur".

[2] http://www.w3.org/TR/xpath-functions/

```
PREFIX wgs: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX dc:  <http://purl.org/dc/terms/>
PREFIX ex:  <http://example.com/SPARQL/functions#>

SELECT lengdegrad, breddegrad
WHERE {
      ?x dc:location ?location .
      ?location wgs:long ?lengdegrad ;
                wgs:lat ?breddegrad .

FILTER(?lengdegrad <= brukerposlengdegrad + l)
FILTER(?lengdegrad >= brukerposlengdegrad - l)
FILTER(?breddegrad >= brukerposbreddegrad - b)
FILTER(?breddegrad <= brukerposbreddegrad + b)
}
```

En utfordring med en slik spørring er å finne verdier for $l$ og $b$ som gjør at det geografiske avgrensningsområdet blir av en størrelse som omslutter alle data som ligger nærmere brukerens posisjon enn den ønskede maksavstanden mellom bruker og dataressurser.

Hvis vi tar utgangspunkt i eksempelvis et koordinat som befinner seg i Skedsmo kommune i Akershus: 60° nord og 11° øst og bruker ovennevnte stedsavgrensningsmetode til å hente data innenfor ca. tre kilometer fra koordinatet, kan filtreringskriteriet skisseres slik:

```
FILTER(?lengdegrad <= 11 + l)
FILTER(?lengdegrad >= 11 - l)
FILTER(?breddegrad >= 60 - b)
FILTER(?breddegrad <= 60 + b)
```

For å sette det geografiske avgrensningsområdet til å omfange cirka tre kilometer rundt koordinatet, må terskelgradene $l$ og $b$ regnes ut.

Verdien til $l$ settes til å være lik antallet lengdegrader som tilsvarer tre kilometer på den 60. breddegraden. Formelen [44]

$$\frac{\pi}{180} M_r \cos\phi$$

hvor jordas meridionale radius[3], $M_r$, tilnærmet 6367,5 kilometer, brukes for å regne ut hvor bred én lengdegrad er på breddegraden $\phi$ i meter. Denne

[3]Jordas meridionale radius er en type gjennomsnittsradius som brukes om jorda. Hvordan denne regnes ut, er utenfor oppgavens omfang.

formelen er ikke helt nøyaktig på grunn av bruken av gjennomsnittsradius, men den er nøyaktig nok til Semantikarts formål. For å finne ut hvor bred én lengdegrad er når $\phi = 60$, blir regnestykket slik:

$$\frac{\pi}{180} \cdot 6367,5\,\text{km} \cdot \cos\phi \approx 55,57\,\text{km}$$

På den 60. breddegraden tilsvarer én lengdegrad ca. 55,57 km. Dette betyr at én kilometer tilsvarer ca. 0.018 grader: $\frac{1}{55,57} \approx 0,018$. Hvis vi ganger dette tallet med tre, finner vi ut at tre kilometer tilsvarer ca. 0.054 lengdegrader på den 60. breddegrad ($0,018 \cdot 3 = 0,054$). Dette svarets unøyaktighet er marginalt, med et avvik på ca 78 cm: $55,57 \cdot 0,054 = 3,00078$.

Det er enklere å regne ut verdien til $b$ siden avstanden mellom breddegradene i to ulike koordinater med samme lengdegrad er tilnærmet den samme uansett hvilken lengdegrad det er snakk om. Det er imidlertid en viss forskjell på grunn av jordas mildt ellipsolideaktige form. Avstanden mellom to etterfølgende breddegrader varierer fra ca 110,57 kilometer ved ekvator, til 111.7 ved polene [33]. For Semantikart er tilnæringen 111 km presis nok. Tre kilometer tilsvarer med en slik tilnærming cirka $0,027$ breddegrader; finner grader for én km ved: $\frac{1^\circ}{111} \approx 0.009^\circ$, ganger så dette med tre og får $0,009^\circ \cdot 3 = 0,27^\circ$.

Spørringen som henter data innenfor tre kilometers avstand fra en posisjon med koordinater 60° nord og 11° øst, får dermed følgende stedsavgrensningsfilter:

```
FILTER(?lengdegrad <= 11 + 0.054)
FILTER(?lengdegrad >= 11 - 0.054)
FILTER(?breddegrad >= 60 - 0.027)
FILTER(?breddegrad <= 60 + 0.027)
```

Siden avstanden mellom lengdegradene blir gradvis mindre når avstanden i antall breddegrader fra ekvator øker, vil filteret representere en figur som er bredest nederst ved "linjen" som dras på $60^\circ - 0,027^\circ$ og som blir stadig smalere fram til den øverste grensen ved $60^\circ + 0,027^\circ$, en "slags firkantet figur" med svakt buede sidekanter. Bueformen vil gjøres tydligere jo større figuren er og jo nærmere terskelbreddegradene er polene. Det gulmarkerte området i 6.2 på side 73 har dermed sterkt buede sidekanter. Når det kommer til avgrensning i Semantikart, vil forskjellen i bredde være marginal siden det er snakk om en liten differanse ($0,027^\circ \cdot 2$ i eksempelet) i breddegrader mellom den sydlige bunnlinjen og den nordlige topplinjen. Vi kan regne ut hvor brede bunn- og topplinjen blir i dette eksempelet ved å putte henholdsvis den sydligste breddegraden og den nordligste breddegraden inn som $\phi$ i formelen for utregning av én breddegrads lengde i meter. Tallene vi får som svar

kan brukes til å finne ut hvor mange meter spennet mellom lengdegradene $11° - 0,054°$ og $11° + 0,054°$ er på.

Én grad på den nordligste breddegraden ($\phi = 60° + 0,027° = 60,027°$) det skal hentes data fra, tilsvarer:

$$\frac{\pi}{180} \cdot 6367,5\,\text{km} \cdot \cos\phi \approx 55,52\,\text{km}$$

Spennet mellom lengdegradene $10,946°$ og $11,054°$ utgjør $0,108°$, som på den på den 60,0272. breddegrad tilsvarer cirka 5996 meter. ($55,52\,\text{km} \cdot 0,108 \approx 5,996\,\text{km}$)

Én grad på den sydligste breddegraden ($\phi = 60° - 0,027° = 59,973$) det skal hentes data fra, tilsvarer:

$$\frac{\pi}{180} \cdot 6367,5\,\text{km} \cdot \cos\phi \approx 55,61\,\text{km}$$

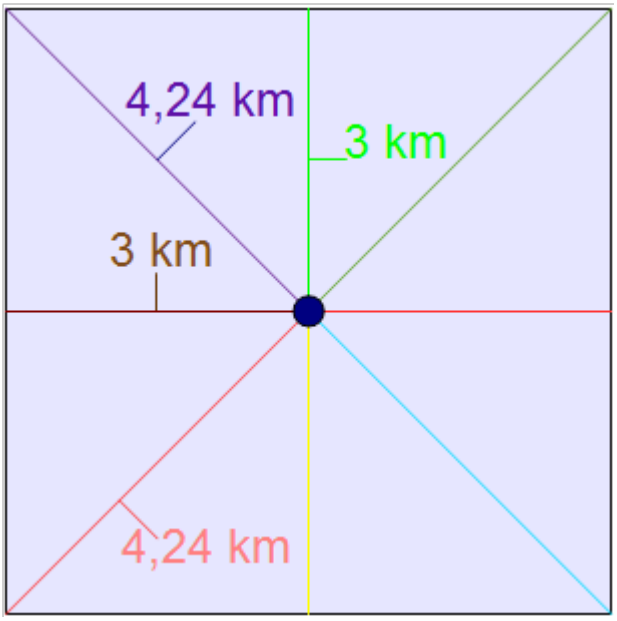
På den 59,973. breddegraden tilsvarer lengdegradenes avstand, $0,108°$, cirka 6006 meter. ($55,61\,\text{km} \cdot 0,108 \approx 6,006\,\text{km}$)

Forskjellen i bredde mellom den sydligste grensen for hvor spørringen henter data fra og den nordligste, er så liten (ca 10 meter i eksempelet) at det for Semantikart ikke har noen betydning. Det høyeste tallet det er mulig å sette stedsavgrensningsfilter til i Semantikart, er (ca.) 100 kilometer. Selv med dette tallet er forskjellene i bredde såpass små, at de for applikasjonen blir ubetydelige.

