

Semantic Web teknologier

RDF – Resource Description Framework

Whitepaper af

Henrik Hvid Jensen

Vidensleverandør og forfatter

SOA Network

Henrikhvid@soanetwork.dk

November 2004

"URI giver mulighed for at vi ved præcist hvad vi snakker om, men det der mangler er muligheden for også at gøre dette forståeligt for computere. En ting er at sige "Jeg holder meget af at spille golf" i et diskussionsforum. Noget helt andet er, at få en computer til at "forstå" hvad man siger."

"RDF giver mulighed for at gøre sætninger tilgængelige for computere. En RDF ligner meget almindelige sætninger på nær at næsten alle ord er URIer."

Indholdsfortegnelse

Kapitel 1 RDF (Resource Description Framework)	3
1.1 RDF Graph	5
1.1.1. Datatyper i RDF.....	8
1.2 Andre relevante funktionaliteter med RDF	8
1.2.1. RDF/XML	8
1.2.2. Eksempler	10
1.3 Opbygning af Vokabularer	12
1.4 RDF Schema.....	12
1.4.1. Beskrivelse af egenskaber.....	13
1.5 Opsummering på RDF.....	14

Figurer

Figur 1-1 Regler RDF er bygget på	3
Figur 1-2 RDF graphs modellering af sætninger	5
Figur 1-3 Simpel RDF sætning.....	5
Figur 1-4 Definition af Creator.....	6
Figur 1-5 Flere sætninger om den samme ressource.	6
Figur 1-6 Yderligere information om forfatteren.	7
Figur 1-7 Temperatur i København er 25 grader celcius	9
Figur 1-8 Tilsvarende triple	9
Figur 1-9 Tilsvarende RDF/XML beskrivelse.....	9
Figur 1-10 RDFs elementer	10
Figur 1-11 Personligt brug af RDF	11
Figur 1-12 RDF beskrivelsen	11
Figur 1-13 rdfs:domain og rdfs:range.....	13
Figur 1-15 Semantic Web indeholder A:Tabeller, B:Træer, C:Alt.....	15

Eksempler

Eksempel 1-1 Simpel sætning med RDF	4
Eksempel 1-2 RDF med XML syntaks.....	4
Eksempel 1-3 triple beskrivelse af sætningerne.	7
Eksempel 1-4 Ophavsmann er det samme som creator	8
Eksempel 1-5 Brug af en datatypeURI	8
Eksempel 1-6 Brug af rdf:type	12
Eksempel 1-7 Brug af rdf:Property.....	13
Eksempel 1-8 Brug af rdfs:range	13
Eksempel 1-9 Brug af rdfs:Datatype	14
Eksempel 1-10 Brug af rdfs:domain	14
Eksempel 1-11 Beskrivelse af farven på ens bil	14

Tabeller

Tabel 1-1 RDFs design karakterstika (Kilde [WIRDF]).....	4
--	---

Kapitel 1 RDF (Resource Description Framework)

Resource Description Framework (RDF) var udviklet af W3C omkring samme tid som XML og det viste sig at være et godt supplement til XML. RDF er et XML format, det giver et sprog til at beskrive og udveksle metadata og muliggør applikationer til styring af viden.

RDF er bygget efter følgende regler

1. En **Ressource** (eng: resource) er alt som kan have en URI, det inkluderer alle web sider såvel som individuelle elementer i et XML dokument.
2. En **egenskab** (eng: property) er en ressource, som har et navn og kan bruges som en egenskab f.eks. forfatter eller titel
3. Et **udsagn/sætning** (eng: statement) består af en kombination af en ressource, en egenskab og en værdi. Disse dele er kendt som en sætnings *subjekt/grundled, prædikat/udsagnsled* og *objekt/genstandsled*. Et eksempel på en sætningen er: "Forfatteren til www.henrikhvid.dk/analyser/semanticweb.html er Henrik Hvid Jensen". Værdien kan være en tekst streng som "Henrik Hvid Jensen" eller det kan være en anden ressource for eksempel "Hjemmesiden for www.henrikhvid.dk/analyser/semanticweb.html er www.henrikhvid.dk/"
4. Der er en ligetil metode at udtrykke disse abstrakte egenskaber i XML, for eksempel:

```
<rdf:Description about="http://www.henrikhvid.dk/analyser/semanticweb.html">
    <forfatter>Henrik Hvid Jensen</forfatter>
    <hjemmeside>rdf:ressource="http://www.henrikhvid.dk"/>
</rdf:Description>
```