Som nevnt utgjør stedsavgrensningsfilteret en slags firkantet figur. Med eksempelets tre kilometer fra denne figurens senter (koordinatet som representerer brukerens posisjon) til hver av linjestykkene som utgjør filterfigurens grenser, får denne firkanten sider som er seks kilometer lange. Det antas for enkelhets skyld at sidene som representer lengdegradstersklene mot øst og vest, er helt rette, selv om de opplagt og som tidligere nevnt ikke er dette grunnet jordens form. Med denne antakelsen utgjør stedsavgrensningsfilteret en figur av kvadratisk form.

Et kvadrat kan opplagt ikke ha like lang avstand fra midtpunktet til alle punkter langs linjestykkene. Hadde den hatt det, ville den vært en perfekt sirkel. Det er derfor umulig å skape et geografisk avgrensningsområde som henter dataene med avstand maksimalt $z|z > 0$ fra brukerposisjonen med denne typen spørring.

For å regne ut avstanden mellom denne figurens nordvestlige og sørøstlige hjørner eller deres nordøstlige og sørvestlige hjørner, brukes formelen $\sqrt{2} \cdot s$ hvor $s$ er en av sidenes lengde. Avstanden mellom to sett av slike hjørner blir dermed i eksempelet $\sqrt{2} \cdot 6\,\text{km} \approx 8,485\,\text{km}$. Følgelig




Figur 6.3: Avstanden fra sentrum av et kvadrat med sider på seks kilometer til midten av hver side blir opplagt tre kilometer, avstanden til hjørnene blir lenger, ca 4.24 km.

blir avstanden mellom figurens midtpunkt og hvert av hjørnene cirka $8,485\,\text{km}/2 \approx 4,243\,\text{km}$. Stedsavgrensningsfilteret for eksempelposisjonen vil derfor akseptere ressurser som ligger ca. 4,24 kilometer unna posisjonen dersom de ligger i en av filtreringsfigurens hjørner, men kun ca. 3 kilometer for ressurser som ligger på samme bredde- eller lengdegrad som ressursen.

Semantikart er en applikasjon som i utgangspunktet kun er tiltenkt bruk i Norge. Dette avgrenser intervallene for aktuelle breddegrader ressurser blir hentet fra til å ligge mellom ca 58° og 71° nord for ekvator og mellom ca $4,6°$ og 31° øst. Dersom filteret skulle brukes for å avgrense ressurser med posisjon sør for ekvator eller for steder hvis posisjon omfanger overgangen fra 180° øst/vest til 0°, måtte det konstrueres med spesialtilfeller som kunne ta høyde for dette.

Semantikarts filtrering på avstand er ment å være til stede for å begrense datamengden som hentes inn i applikasjonen. Om denne filtreringen har sin grense akkurat tre kilometer fra brukerens posisjon, eller om den varierer mellom 3 og 4,5 kilometer fra brukeren avhengig av dens himmelretning i forhold til brukerposisjonen, er mindre viktig. Det som i applikasjonen er viktig, er det skal kunne velges mellom ulike avstander for stedsfiltrering, at spørretiden skal være så lav som mulig og at tilnærmet den samme måten å

bygge spørringer på skal kunne benyttes ved hvert av endepunktene.

Bruk av en ekstern funksjon for avgrensning har fordelen av at avgrensningen kan settes til å være helt nøyaktig $x$ km unna brukerposisjonen. En ulempe med bruken av en slik funksjon er at spørretiden vil være relativt høy da det for hver ressurs må gjøres en utregning av dens avstand fra brukerposisjonen. Dessuten støttes ikke slike funksjoner av alle SPARQL-endepunkter. Derfor er det for oppgaven ønskelig med en mer generell løsning på avstandsfiltreringsproblemet enn hva som oppnås med bruk av eksterne funksjoner for stedsavgrensning. Å sette det geografiske avgrensningsområdet med maksimum- og minimumsverdier for bredde- og lengdegradene er en metode som støttes av alle endepunkter og som har langt lavere spørretid enn ekstern funksjonsbruk. At denne metoden avgrenser dataene ulikt utifra hvilken himmelretning de er plassert på i forhold til brukeren, er i Semantikart uvesentlig. Sistnevnte alternativ velges derfor som metode for avstandsfiltrering i systemet.

## 6.2 Ulike geoformater

Det finnes mange standarder for å angi geolokalitet. De fleste dataene Semantikart kjører spørringer mot, bruker standarden "World Geodetic System" fra 1984 (WGS84) for å knytte en ressurs til en geografisk plassering. WGS84 er en av flere referansesystemer som fastsetter jordas midtpunkt, jordas form og hvor 0° bredde- og lengdegrader ligger. Geografiske punkter angis så som koordinatpar (breddegrad, lengdegrad) med utgangspunkt i dette. W3Cs Semantic Web Interest Group har definert et RDF-vokabular til bruk for å angi geolokalitet der WGS84 brukes som referansesystem [7]. Dette vokabularet, http://www.w3.org/2003/01/geo/wgs84_pos#, brukes for å angi geolokalitet i de fleste datasettene som er i bruk av Semantikart.

Et av datasettene Semantikart henter data fra, Oslo Byantikvars gule liste over verneverdige bygg, 5.3.3 på side 66, bruker en annen type referansesystem for geolokalitet, nemlig Universal Transverse Mercator-koordinatsystemet (UTM). Med UTM som referansesystem deles kloden inn i 60 nummererte soner. Et geografisk punkt på jordas nordlige halvkule angis i UTM-systemet som et koordinat (X,Y,Z). X (øst) er uavhengig for hver sone Z og har en verdi på 500 000 meter ved den sentrale meridianen i hver av dem [29]. Y (nord) svarer til antallet meter punktet befinner seg fra ekvator.

I Google Maps' APIer brukes WGS84 som referansesystem. Dermed kan ressurser geolokalisert med WGS84 enkelt og direkte visualiseres i Google Maps. For å plotte ressurser fra Byantikvarens gule liste i Semantikart må kartposisjonene først konverteres fra UTM til WGS84. En algoritme som

gjør en slik konvertering ble funnet etter litt søking på nettet[4]. Algoritmen tar sonenummer, antall meter øst og vest som parametere og returnerer koordinatpunktet (lengdegrad, breddegrad) dette tilsvarer. Sør-Norge hører inn under sone nummer 32. Siden Oslo Byantikvars gule liste kun omfatter data fra Oslo Kommune, trengs heller ingen andre soner for Semantikart. Ved bruk av algoritmen, som jeg har oversatt til Java-kode, plottes dataene fra Byantikvarens liste til rett sted, i hvert fall med en grad av presisjon som er tilstrekkelig for Semantikart, på kartet. At konverteringen stemmer, kan lett sees da dataene inneholder adresser, med gatenavn og nummer, som stemmer overens med kartplasseringene som gjøres i applikasjonen.

Hovedproblemet med Byantikvardataene dreier seg dog ikke om å få dem plottet på kart. Problemet ligger i at de kun finnes tilgjengelige på endepunktet med UTM-enkodede geodata. Dermed kan ikke avstandsfiltreringskriteriumet beskrevet ovenfor anvendes på dem. Dataene må hentes inn i applikasjonen før de konverteres til lengde- og breddegrader. Derfor lar ikke Byantikvardataene seg avgrenses på bakgrunn av brukerens posisjon med de virkemidlene som er brukt i Semantikart. Dette problemet kunne vært løst ved å lage et eget filtreringskriterium til bruk på data på UTM-format, eller ved å konvertere dataene før de legges på serveren. Dette er ikke prioritert i Semantikart. Forskjellen i bruk av geoformat fra datasett til datasett illustrerer behovet for enighet om felles vokabular brukt ved publisering av offentlige data, blant annet http://www.w3.org/2003/01/geo/wgs84_pos# brukt om geolokaliserte data.

[4]http://mediakey.dk/ cc/convert-northing-and-easting-utm-to-longitude-and-latitude/

# Kapittel 7

# Oppsummering og veien videre

## 7.1 Åpne offentlige data

Stadig flere offentlige data frigjøres som følge av den politiske viljen og fokuset som i økende grad rettes mot denne tematikken. Frigjøring av offentlige data åpner for viderebruk og sammenstilling på tvers av ulike datasett. Dette vil gavne det offentlige i form av bedret informasjonsflyt mellom ulike etater, mer effektiv informasjonsutveksling og økt felles forståelse. Dessuten skapes et stort potensiale for innovasjon basert på utvikling av nye applikasjoner bygd på offentlige data.

Ny, viktig informasjon og nye sammenhenger kan synliggjøres gjennom sammenstilling av offentlige datasett. Videre blir samfunnet mer transparent når datagrunnlaget politiske beslutninger og saker i media baseres på finnes offentlig tilgjengelig.

Det er mange argumenter for økt andel av åpne, offentlige data. Potensialet for innovasjon og økonomisk vekst tilknyttet dette er spesielt vektlagt i denne sammenhengen. Semantikart er en applikasjon som utnytter dette potensialet på en plattform som gjennom mobilitet og innebygde verktøy bidrar til nye muligheter for applikasjonsutvikling: mobile, håndholdte enheter.

## 7.2 Semantiske vevapplikasjoner på mobile, håndholdte enheter

Problemene som oppstår i forbindelse med å bygge applikasjoner rundt åpne, offentlige data, knyttes først og fremst til ulikhetene mellom måtene

slike data er frigjort på. Valget av formater og bruken av identifikatorer og representasjonsformer om de samme tingene varierer fra datasett til datasett.

Gjennom semantiske vevteknologier, data beskrevet i RDF-formater og herunder bruk av URIer til navgivning av ressurser, tilbys en egnet måte å overkomme disse utfordringene på. Med URIer sikres en utvetydig måte å referere til vilkårlige entiteter på. Dette inkluderer lenkene mellom data, som i RDF også beskrives med URIer. RDF sikrer dermed en felles måte å beskrive data på som gjør at heterogene datakilder kan sammenstilles.

Data beskrevet med RDF bør tilgjengeliggjøres på nett, på en server som inneholder et SPARQL-endepunkt. Gjennom SPARQL kan klienter, for eksempel applikasjoner som Semantikart, få tilgang til de offentlige data som er tilgjengeliggjort med RDF. Via SPARQL-spørringer kan applikasjonene be om de dataene de måtte ønske og kombinere slike på tvers av datasett.

Som i RDF, identifiseres ressurser i RESTfulle systemer med URIer. I et RESTfullt system kan det gjøres HTTP-operasjoner på URIer, både de som identifiserer instanser og de som identifiserer samlinger av slike. Gjennom disse operasjonene kan ressurser enkelt leses, modifiseres, legges til og fjernes. Et oppslag på en ressurs vil returnere en representasjon av ressursen i REST. Dermed sikres det at URIene som brukes er selvdokumenterende, noe som er i tråd med semantisk vev-tankegang jamfør Berners-Lees retningslinjer: lenkede data. Ved å beskrive data på RESTfulle systemer med RDF skapes lenker til andre data. Applikasjoner som kobles opp mot slike systemer kan gjennom lenkene traversere dataene, som en vev av data. REST er av disse grunnene en type arkitektur som passer godt sammen med semantiske vevteknologier og dermed publisering av åpne, offentlige data.

REST tilbyr klienter lettvin tilgang til og mulighet for modifisering av ressurser via HTTP. Gjennom SPARQL-endepunkter sikres mulighet for avanserte spørringer mot data, også dette via HTTP. Servere der det ønskes å tilby lettvektsoppslag på, såvel som avanserte spørringer mot åpne offentlige data, bør derfor ha data beskrevet med RDF, være RESTfulle og tilby et SPARQL-endepunkt.

Siden kommunikasjonen i REST og SPARQL foregår via HTTP, kan en klient få tilgang til all nødvendig funksjonalitet over nett. Dette gjør at tilgang på ressurser på "den semantiske veven" er tilgjengelig for alle enheter med nettilgang, blant annet mobile, håndholdte enheter.

Mobile, håndholdte enheters mobilitet kombinert med innebygde verktøy for stedfesting av enhetens geografiske posisjon, slik som GPS, gjør slike enheter spesielt egnet for tjenester som utnytter konteksten gitt av brukerens lokalitet. Med semantiske vevteknologier (RDF) kan data beskrives med

standardvokabularer som blant annet kan gi dem kategori- og geografisk posisjonstilknytninger. RDFs gjenbruk av vokabularer og lenking av data er med på å skape en kontekst på tvers av datasett. Dette gjør at applikasjoner kan bygges på en enkel måte med en generell mulighet for å tilknytte seg data med bestemte egenskaper fra ulike kilder. Semantikart utnytter dette ved å knytte valgbare kategorier til geografisk visualiserbare data fra ulike kilder.

Semantikart er en applikasjon som illustrerer noen av mulighetene som gis av potensialet for nye anvendelser basert på åpne, offentlige data. Det å få anledning til å finne nærmeste ladestasjon for elbil og annen informasjon om nærmiljøet rundt deg, kan utvilsomt sies å være nyttig for mange brukere.

## 7.3 Videre arbeid

### 7.3.1 Datafrigjøring

Et håp med denne rapporten, Semantikart og Semicolon forøvrig, er å belyse mulighetene åpne data er med på å gi, og å synliggjøre viktigheten av at det tilgjengeliggjøres flere slike data. Imidlertid er det mange data som har stort potensiale for viderebruk, både med tanke på bedring av interoperabilitet i offentlig sektor og potensiale for innovasjon, som ikke finnes åpne i Norge i dag. Kartdata og data fra Enhetsregisteret står som eksempler på slike.

Med flere tilgjengelige offentlige data kan Semantikart og andre applikasjoner basert på åpne, offentlige data gis nye muligheter. I dag er enkelte tjenester og applikasjoner, slik som Borgerkanalen, hemmet av mangel på åpne kartdata, blant annet kommunegrenser og veidata. Slike applikasjoner vil først kunne vokse fritt når alle nødvendige offentlige data er tilgjengeliggjort.

#### Åpne, offentlige kartdata

Kartdata er nevnt flere steder som en type data mange ønsker å få tilgang på, blant annet i 2.2.1 på side 11. I Norge er det Statens Kartverk som forvalter kartdata, noe de mottar offentlig støtte for å gjøre. Blant Kartverkets data er det foreløpig kun selve kartbladene som er (delvis) åpne. Disse kartbladene er velegnede til å visualisere data med en geografisk tilknytning på, og bidrar dermed i en viss forstand til å skape muligheter for applikasjonsutvikling basert på dette. For eksempel kunne Semantikart vært realisert med Statens Kartverks kartblader i kombinasjon med Google Maps' visningstjenster. Google Maps' egne kartblader er dog valgt istedenfor

Kartverket sine da sistnevnte har en begrensing i åpenhet gitt av vilkåret om at de ikke kan brukes i inntektsgivende tjenester og et maksimumsantall av kartbilder som tillates hentet ut per sluttbruker over en viss tidsperiode [20]. Dessuten slipper man ved bruk av de innebygde kartbladene i Google Maps utviklingstid brukt på å integrere to ulike tjenester.

Dataene Kartverket forvalter har en større nytteverdi for nye applikasjoner enn hva som finnes iboende i kartbladene alene. Dataene som kartbladene bygger på, kartdata, har mange anvendelsesområder. Med data om høydeforskjell i terrenget, kan tjenester som viser og varsler om snøskredutsatte partier (helning på over 30°) lages[1], eller illustrere hvilke områder som kan omfattes av en flom av en viss størrelse. Med tilgang på kommunegrenser kan tjenester som Borgerkanalen, som baseres på rapportering av feil og mangler i det offentlige, finne ut hvilken kommune som står ansvarlig for å ta hånd om slike på en gitt lokalitet. Med tilgjengelige data om hva slags type en vei er av (fylkesvei, kommunal-, privat- eller riksvei), kan slike tjenester sørge for kontakt med riktig myndighet ved teleskader eller hull i veien etc. Kommunegrenser og andre kartdata finnes imidlertid ikke åpent tilgjengelig hos Statens Kartverk i dag.

Jamfør fellesføringene for 2011, 2.1.3 på side 11, skal alle etater tilgjengeliggjøre egnede og eksisterende rådata basert på informasjon av samfunnsmessig verdi i maskinlesbare formater. Statens Kartverks kartdata er utvilsomt av samfunnsmessig verdi. Kartverket har dog betydelige inntekter i forbindelse med salg av kartdata. Siden fellesføringen kun omfatter data der kostnaden av tilgjengeliggjøring antas å være beskjeden og bortfall av inntekter ved salg av data anses som en kostnad, slipper Kartverket unna dette pålegget om å tilgjengeliggjøre data.