Kilde [WIRDF]

Figur 1-1 Regler RDF er bygget på

RDF er designet til at have karakteristika angivet i Tabel 1-1

Uafhængighed	Da et udsagn er en ressource, kan uafhængige organisationer eller personer selv opfinde dem.
Udveksling	Da RDF sætninger kan konverteres til XML, er de lette at udveksle.
Skalerbarhed	RDF sætninger er simple, tredelte poster (grundled, udsagnsled og genstandsled), så de er lette at håndtere og slå ting op efter, selv ved store antal. Internettet er allerede stort og bliver kontant større. [WIRDF] vurderer at der vil være milliarder spredt rundt omkring. Selv for et Intranet vil der være millioner, skalerbarhed er derfor vigtigt.
Egenskaber er ressourcer.	Egenskaber kan have deres egne egenskaber og kan blive fundet og behandlet lige som enhver anden ressource. Det er vigtigt, fordi der vil være mange af dem, alt for mange til at

	undersøge en af gangen.
Værdier kan være ressourcer.	For eksempel vil de fleste web sider have en egenskab der hedder hjemmeside, som peger på sitens hjemmeside. Så værdien af egenskaberne, som må inkludere ting som titel og forfatter, kan også inkludere ressourcer.
Udsagn kan være ressourcer.	Udsagn kan også have egenskaber. Da der ikke vil være nogen overordnet organisation, som leverer brugbar beskrivelse af alle ressourcer og da internettet er alt for stort, til at man kan bygge sin egen, er det nødvendigt at foretage opslag baseret på andre personers metadata (det er hvad Yahoo! tilbyder i dag). Det betyder at man for et udsagn som ”Emnet for denne side er golf” skal have mulighed for at spørge ”Hvem sagde det? Hvornår?”. En brugbar måde vil være med metadata, hvorfor udsagn også har behov for egenskaber.

Tabel 1-1 RDFs design karakterstika (Kilde [WIRDF])

URI giver mulighed for at vi ved præcist hvad vi snakker om, men det der mangler er muligheden for også at gøre dette forståeligt for computere. En ting er at sige ”Jeg holder meget af at spille golf” i et diskussionsforum. Noget helt andet er, at få en computer til at ”forstå” hvad man siger.

RDF giver mulighed for at gøre sætninger tilgængelige for computerer. En RDF ligner meget almindelige sætninger på nær at næsten alle ord er URIer. Hvis vi ser på Eksempel 1-1, står der ”Henrik Hvid holder meget af at spille golf i Fredensborg Golf klub (FGC)”. Hvis FGC har defineret begrebet ”spille golf hos FGC” ved brug af en global standard for golfdudtryk, f.eks. en defineret hos Royal & Ancients (<http://www.randa.org/terms/play>), vil sætningen kunne forstås af alle uafhængig af sprog.

< http://henrikhvid.dk/ >	< http://like.example.org/terms/reallyLikes >	< http://www.fgc.dk/begreber/spille >
subjekt/grundled	prædikat/udsagnsled	objekt/genstandsled

Eksempel 1-1 Simpel sætning med RDF

Der er ikke nogen officiel web side der udtrykker alt hvad jeg ønsker at sige, i stedet er informationen spredt ud over hele internettet.

Dette eksempel er beskrevet i Notation3 (N3), et sprog som tillader en at skrive simple RDF sætninger. Den officielle RDF specifikation definerer en XML repræsentation af RDF.

```

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntaks-ns#"
  xmlns:like="http://like.example.org/terms/"
>
<rdf:Description rdf:about="http://henrikhvid.dk/">
<like:reallyLikes
  rdf:resource="http://www.fgc.dk/begreber/spille"/>
</rdf:Description>
</rdf:RDF>

```

Eksempel 1-2 RDF med XML syntaks

Læg især mærke til hvorledes alt er identificeret ved hjælp af URI..