Mangelen på åpne data hos Kartverket har skapt stor debatt. Som følge av avslag på tilgang til kommunegrensedata, har prosjektet "Fri Geo Norge" opprettet en klagesak mot det offentlige [32]. Siden Statens Kartverk får statsstøtte for å samle inn og forvalte geodata, synes mange det er underlig at slike ikke gjøres åpent tilgjengelig til folket som i utgangspunktet har betalt for dem gjennom skattepenger. Potensialet for verdiskapning i Norge som ligger i fri tilgang på slike data, er trolig større enn inntekten staten får ved å ta betalt for dataene. Fram til slike data finnes åpne i Norge, må utviklere av tjenester som ønsker tilgang på kartdata enten avfinne seg med å betale for tilgang, eller se etter andre alternativer. Det sistnevnte har blant annet Norwegian Unix User Group (NUUG) gjort. Som konsekvens av mangelen på åpne kommunegrensedata har NUUG lasert en egen API-tjeneste, kalt

[1] En slik tjeneste finnes tilgjengelig på http://www.skrednett.no, men er ikke optimalisert for visning på små skjermer eller mulighetene for utnyttelse av funksjonaliteten, slik som GPS, tilbudt av MHEer

Mapit[2], som lar en få automatisk svar på spørsmål som "i hvilken kommune og i hvilket fylke befinner dette geografiske punktet seg?". Svar returneres i maskinlesbart format (JSON). Kommunegrensene i tjenesten, bygger på data fra dugnadsinitiativet Openstreetmap.org [32] som står som et åpent alternativ til Kartverkets lukkede data.

### Enhetsregisteret

Brønnøysundsregistrenes (BRReg) enhetsregister inneholder informasjon om alle norske foretak: firmanavn, navn på daglig leder, adresseinformasjon, telefonummer, næringskode osv.

I samarbeid med Semicolon og Institutt for informatikk ved UIO, har BRReg lansert en vevtjeneste som leverer data om en gitt enhet; enten i form av en nettside, eller på maskinlesbart og semantisk vev-støttet RDF/XML-format. Ved å angi organisasjonsnummeret som identifiserende del av tjenestens URI, returneres en representasjon av enheten nummeret identifiserer. For eksempel vil et oppslag på https://ws.brreg.no/lod/page/974760673 åpne en nettside med informasjon om enheten med organisasjonsnummer 974760673 (nummeret til Registerenheten i Brønnøysund selv), mens et oppslag på https://ws.brreg.no/lod/data/974760673, vil gi en RDF/XML-representasjon av den samme enheten. Informasjonen vevtjenesten tilbyr, er foreløpig begrenset til å omfatte enhetens navn, organisasjonsnummer og adresse.

Enhetsregisteret er interessant som datasett for mange anvendelser, også Semantikart. I Semantikart kunne man tenke seg en egen kategori, “enheter”, som dersom den var valgt, viste alle registrerte enheter i nærheten av brukerens posisjon og tilhørende informasjon om disse. Slik funksjonalitet lar seg imidlertid ikke implementere gjennom bruk av BRRegs vevtjeneste mot enhetsregisteret. Denne vevtjenesten forutsetter inndata i form av et organisasjonsnummer for å kunne returnere data om en organisasjon. En slik type vevtjeneste, der man må vite noe sentralt på forhånd for å kunne få informasjon tilbake, begrenser dataenes åpenhet. Dette står som motsetning til tanken bak semantiske vevtjenester, der for eksempel et HTTP-GET-oppslag på samlingsnivå i et RESTfult system (se 3.4 på side 42), kan returnere alle aktuelle data innenfor samlingen, eller et SPARQL-endepunkt der det gis mulighet for å spørre mot alle tilgjengelige ressurser.

Tilnærming til dataåpenhet i vevtjenesten mot enhetsregisteret, der en identifikator er eneste innsynsmulighet til data, begrenser hvilke anvendelser denne tjenesten kan ha. Dermed svekkes også dataenes åpenhet. For Semantikart er det geolokalitet som står sentralt for hvilke data som skal

[2] http://mapit.nuug.no/

tilbys brukeren. Siden informasjon, deriblant geografisk tilknytning, ikke kan utledes direkte fra et organisasjonsnummer [37], er ikke vevtjenesten mot enhetsregisteret alene anvendbar for Semantikart i sin nåværende form. Semantikart vil måtte på forhånd vite hvilke organisasjonsnumre som finnes i nærheten av brukerens posisjon for å kunne dra nytte av denne vevtjenesten. Denne informasjonen finnes ikke åpent tilgjengelig.

### 7.3.2 Videre implementasjon i Semantikart

#### Distribuering av spørringer til egen server

Slik Semantikart er konstruert nå, ligger alle SPARQL-spørringene mer eller mindre hardkodet i programkoden. Det eneste som er variabelt i spørringene er avstandsfiltreringen som skapes på bakgrunn av brukerens posisjon.

Som poengtert, er det ønske om og sannsynlighet for at den åpne, offentlige datamengden i Norge skal øke med tiden. Nye datatyper som ikke finnes tilgjengelige i dag, men som burde vært visualisert i Semantikart vil med tiden tilgjengeliggjøres og ønskes tilsluttet i applikasjonen. Slik Semantikarts kode er konstruert på nåværende tidspunkt, må slike nye data tilknyttes ved å tilføre programkoden nye spørringer som utløses dersom nye, tilknyttede datakategorier velges av brukeren. Denne koden må så rulles ut til Android Market hvorpå brukerne blir bedt om å oppdatere applikasjonen til nyeste versjon.

Som utvikler har jeg ingen kontroll over de eksisterende datasettene Semantikart benytter seg av. Semantikart henter data fra servere kontrollert av mer eller mindre kjente og ukjente mennesker. Det er mulig og sannsynlig at flere av de benyttede datasettene med tiden kan endre seg med tanke på struktur, flyttes til en annen server, endre endepunktadresse eller i verste fall fjernes. En slik endring i struktur har jeg allerede opplevd med DBpedia-datasettet. Disse dataene var opprinnelig annotert, i RDF, med språkmerket "nn" (nynorsk) på literalene tilknyttet predikatene "dc:title" og "dc:description" til tross for at informasjonen stod skrevet på bokmål. Spørringen i Semantikart tok høyde for dette og ba om data med dette språkmerket. Da DBpedia så plutselig rettet opp i denne "feilen" og endret dataene til å istedenfor være beskrevet med språkmerket "no", fungerte ikke lenger DBPedia-spørringen slik den skulle i applikasjonen. Alle ressursene applikasjonen hentet fra DBPedia ble derfor stående med tomme titler og beskrivelser.

For å adressere dette problemet: mulige endringer i eksisterende datasett og tilslutning av nye datasett i Semantikart, må applikasjonen struktureres

annerledes. En måte dette kunne vært gjort på, var om alle spørringene Semantikart benyttet seg av, istedenfor å finnes i programkoden, ble hentet inn fra nettet. På denne måten kunne spørringer endres og tilføyes i sanntid, uten at brukeren legger merke til det. Jamfør en semantisk vev-tilnærming kunne en egen server opprettes som inneholdt data beskrevet med RDF med literaler som beskrev spørrestrenger, endepunktadresser og hva slags datatyper og tihørende kartvisualiseringsikon data hentet med de ulike spørringen skulle tilknyttes. Semantikart kunne så hente spørredataene fra denne serveren og pare dem med en stedsfiltrering basert på brukerens posisjon før de ble sendt de respektive endepunktene.

### Optimalisering av funksjonalitet ved endret avstandsfiltrering

I Semantikart gis det mulighet, gjennom menyvalg, til å endre hvor stort område applikasjonen skal hente sine data fra, for eksempel innenfor cirka tre eller fem kilometer. I forbindelse med en endring av avstandsfiltrering til en høyere verdi, vil datamengden i applikasjonen økes. Det ligger normalt flere data innenfor fem kilometers avstand fra brukeren enn det gjør for tre kilometer. Ved at avstandsfiltreringen endres til en lavere verdi, reduseres derfor datamengden.

Slik Semantikart er implementert vil en endring av avstandsfiltrering til en lavere verdi føre til at alle dataene (stedene) først fjernes fra applikasjonen før nye data hentes ved hjelp av spørringer med oppdatert (redusert) geografisk avgrensningsområde. Hvis nettverkshastigheten er lav, kan det ta betydelig med tid å få svar på spørringer. Datakommunikasjon, blant annet via spørringer, kan i tillegg være kostbart, både med tanke på pris for dataoverføring og batteribruk. Antallet spørringer Semantikart gjør, bør derfor begrenses så langt det lar seg gjøre.