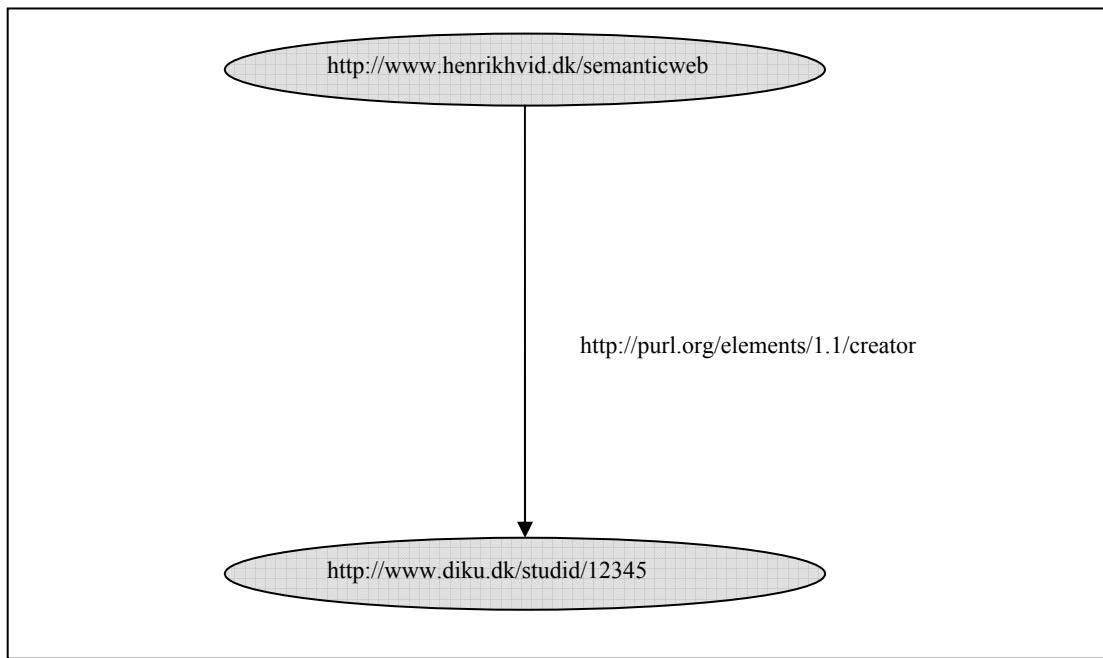
1.1 RDF Graph

RDF modellerer sætninger som punkter og pile. I denne model er sætninger repræsenteret ved

- En knude for dets grundled, opmærket med dets URIref
- En knude for dets genstandsled, opmærket med dets URIref
- En pil for dets udsagsnsled, opmærket med dets URIref og med retningen fra grundleddet til genstandsleddet.

Figur 1-2 RDF graphs modellering af sætninger

Det vil sige at hvis dette dokument er identificeret ved URIen <http://www.henrikhvid.dk/semanticweb>, vil en sætning som ”Dokumentet ”Analyse af The Semantic Webs betydning for virksomhederne og de teknologiske standarder” har en forfatter med en værdi af Henrik Hvid Jensen” blive repræsenteret af en graf som vist i Figur 1-3.



Figur 1-3 Simpel RDF sætning

Udsagsnsleddet/prædikatet er en metadata term defineret af Dublin Core initiativet, se Figur 1-4. For en komplet oversigt over Dublin Core termer henvises til **Fejl! Henvisningskilde ikke fundet.**, **Fejl! Henvisningskilde ikke fundet.** og **Fejl! Henvisningskilde ikke fundet.**

```

<rdf:Property rdf:about="http://purl.org/dc/elements/1.1/creator">
    <rdfs:label xml:lang="en-US">Creator</rdfs:label>
    <rdfs:comment xml:lang="en-US">An entity primarily responsible for making the content of the resource.</rdfs:comment>
    <dc:description xml:lang="en-US">Examples of a Creator include a person, an organisation, or a service. Typically, the name of a Creator should be used to indicate the entity.</dc:description>
    <rdfs:isDefinedBy rdf:resource="http://purl.org/dc/elements/1.1/" />
    <dcterms:issued>1999-07-02</dcterms:issued>
    <dcterms:modified>2002-10-04</dcterms:modified>
    <dc:type rdf:resource="http://dublincore.org/usage/documents/principles/#element" />
    <dcterms:hasVersion rdf:resource="http://dublincore.org/usage/terms/history/#creator-004" />
</rdf:Property>
  
```

Figur 1-4 Definition af Creator

Genstandsleddet/objektet er identificeret som en studerende hos DIKU med studie ID 12345. Det kan selvfølgelig diskuteres, om dette er det rette sted at pege hen, da den studerende på et tidspunkt holder op med at studere på DIKU og derved skal DIKU vedligeholde denne information også efter at den studerende er stoppet. En anden løsning kunne være at f.eks. CPR registeret lavede sådanne URI for alle borgere i Danmark. Det vil måske stadig give god mening for DIKU at have deres egne URI for dets studerende, men så længe de er defineret ved hjælp af den overordnede URI, vil det stadig være brugbart også ud i fremtiden.

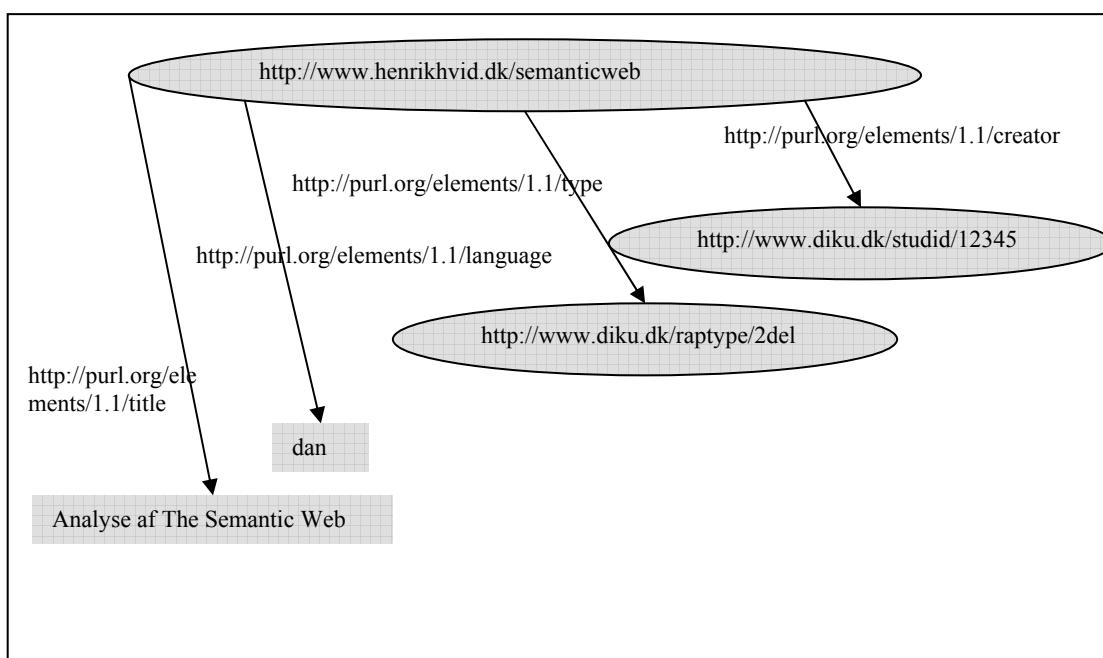
Grupper af sætninger er repræsenteret ved tilsvarende grupper af knuder og pile. Så hvis vi også ønskede at udtrykke følgende sætninger:

"http://www.henrikhvid.dk/semanticweb er af typen 2. dels rapport"

"http://www.henrikhvid.dk/semanticweb er skrevet på dansk"

"http://www.henrikhvid.dk/semanticweb har titlen "Analyse af The Semantic web""

kan det gøres som illustreret i Figur 1-5.



Figur 1-5 Flere sætninger om den samme ressource.

Her kan det også ses at RDF sætningers objekter enten kan være ressourcer identificeret ved URIer eller konstanter repræsenteret ved en tekststreng.