Det er imidlertid ikke nødvendig å gjøre spørringer på nytt når avstandsfiltreringen endres til et mindre område enn det den opprinnelig var på. Hvis området endres fra for eksempel “ca 5 km” til “ca 3 km”, ville det være mer hensiktsmessig å løpe gjennom alle stedene i applikasjonen og fjerne de av dem som lå lenger unna enn 3 km. Denne metoden ville i de fleste situasjoner være mer effektiv enn den som benyttes i Semantikart i dag, da det åpenbart er mer effektivt å løpe gjennom et sett $A$ bestående av $n$ steder og slette $n - m$ av dem, enn det er å slette hele settet for så å hente inn $m$ steder til et nytt sett, $B$, hvor $B \subset A$.

Andre veien, endret avstandsfiltrering til en større avstand, kan også optimaliseres for Semantikart. Dette kan gjøres ved å redusere antallet nye data som innhentes med en spørring kontra mengden som hentes i dag. Når

Semantikarts avstandsfiltrering endres fra "ca 3 km" til "ca 5 km", slettes naturlig nok ingen data i applikasjonen. Det som skjer, er at det skapes en spørring som ber om alle data geoposisjonert inntil ca fem kilometer unna brukeren. Denne spørringen vil nødvendigvis matche en rekke data som allerede finnes i applikasjonen. Unødvendig mye data blir dermed levert applikasjonen fra serveren. Disse dataene som finnes i applikasjonen fra før, gjenkjennes, og forblir uendret. Mange data som hentes i Semantikart gjennom spørringer i slike tilfeller er derfor redundante.

Ved å justere spørrestrukturen som brukes ved endret (økt) avstandsfiltrering, kan redundant datainnhenting minimaliseres. Det geografiske avgrensningsområdet som brukes i spørringen kan istedenfor å dekke alt innenfor fem kilometer, justeres til å omslutte et område som strekker seg fra tre kilometer fra brukerens posisjon til fem kilometer fra brukerens posisjon. Et SPARQL-filter som gjør en slik avgrensning kan skisseres som følger:

```
FILTER(((?b >= bposB - femkm) && (?b <= bposB + femkm)
       &&
       (((?l <= bposL - tokm) && (?l >= bposL - femkm)) ||
        ((?l >= bposL + tokm) && (?l <= bposL + femkm))))
      ||
       ((?l >= bposL - femkm) && (?l <= bposL + femkm)
       &&
       (((?b <= bposB - tokm) && (?b >= bposB - femkm)) ||
        ((?b >= bposB + tokm) && (?b <= bposB + femkm)))))
```

I denne skissen står $?b$ for stedets breddegrad, $?l$ for stedets lengdegrad, $bposL$ og $bposB$ for henholdsvis lengde- og breddegraden til brukerens posisjon, mens $femkm$ og $tokm$ står for et antall grader som tilsvarer fem eller to kilometer i bredde- eller lengdegrader avhenging av om de står sammen med $bposB$ eller $bposL$.

Det at brukeren kan være i bevegelse er med på å vanskeliggjøre problematikken rundt denne typen innhenting av data. Dersom brukeren har beveget seg for eksempel 200 meter østover siden data sist ble lastet inn i applikasjonen, vil ikke filteret beskrevet ovenfor inkludere data som lå for eksempel 3,1 kilometer øst for brukerens opprinnelige posisjon. Dette vil skje fordi disse nå ligger 2,9 kilometer unna brukeren og dermed utenfor området som strekker seg mellom 3 og 5 kilometer fra brukerens nye posisjon som den nye spørringen ber om data med utgangspunkt i.

Slik Semantikart er laget i dag, hentes data på nytt når brukeren kommer til et punkt som ligger mer enn en tredjedel av valgte avstandsfiltrerings avstand fra det punkt applikasjonen sist hentet data fra, for eksempel når brukeren

befinner seg én kilometer unna punktet han først startet applikasjonen fra med "ca 3 km" som avstandsfilter. Semantikart kjører i slike tilfeller spørringene på nytt uten at det gjøres noe spesielt med det geografiske avgrensningsområdet i filteret. Dermed er det også i denne situasjonen potensiale for redundans i form av data som allerede finnes i applikasjonen som hentes på nytt.

Innhenting av nye data når brukerens posisjon er endret bør derfor gjøres på en annen måte enn den skissert ovenfor, kanskje en som tar hensyn til brukerens bevegelsesretning fra siste posisjon data ble oppdatert på.

# Tillegg A

# Semantikart

Dette tillegget starter med en beskrivelse av hvordan applikasjonen Semantikart og dens kildekode kan installereres og testes ut. Videre følger en beskrivelse av applikasjonens tekniske oppbygning, blant annet hvilke klasser den består av og hvordan de ulike SPARQL-spørringene ser ut. Avslutningsvis følger en brukerveiledning for hvordan Semantikart brukes.

## A.1 Installasjon av applikasjonen

Semantikart kan testes på to ulike måter, i to litt forskjellige utgaver. Applikasjonen finnes tilgjengelig både gjennom Android Market og som kildekode åpent tilgjengelig på nett. Denne kildekoden kan åpnes i et utviklingsmiljø og testes ved hjelp en Android-emulator eller en tilkoblet enhet.

### A.1.1 Gjennom Android Market

Semantikart finnes tilgjengeliggjort for allmennheten gjennom Android Market. For å installere Semantikart på en enhet, kan knappen "INSTALLER" fra applikasjonens autogenererte nettside[1] brukes. Alternativt kan applikasjonen lastes ned gjennom Android Marked-applikasjonen som finnes installert på alle Android-enheter. Søk i tilfelle etter "Semantikart" i denne applikasjonen.

Utgaven av Semantikart som finnes tilgjengelig i Android Market er noe begrenset. Borgerkanalen er foreløpig ikke på et stadium, på serversiden, der

[1] https://market.android.com/details?id=no.uio.ifi.at.jenskm.semantikart

det gjøres noe med en innrapportert sak. En borgerkanalsak som opprettes sendes med andre ord foreløpig ikke til noen myndighet. Av denne grunn er funksjonaliteten for oppretting av borgerkanalsaker strippet vekk i versjonen av Semantikart som finnes tilgjengelig i Android Market. Denne versjonen er dermed kun beregnet på konsumering av informasjon. Brukeren har ikke tilgang til å opprette informasjon.

Semantikart krever versjon 2.2 eller nyere av Android-operativsystemet. Enheter som ikke har dette, har ikke mulighet til å kunne kjøre Semantikart.

### A.1.2 Fra kildekode

Kildekoden finnes tilgjengelig på http://jenskm.at.ifi.uio.no/semantikart/. Her kan den blas i på nett, eller lastes ned i sin helhet som et komprimert arkiv (Semantikart.zip). Se A.2 på neste side for informasjon om kildekodens bestanddeler.

For å teste kildekoden trengs Androids SDK[2] og Java Development Kit[3] (JDK) 5 eller 6.

Utviklingsmiljøet Eclipse har utvidelser som støtter Android. Bruk av Eclipse forenkler prosessen med å utvikle og teste Android-programmer og anbefales derfor brukt ved testing eller eventuell modifisering av Semantikarts kildekode.

En utfyllende beskrivelse av hvordan Android-utviklingsmiljøet kan settes opp, finnes tilgjengelig på nett[4].

#### Eksekvering av Semantikarts kildekode

Semantikarts kildekode kan kjøres på en maskin, enten via en tilkoblet enhet med versjon 2.2 eller nyere av Androids operativsystem, eller ved hjelp av Android SDKs innebygde emulator.

For at funksjonaliteten fra Google Maps skal kunne benyttes i en Android-applikasjon, må en personlig API-nøkkel anskaffes fra Google som illustrerer et samtykke om bruksvilkårene. Framgangsmåten beskrevet i Googles Android-API-sider[5] følges for å skaffe en slik nøkkel. For at Google Maps-

[2] http://developer.android.com/sdk/index.html

[3] http://www.oracle.com/technetwork/java/javase/downloads/index.html

[4] http://developer.android.com/sdk/installing.html

[5] http://code.google.com/intl/no/android/add-ons/google-apis/mapkey.html#getdebugfingerprint

funksjonaliteten skal kunne benyttes i Semantikart, må en slik nøkkel skapes og plasseres som verdi i attributtet "android:apiKey" i XML-fila som definerer utseendet til aktiviteten som viser kart i applikasjonen: `res/layout/map.xml`.