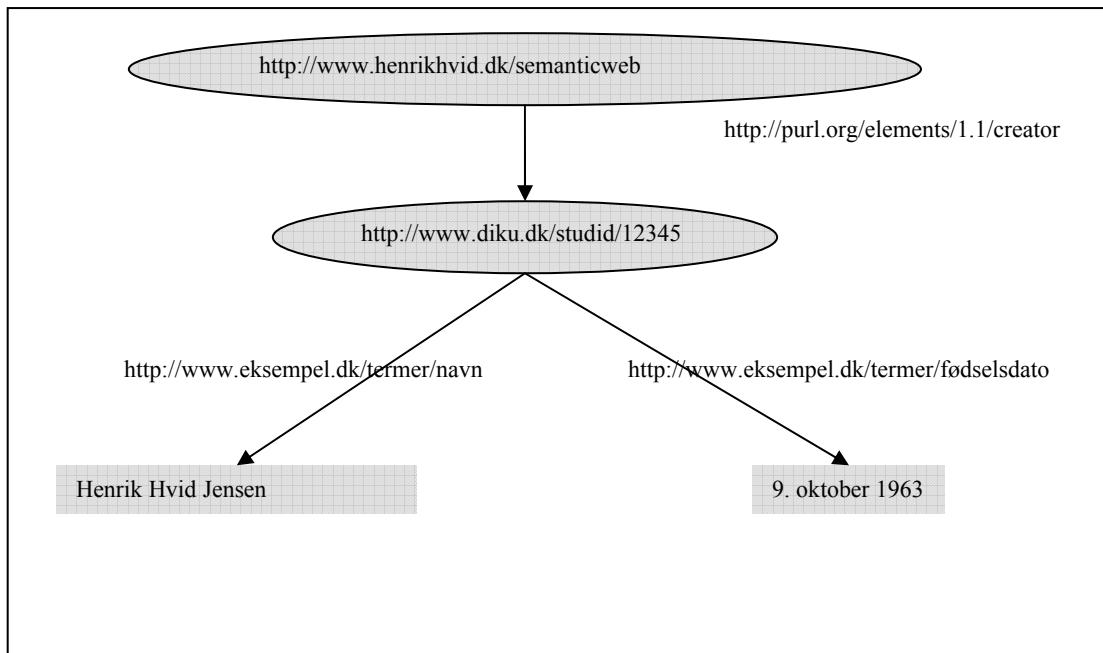
En alternativ måde at repræsentere sætninger på i RDF er at lave såkaldte *"triples"* (beskrevet i N3). I denne notation er hver sætning skrevet som en simpel trefoldighed af grundled/subjekt, udsagnsled/prædikat og genstandsled/objekt betegnelser. De *triple* der beskriver sætningerne i Figur 1-5 beskrives som vist i Eksempel 1-3.

```
<http://www.henrikhvid.dk/semanticweb> <http://purl.org/elements/1.1/creator>
<http://www.diku.dk/studid/12345>
<http://www.henrikhvid.dk/semanticweb> <http://purl.org/elements/1.1/type>
<http://www.diku.dk/raptype/2del>
<http://www.henrikhvid.dk/semanticweb> <http://purl.org/elements/1.1/language> "dan"
<http://www.henrikhvid.dk/semanticweb> <http://purl.org/elements/1.1/title> "Analyse af The
Semantic Web"
```

Eksempel 1-3 triple beskrivelse af sætningerne.

Disse sætninger kan blive meget lange hvorfor brug af namespaces er relevant. I resten af rapporten vil namespaces blive brugt uden de er foruddefineret. De bruges primært til at begrænse størrelsen og give en ide om hvor begreber og ressourcer evt. kunne blive defineret.

Ovenstående sætninger illustrerer noget af styrken ved at bruge URIref. F.eks. er forfatteren identificeret ved en <http://www.diku.dk/studid/12345> i stedet for bare navnet "Henrik Hvid Jensen", som er en væsentlig mere upræcis identifikation. Da forfatteren er identificeret som en rigtig ressource, kan vi registrere yderligere information omkring ham (Figur 1-6).

**Figur 1-6 Yderligere information om forfatteren.**

Figur 1-6 viser også hvorledes RDF bruger URIref til prædikatet/udsagnsleddet i stedet for at bruge tekststrenge som "navn" og "fødselsdato". Styrken ved dette er at det gør det muligt at skelne mellem de egenskaber vi bruger, fra egenskaber andre må bruge. Vi bruger f.eks. "navn" til at mene forfatterens navn, men andre kunne godt bruge "navn" til noget andet. F.eks. navnet på en variabel i et program. Et program som støder på udtrykket "navn" vil ikke nødvendigvis kunne skelne mellem betydningen. Men hvis man bruger henholdsvis <http://www.eksempel.dk/termer/navn> og <http://www.medicin.dk/termer/navn>, vil det være simpelt at forstå, at der er forskellige betydninger for de to begreber. Selv om en computer måske ikke kan forstå betydningen, kan den i hvert fald kende forskel. En anden vigtig faktor ved at definere prædikatet som en ressource er, at man kan lave yderligere RDF sætninger, som beskriver denne ressource.

Ved at bruge URIref for såvel subjekt/grundled, prædikat/udsagnsled og objekt/genstandsled i RDF sætninger, giver det mulighed for, at man kan udvikle og bruge delte vokabularer på nettet, som underbygger en fælles forståelse af de koncepter der behandles.

Det betyder at ethvert program der forstår hvad <http://purl.org/elements/1.1/creator> betyder, kan, uafhængigt af hvilket sprog web siden ellers er i, finde hvem forfatteren er. Dette gælder selvfølgelig også hvis web siden bruger lokale betegnelser som f.eks.

”ophavsmand”. Bare ”ophavsmand” et eller andet sted er defineret til at være lig ”creator”, kan en Semantic Web software agent selv ræsonnere, at de to begreber er identiske.

www.termer.dk/ophavsmand	www.vocabular.uk>equals ¹	http://purl.org/elements/1.1/creator
--------------------------	--------------------------------------	--------------------------------------

Eksempel 1-4 Ophavsmand er det samme som creator

1.1.1. Datatyper i RDF

I modsætning til typiske programmeringssprog og database systemer har RDF ingen indbyggede datatyper, såsom integer, strings, date. I stedet bruges datatyper defineret andre steder og som kan identificeres af en datatype URI (se Eksempel 1-5).

< http://www.diku.dk/studid/12345 >	< http://www.eksempel.dk/termer/fødselsdato >	”1963-10-09”^^xsd:date
xsd: er namespace for	http://www.w3.org/2001/XMLSchema#	

Eksempel 1-5 Brug af en datatypeURI

RDF laver ikke noget check af om dataværdien overholder den definerede datatyper, det er helt op til det program, som behandler RDF sætningen. Det er forventningen at datatyperne defineret til XML Schema, vil være de mest brugte og derfor dem, der er størst sandsynlig for, vil være interoperable mellem forskelligt software. Det må derfor også forventes at programmer til behandling af RDF, vil kunne genkende disse datatyper.

1.2 Andre relevante funktionaliteter med RDF

RDF er altså simple punkt og pile grafer, som fortolkes som sætninger om ting identificeret ved URIref.

Formålet med dette kapitel er ikke at lave en komplet gennemgang af RDF, men mere at sikre en forståelse af konceptet, som det er gjort med de forrige afsnit.

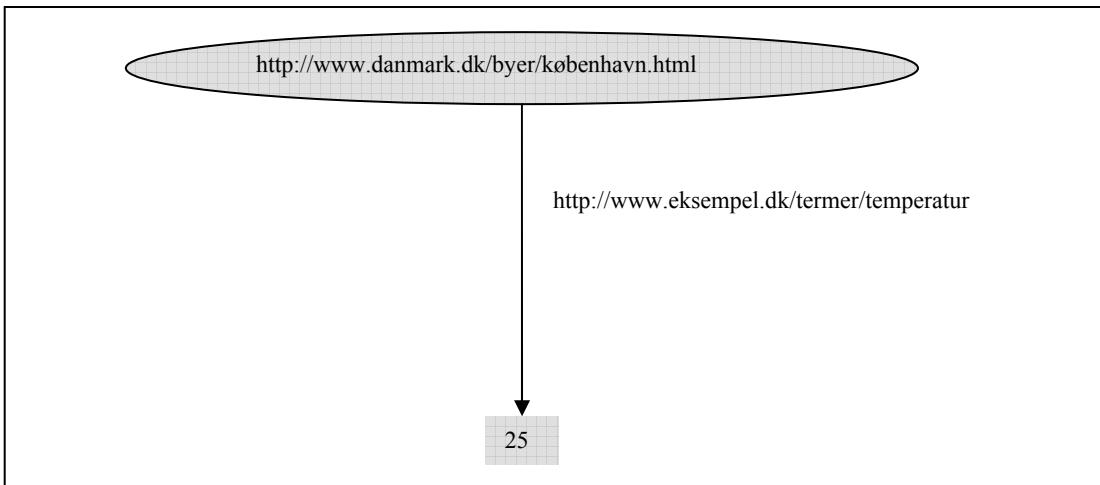
For at sikre et overblik over RDFs andre muligheder vil dette afsnit give en overordnet gennemgang af andre væsentlige elementer i RDF.