## A.2 Teknisk oppbygning

En Android-applikasjons funksjonalitet og oppførsel beskrives i Java-klasser og XML-filer. Dette avsnittet beskriver de sentrale komponentene som utgjør implementasjonen av Semantikart.

### A.2.1 Klasser

Semanikarts programkode er spredt over en rekke klasser. Javadoc for disse klassene finnes tilgjengelig på nett[6]. De viktigste klassene i applikasjonen beskrives også nedenfor.

Det finnes mange typer klasser spesielt laget for Android. En aktivitet, klassen Activity, representerer et idéperspektiv. Alle de forskjellige perspektivene som vises i Semantikart er derfor subklasser av denne klassen.

- Semantikart — Subklasse av "Application", noe som gjør at den instansieres når applikasjonen startes opp og er den klassen som sist terminerer. Denne klassen inneholder alle globale variabler som brukes av de andre klassene, blant annet en ArrayList som inneholder alle stedene og variabler som holder informasjon om posisjonen brukeren sist hentet data fra.
- MyTabActivity — Applikasjonens hovedaktivitet, subklasse av Actitivity og ansvarlig for å vise en fane med to mulige perspektivvalg, "Kart" og "Steder". Disse perspektivene har hver sin tilhørende aktivitet, MyMap.java og PlacesList.java. MyTabActitivity tar seg av oppførsel disse to aktivitene deler, slik som bytte fra den ene til den andre og felles menyelementer, slik som tilgang til datainnstillinger.
- Infopunkt — En enkel klasse som representerer en ressurs, et infopunkt og tilhørende data som tittel, beskrivelse og URI til eventuelt tilknyttet bilde.
- Place — Klasse som utvider Android-klassen OverlayItem, hvilket betyr at den kan brukes for å modellere steder på et kart. Klassen

[6] http://jenskm.at.ifi.uio.no/semantikart/Javadoc/index.html

inneholder et objekt som beskriver stedets lokalitet og en ArrayList som inneholder tilhørende infopunkter.

- Datatype — En enumerator som beskriver alle de forskjellige datatypene (datasettene) applikasjonen nyttiggjør seg av. Hver av dem har tilknyttede variabler som beskriver hvordan de skal visualiseres på kart og hvilke endepunkter de finnes tilgjengelige på.

- HTTPgetData — En klasse som utvider AsyncTask og dermed opererer på en egen tråd. Denne klassen brukes for å hente data fra et endepunkt, tolke disse dataene og så sende Place-objekter til lagring i Semantikart-klassen på "den vanlige" applikasjonstråden.

- HTTPpostData — Denne klassen opererer også på en egen tråd ved å utvide AsyncTask-klassen og brukes for å poste data (Borgerkanalen), gitt av en TURTLE-representasjon av et Place-objekt klassen instansieres med, ved hjelp av HTTP til Computas Linked Open Data Server.

- SparqlQueryString — I denne klassen finnes de ulike SPARQL-spørringene hardkodet i ulike metoder som returnerer tekststrengrepresentasjoner av spørringene. Klassen inneholder også diverse hjelpemetoder som brukes av spørringsmetodene for å skape et filteringskriterium basert på brukerens nåværende posisjon.

- InfopunktDetails — Denne aktiviteten definerer hvordan et detaljert perspektiv over et infopunkt skal oppføre seg. I perspektivet vises alle tilgjengelige data om et infopunkt.

- InfopunktEdit — En aktivitet som definerer oppførselen ved endring av et Infopunkt. Klassen utvider Activity og brukes når det lages en ny borgerkanalsak.

- MyMap — Her styres funksjonaliteten knyttet til kartvisning av data.

- UserLocationOverlay — Subklasse av klassen MyUserLocation fra Androidbiblioteket. Denne klassen brukes for å vise brukerens posisjon på kart og å kontrollere oppførsel knyttet til endring av denne.

- PlacesList — Denne aktiviteten brukes for å vise en klikkbar liste over steder og kontrollere oppførsel knyttet til denne.

- DatatypeSettingsAcitivty — En egen aktivitet som viser en liste over de ulike datatypene applikasjonen benytter seg av og knytter funksjonalitet til valg av om hver av disse skal vises i Semantikart.

### A.2.2 AndroidManifest.xml

Dette er en XML-fil som må defineres i enhver Android-applikasjon. Her beskrives generelle ting som applikasjonens versjonskode, navn, ikon og aktivitetene som inngår i applikasjonen.

### A.2.3 Brukergrensesnittfiler

I mappen "res" finnes filer brukt til utforming av Semantikarts grensesnitt. Denne mappen inneholder flere undermapper som hver består av flere filer.

- drawable-mappene — Inneholder billedfiler, i ulike oppløsninger basert på mappenes navnesuffiks (high, low og medium DPI). Filene i disse brukes som grafiske elementer i Semantikart, blant annet som applikasjonens oppstartsikon, kartdatatypeikoner og ulike menyikoner.
- layout — Består av filer på XML-format som definerer utseendet til de ulike aktivitene i Semantikart, slik som hvordan kartperspektivet eller det detaljerte infopunktperspektivet skal se ut.
- menu — Omfatter XML-filer som beskriver hvordan de ulike menyene som dukker opp ved at man trykker på en Android-enhets fysiske menyknapp skal se ut i de ulike aktivitetene.
- values — Inneholder XML-beskrivelser av enkelte tekststrenger brukt av programmet. Hver får sin egen identifikator. Denne typen definering av strenger skiller tekst fra programkode, gjør at samme streng enkelt kan benyttes flere steder i programmet og forenkler mulig senere oversettelse av applikasjonen til andre språk.

### A.2.4 Spørringer

Her beskrives det hvordan spørresteksttrengene Semantikart sender ut for hver av datakildene ser ut. Avstansfilterets verdier vil regnes ut av applikasjonen basert på brukerens posisjon og valgte preferanser og er her kun satt med eksempelverdier. I applikasjonens kildekode finnes disse spørringene i klassen "SparqlQueryString".

#### Borgerkanalen

```
PREFIX dc:    <http://purl.org/dc/terms/>
```

```
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
PREFIX wgs: <http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT ?identifier ?title ?subtitle ?type ?longitude ?latitude
WHERE {
GRAPH <http://opendata.computas.no/dataset/borgerkanalen> {

?identifier a <http://opendata.computas.no/voc/borgerkanalen/Sak> ;
a ?type ;
dc:title ?title ;
dc:description ?subtitle ;
dc:location ?location .

?location wgs:long ?longitude ;
    wgs:lat ?latitude .


FILTER  (xsd:float(?latitude) < 109.909865)
FILTER (xsd:float(?latitude) > 9.909865)
FILTER  (xsd:float(?longitude) < 110.766138)
FILTER (xsd:float(?longitude) > -89.233862)
}}
```

### Byantikvarens gule liste

```
PREFIX byantikvaren: <http://www.byantikvaren.oslo.kommune.no/>
PREFIX etrs89: <http://www.euref.eu/etrs89/>
PREFIX statkart: <http://www.statkart.no/GAB-registeret.>
PREFIX fn: <http://www.w3.org/2005/xpath-functions#>

SELECT ?type ?latitude ?longitude ?title ?subtitle

WHERE {
?place byantikvaren:byggeaar ?byggeaar ;
   byantikvaren:vernedato ?vernedato ;
   byantikvaren:sted ?sted ;
   statkart:ipsgatenummer ?gatenr ;
   byantikvaren:vernestatus ?vernestatus ;
   etrs89:north ?latitude ;
   etrs89:east ?longitude .

LET (?type := "http://www.byantikvaren.oslo.kommune.no/gul_liste")
LET (?title := fn:concat(?sted, " ",  ?gatenr))
```

```
LET (?subtitle := fn:concat("Byggeår: ", ?byggeaar,
                            ", vernestatus: ", ?vernestatus,
                            ", vernedato: ", ?vernedato))
}
LIMIT 200
```

### Byggesaker

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX dc:  <http://purl.org/dc/terms/>
PREFIX wgs: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX byggesak: <http://opendata.computas.no/voc/oslo.kommune.no/>
PREFIX fn: <http://www.w3.org/2005/xpath-functions#>

SELECT  DISTINCT ?identifier ?type ?title ?subtitle
                 ?latitude ?longitude
WHERE {
GRAPH <http://opendata.computas.no/dataset/byggesak_oslo> {

?identifier  a ?type ;
     a byggesak:Byggesak ;
     byggesak:streetname ?streetname;
     byggesak:streetnumber ?streetnumber;
     dc:description ?subtitle;
     dc:location ?loc .
?loc wgs:lat ?latitude ;
     wgs:long  ?longitude .

LET (?title := fn:concat(?streetname, " ", ?streetnumber))

FILTER  (xsd:float(?latitude) < 109.909865)
FILTER (xsd:float(?latitude) > 9.909865)
FILTER  (xsd:float(?longitude) < 110.766138)
FILTER (xsd:float(?longitude) > -89.233862)
}}
```

### DBpedia

```
PREFIX geo:   <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

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

```
PREFIX dbpedia: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?identifier ?picture ?title ?subtitle
                ?latitude ?longitude
WHERE {
?identifier rdfs:label ?title ;

    geo:lat ?latitude ;
geo:long ?longitude .

 OPTIONAL {
 ?identifier dbpedia:thumbnail ?picture.
 }
 OPTIONAL {
 ?identifier dbpedia:abstract ?subtitle .
 FILTER (lang(?subtitle) = "no")
 }

FILTER (lang(?title) = "no")
FILTER ((?latitude) < 60.0)
FILTER ((?latitude) > 59.6)
FILTER ((?longitude) < 11.0)
FILTER ((?longitude) > 10.7)
}
```

### Tellus

Her må URI-en som står som objekt til "dct:subject" endres basert på kategori det ønskes data fra: spisesteder, hoteller eller butikker etc.

```
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
prefix dct: <http://purl.org/dc/terms/>
prefix tellus: <http://opendata.computas.no/voc/tellus/>
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT ?identifier ?title ?subtitle ?type ?longitude ?latitude
WHERE {
  GRAPH <http://opendata.computas.no/dataset/tellus> {
   ?identifier a tellus:Product ;
       a ?type ;
       dct:title ?title ;
       dct:location  ?loc ;
```

```
       dct:subject
        <http://opendata.computas.no/data/tellus-categories/2> .

   ?loc geo:long ?longitude ;
           geo:lat ?latitude .

   OPTIONAL {
?identifier dct:description ?subtitle .
FILTER (lang(?subtitle) = "no")
}

  FILTER ((xsd:decimal(?latitude)>59.95) &&
          (xsd:decimal(?latitude) <60))
  FILTER ((xsd:decimal(?longitude)>10.6) &&
          (xsd:decimal(?longitude) <11))
}}
```

### Ladestasjoner

```
prefix off: <http://opendata.computas.no/voc/ladestasjoner.no/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX geonames: <http://www.geonames.org/ontology#>
PREFIX dc:  <http://purl.org/dc/terms/>
PREFIX smf: <http://topbraid.org/sparqlmotionfunctions#>

SELECT ?identifier ?title ?type ?latitude ?longitude
WHERE {
GRAPH <http://opendata.computas.no/dataset/ladestasjoner> {
   ?identifier a off:Ladestasjon ;
               a ?type ;
               dc:title ?title ;
               geonames:lat ?latitude ;
               geonames:long ?longitude .

LET  (?latitude := smf:cast(?latitude, xsd:float))
LET  (?longitude := smf:cast(?longitude, xsd:float))

FILTER  (xsd:float(?latitude) < 109.909865)
FILTER (xsd:float(?latitude) > 9.909865)
FILTER  (xsd:float(?longitude) < 110.766138)
FILTER (xsd:float(?longitude) > -89.233862)
}}
```

## A.3 Brukerveiledning

### A.3.1 Oppstart

Semantikart startes som andre Android-applikasjoner ved at man trykker på applikasjonens oppstartsikon i listen over applikasjoner.

Når applikasjonen startes tas brukeren til hovedaktiviteten, hvor det vises to velgbare faner øverst i bildet: "Kart" og "Steder". Kartfanen er valgt opprinnelig og kartaktiviteten, A.3.2, vises derfor. Ved å trykke på stederfanen, tas brukeren til stederaktiviteten. "Kart" og "Steder" er to ulike aktiviteter som visualiserer de samme dataene på ulike måter.

Et trykk på Android-enhetens fysiske menyknapp ("menu") bringer fram hovedmenyen. Denne menyen gir valgene "Datainnstillinger", A.3.4 på neste side og "Opprett ny Borgerkanalsak" (ikke tilgjengelig i applikasjonsversjonen som finnes i Android Market), A.3.5 på neste side. Hvis kartfanen er valgt gir menyen i tillegg valgene "Endre karttype" og "Min posisjon".

Ved å velge "Endre karttype" veksler Google Map-typen som vises fra et tegnet kart til et satellittbilde, eller omvendt. Trykk på "Min posisjon" justerer kartvisningen slik at brukerens geografiske posisjon sentreres i bildet.

Dersom stederfanen er valgt vises ikke de to sistnevnte menyvalgene i hovedmenyen, men derimot valget "Sortér". Et trykk på dette menyvalget gir brukeren mulighet til å velge blant fire ulike sorteringsmåter som stedene i stedslisten kan sorteres etter: stigende eller synkende alfabetisk orden, eller etter stigende eller synkende rekkefølge basert på stedenes avstand fra brukerens posisjon.

Hvis verktøy for Internett-tilgang og/eller geoposisjonering ikke er aktivert på MHE-en når hovedaktiviteten åpnes, blir brukeren bedt om å skru på dette. Dersom brukeren takker "ja" til dette, blir han sendt fra applikasjonen til en aktivitet som beskriver telefonens innstillinger for å aktivere nevnte verktøy.

### A.3.2 Kart

I kartaktiviteten vises et navigerbart kart. Brukerens posisjon indikeres av en blå runding på kartet, mens de ulike stedene vises som firkantede "nåler". Ved at man trykker på et sted åpnes en tekstballong som indikerer stedets navn. Tekstballongen inneholder også et rundt, svart ikon med en hvit pil på. Når man trykker på dette ikonet, åpnes en aktivitet: "Infopunkt", som

inneholder detaljert informasjon om infopunktet som finnes på dette stedet — dersom stedet inneholder kun ett infopunkt. Hvis stedet inneholder flere enn ett infopunkt, åpnes først en velgbarliste over stedets infopunkter. Ved å trykke på et av infopunktene, åpnes ovennevnte infopunktaktivitet.

De forskjellige stedene vises med ulike kartnåler avhengig av hva slags datatype infopunktene de inneholder er av. Dersom stedet inneholder flere infopunkter av forskjellige typer, vises et generelt ikon som indikerer dette.

### A.3.3 Steder

Stederaktiviteten viser en liste over alle stedene som er hentet inn i applikasjonen. Ved siden av hvert sted finnes et ikon som indikerer hva slags type eller typer infopunktet/-ene på stedet er av, og et tall som indikerer en anslagsvis avstand stedet ligger unna brukerens posisjon.

Ved å trykke på et sted i listen åpnes en liste over infopunktene som finnes på stedet, eller infopunktaktiviteten, avhengig av om det er flere, eller kun ett infopunkt tilknyttet det valgte stedet. Med andre ord er samme funksjonalitet koblet til et valg av sted i stedslisten som til ikonet på et sted i kartet.

### A.3.4 Datainnstillinger

Dette er en dialogboks som åpnes gjennom et valg i hovedmenyen. I dialogboksen finnes to alternativer: “Legg til/fjern datatyper” og “Endre avstandsfiltrering”. Ved å trykke på sistnevnte alternativ, gis brukeren mulighet til å endre det geografiske området data hentes inn fra, basert på en valgt approksimativ maksimumsavstand.

“Legg til/fjern datatyper” åpner en ny aktivitet der brukeren får velge hvilke datatyper som skal vises i applikasjonen. Dette skjer i form av en liste over disse med tilhørende avhukningsbokser.

Når datainnstillinger-dialogboksen lukkes, oppdateres dataene i applikasjonen dersom det er gjort endringer på hvilke datatyper som skal vises, eller valgt avstandsfiltrering.

### A.3.5 Ny Borgerkanalsak

Dette menyvalget gir brukeren mulighet til å velge et sted i kartet som en ny borgerkanalsak skal opprettes på. Når stedet er valgt, åpnes en ny

aktivitet for redigering av informasjon om saken. I denne aktiviteten gis det mulighet for å skrive inn tittel og beskrivelse, og å tilknytte saken et bilde gjennom enhetens bildetakningsprogram eller bibliotek over lagrede bilder. Brukeren gis så mulighet til å lagre saken, hvoretter den sendes til og lagres på Computas LOD-server.