1.2.1. RDF/XML

Som tidligere beskrevet er RDF konceptuelle model en graf. RDF leverer også en XML syntaks for at kunne skrive og udveksle RDF grafer. Denne syntaks kaldes RDF/XML. I modsætning til *triples* som er en notation til hurtigt at skrive sætninger er RDF/XML den normative/retningsgivende syntaks for at skrive RDF.

Hvis vi tager sætningen ”Temperaturen i København er 25 grader celcius”, vil den tilhørende graf være

¹ se afsnit **Fejl! Henvisningskilde ikke fundet.** for hvorledes dette vil defineres i en Semantic web ontologi



Figur 1-7 Temperatur i København er 25 grader celcius

Den tilsvarende *triple* repræsentation

DKbyer:københavn.html	ekstermer:temperatur	"25"
-----------------------	----------------------	------

Figur 1-8 Tilsvarende triple

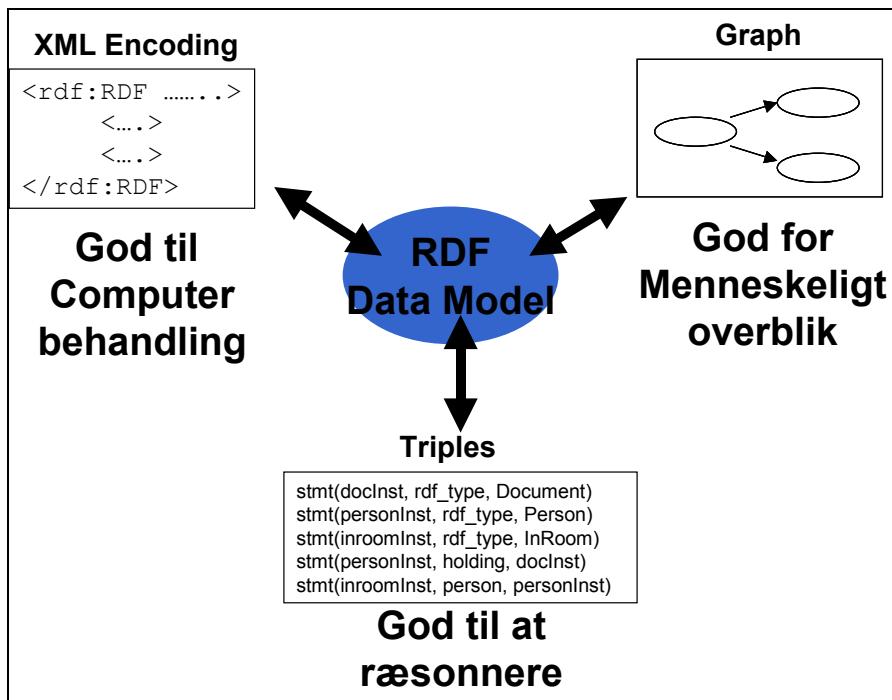
I RDF/XML syntaks beskrives det som

```
<? Xml version="1.0"?>
<rdf:RDF    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:ekstermer="http://www.eksempel.dk/termer/">
    <rdf:Description rdf:about="http://www.danmark.dk/byer/københavn.html">
        <ekstermer:temperatur>25</ekstermer:temperatur>
    </rdf:Description>
</rdf:RDF>
```

Figur 1-9 Tilsvarende RDF/XML beskrivelse.

Figur 1-9 illustrerer den basale ide brugt i RDF/XML, at indkode en RDF graf som XML elementer, attributter, element indhold og attribut værdier.

- URIref værdien for egenskaberne er skrevet som XML QNames, der er namespace kvalificeret.
- URIref værdien for subjektet/grundleddet er skrevet som en XML attribut værdi.
- Værdien for genstandsleddet bliver tekst indhold i elementet eller en attribut værdi.



Figur 1-10 RDFs elementer

1.2.2. Eksempler

Et af de basale arkitektur principper bag internettet er, at alle skal have mulighed for at sige, alt de ønsker om alle eksisterende ressourcer. F.eks. beskriver [MSW] nogle eksempler på, hvilke muligheder det giver selv at kunne tilføje metadata til eksisterende ressourcer, eksemplerne er angivet i Figur 1-11.