# Bibliografi


[1] Areeb Alowisheq og David E. Millard. EXPRESS: EXPressing REstful semantic services. I *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, WI-IAT '09, side 453–456, Washington, DC, USA, 2009. IEEE Computer Society. ISBN 978-0-7695-3801-3. 41

[2] Arbeids- og administrasjonsdepartementet. Fra bruk til gjenbruk. Rapport, august 2004. 7, 9

[3] Avinor. Avinor.no, 2009. http://www.avinor.no/avinor/trafikk/50_Flydata (2010-04-29). 12

[4] Robert Battle og Edward Benson. Bridging the semantic web and web 2.0 with representational state transfer (REST). *Web Semant.*, 6:61–69, februar 2008. ISSN 1570-8268. 42, 43

[5] T. Berners-Lee og L. Masinter R. Fielding. Rfc 3986: Uniform resource identifier (uri): Generic syntax, 2005. http://www.ietf.org/rfc/rfc3986.txt (2011.04.14). 18

[6] Tim Berners-Lee. Linked data, juli 2006. http://www.w3.org/DesignIssues/LinkedData.html (2010.03.22). 17, 18, 21

[7] Dan Brickley. W3C Semantic Web Interest Group: Basic Geo (WGS84 lat/long) Vocabulary, januar 2003. http://www.w3.org/2003/01/geo/ (2011.04.04). 79

[8] Byantikvaren. Byantikvarens gule liste, 2005. http://www.byantikvaren.oslo.kommune.no/gul_liste/ (2010.05.07). 14

[9] Tim Berners-Lee Christian Bizer Tom Heath. Linked data - the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009. 15, 18

[10] Kendall Grant Clark, Lee Feigenbaum og Elias Torres. Sparql protocol for RDF - W3C Recommendation, januar 2008. http://www.w3.org/TR/rdf-sparql-protocol/ (2011.01.25). 35

[11] Peter Corbett. How to run your own apps for democracy innovation contest, 2011. http://www.usa.gov/webcontent/documents/create_an_apps_for_democracy%5B1%5D.pdf (2011.02.14). 9

[12] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures.* Doktorgradsoppgave, University of California, Irvine, Irvine, California, 2000. 41

[13] Fornyings-, administrasjons- og kirkedepartementet. Nettskap 2.0. http://www.regjeringen.no/nb/dep/fad/kampanjer/nettskap.html (2011.04.14). 52

[14] Fornyings-, administrasjons- og kirkedepartementet. Stortingsmelding nr. 17 - eit informasjonssamfunn for alle, desember 2006. 10

[15] Fornyings-, administrasjons- og kirkedepartementet. data.norge.no, 2009. http://data.norge.no (2010-04-29). 11, 13

[16] Fornyings-, administrasjons-, og kirkedepartementet. Fellesføringar i tildelingsbreva for 2010 - orientering om ny ordning og tinging til departementa om forslag til fellesføringar for 2010, mai 2009. Rundskriv P 2/2009. 11

[17] Fornyings- administrasjons- og kirkedepartementet. Fellesføringer i tildelingsbrevene for 2011, september 2010. Rundskriv P 5/2010. 11

[18] Gartner, Inc. Gartner highlights key predictions for IT organizations and users in 2010 and beyond, januar 2010. http://www.gartner.com/it/page.jsp?id=1278413 (2011.04.14). 47

[19] Gudrun Tokle Grene, Line Thams Reiersen, Olav Anders Øvrebø. Fakta først - viderebruk av datakilder i offentlig sektor: potensial og hindringer. Rapport, Institutt for informasjons- og medievitenskap, Universitetet i Bergen, 2010. 8, 9, 12, 13

[20] Marianne Gusgaard. Visningstjenester: Statens Kartverk, desember 2009. http://www.statkart.no/nor/Land/Kart_og_produkter/Visningstjenester (2011.04.14). 84

[21] Michael Hausenblas. Linked data applications. Rapport, DERI, 2009. 15

[22] Tom Heath og Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*, bind 1. Morgan & Claypool, 2011. 18

[23] John Hebeler, Matthew Fischer, Ryan Blace og Andrew Perez-Lopez. *Semantic Web Programming.* Wiley, 2009. 25, 30, 35, 40

[24] Pascal Hitzler, Markus Krötzsch og Sebastian Rudolph. *Foundations of Semantic Web Technologies*. CRC, 2010. 18, 23, 26, 28, 29, 32, 33, 35, 37, 38, 40

[25] Tim Berners Lee. Talk: Linked open data, 2005. http://www.w3.org/2008/Talks/0617-lod-tbl (2011.04.14). 17

[26] George F. Luger. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Pearson Education Limited, 2002. 35

[27] Frank Manola og Eric Miller. RDF primer. World Wide Web Consortium, Recommendation REC-rdf-primer-20040210, February 2004. http://www.w3.org/TR/2004/REC-rdf-primer-20040210 (2011.04.14). 23, 25, 27

[28] Moderniseringsdepartementet. eNorge 2009 - det digitale spranget. Rapport, juni 2005. 10

[29] NGA Coordinate Systems Analysis Team (CSAT). Grid coordinates - a simplified definition and explanation of utm and related systems, august 2005. http://earth-info.nga.mil/GandG/coordsys/grids/utm_basics_2005.doc (2011.04.14). 79

[30] PlanetData. Data and technology, 2011. http://planet-data.eu/data-and-technology (2011.04.29). 21

[31] Eric Prud'hommeaux og Andy Seaborne. Sparql query language for RDF - W3C Recommendation, January 2008. http://www.w3.org/TR/rdf-sparql-query/ (2011.01.26). 35, 36, 39, 74

[32] Petter Reinholdtsen. Pressemelding: Kommunegrenser uten bruksbegrensninger til alle fra NUUG, mars 2011. http://www.nuug.no/news.shtml (2011.04.14). 84, 85

[33] Matt Rosenberg. What is the distance between a degree of latitude and logitude? http://geography.about.com/library/faq/blqzdistancedegree.htm (2011.04.14). 76

[34] Leo Sauermann og Richard Cyganiak. Cool uris for the semantic web - W3C Interest Group Note, desember 2008. http://www.w3.org/TR/cooluris/ (2011.01.24). 17, 27, 29

[35] Europe's Information Society. Public sector information — raw data for new services and products. http://ec.europa.eu/information_society/policy/psi/index_en.htm (2010-04-20). 6

[36] John F. Sowa. Semantic networks, 2006. http://www.jfsowa.com/pubs/semnet.htm (2011.04.25). 19

[37] Statistisk Sentralbyrå. SSBs Metadata - Variabeldefinisjon - Organisasjonsnummer. http://www.ssb.no/metadata/conceptvariable/vardok/1350/nb(2011.04.14). 86

[38] Sinead Carew Tarmo Virki. Google topples Symbian from smartphones top spot, januar 2011. http://uk.reuters.com/article/2011/01/31/oukin-uk-google-nokia-idUKTRE70U1YT20110131 (2011.04.14). 52

[39] Joshua Tauberer. Open data is civic capital: Best practices for "open government data", 2009. http://razor.occams.info/pubdocs/opendataciviccapital.html (2011.01.29). 6

[40] Teknologirådet. Fra altinn til alt ut - offentlige data for innovasjon og demokrati. Rapport, April 2010. 12, 13

[41] The United States Government. Data.gov - about, 2009. http://www.data.gov/about (2010-04-21). 8

[42] Chris Veness. Calculate distance and bearing between to latitude/longitude points using haversine formula in javascript, 2002. http://www.movable-type.co.uk/scripts/latlong.html (2011.04.14). 71, 72

[43] S. Vinoski. Restful web services development checklist. *Internet Computing, IEEE*, 12(6):96 –95, nov.-des. 2008. ISSN 1089-7801. 59

[44] Wikipedia. Geographic coordinate system — Wikipedia, the free encyclopedia, 2011. http://en.wikipedia.org/wiki/Coordinates_%28geographic%29 (2011.04.27). 75

[45] Erik Wilde og Michael Hausenblas. RESTful SPARQL? you name it!: aligning SPARQL with REST and resource orientation. I *Proceedings of the 4th Workshop on Emerging Web Services Technology*, WEWST '09, side 39–43, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-776-9. 41, 43