Indholds afhængige links

Forestil dig at man browser en web side der bruger begrebet ”reifikation” og du ønsker at forstå hvad det betyder. Du ved også at man kan slå et vilkårligt ord op ved at bruge en URL som <http://www.ordbog.dk/cgi-bin/ordb.pl?begreb=reifikation>. Hvis forfatteren af web siden havde tilføjet et hyperlink som dette til ordet ”reifikation”, ville man kunne klikke på det for at slå det op. Men hvorfor være afhængig af forfatteren? Hvorfor kan ens web browser ikke bare fremhæve et ord og så tillade at man vælger en kommando for at definere ordet, som i ovenstående. Så ville man automatisk blive ført hen til definitionen af ordet ”reifikation”. Forestil dig at web browseren kunne gå et trin videre og vide at begrebet ”Grateful Dead” refererer til en musikgruppe. Det kunne så linke til RollingStone.com, selv hvis den oprindelige forfatter ikke havde gjort det.

Site information

Man browser på en side som www.cw.dk og en browser har mulighed for at bestemme, at det er en nyheds site og levere en mulighed på menuen om, at man kan få mere information om IDG (ejeren af cw.dk).

Det er meget lig det første eksempel, men her har browseren kendskab til hele siten eller domænet i stedet for enkelte ord på siden.

Filtrering samarbejde

Forstil også at ens browser har 2 knapper en ”thumbs up” og en ”thumbs down” knap. Hver gang man besøger en hjemmeside, der er dårlig, trykker man på ”thumbs down” og hver gang man støder på en, man synes er rigtig god, trykker man på ”thumbs up”. Hver gang man trykker på knappen, bliver ens holdning registreret et eller andet sted.

Når man så browser hen til et sted, kan ens browser give noget relevant information såsom ”80 % af dine venner gav denne side ”thumbs up” eller ”90 % af dine venner siger at denne side er dårlig”. Google kunne bruge sådan information til at rangere søgeresultater eller man kunne lave egne rangeringer.

Kategoriserings samarbejde

Forstil dig, at man ser en side, der giver en god beskrivelse af delfiner. Man ønsker at registrere den for senere brug. Så man går til ens browser ”Min encyklopædi” menu og vælger ”tilføj side”. Man bliver bedt om at vælge kategorier for denne encyklopædi artikel (f.eks. ”Havets pattedyr”, ”delfin”). Denne metadata gemmes et centralet sted. Nu kan alle gå til ”Min encyklopædi” menu i deres browser og finde artiklen i ”Havets pattedyr ” og ”Delfin” kategorierne.

Kommentarer

Man kunne også forestille sig, at man browser på hjemmesiden, hvor man ønsker at inkludere nogle relevante kommentarer. Man vil sandsynligvis ikke have mulighed for at tilføje noget til den enkelte hjemmeside, så ens kommentarer kunne sendes til en *kommentarserver* et andet sted. Når ens venner så browser denne hjemmeside, vil de kunne se ens tilføjelser. Fordi deres browser er sat op til automatisk at kontakte kommentarserveren for at finde deres venner kommentarer.

Figur 1-11 Personligt brug af RDF

Hvis man prøver at angive ovenstående information i RDF kunne det se ud som Figur 1-12

Indholds afhængige links

Ordbog:reifikation eksterm:forklarer ”reifikation”
Ordbog:reifikation eksterm:betyder ”tingsliggørelse”

Site information

```
<http://www.cw.dk> eksterm:ejes_af "http://www.krak.dk/idg"  
<http://www.cw.dk> eksterm:type eksterm:nyheder
```

Filtrering samarbejde

```
<http://www.cpr.dk/123456-1234> eksterm:thumbs_up <http://www.cw.dk/artikel>
```

Kategoriserings samarbejde

```
<http://www.zoo.dk/delfiner> eksterm:kategoriseres eksterm:delfiner  
<eksterm:delfiner> eksterm:hørerUnder eksterm:havpattedyr
```

Kommentarer

```
<?xml version="1.0"?>  
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:kommentar="http://www.webkommentar.dk/termer/">  
<rdf:Description rdf:about="http://www.cw.dk/SWartikel">  
    <kommentar:af>http://www.cpr.dk/123456-1234 </kommentar:af>  
    <kommentar:kommentar> "Jeg er meget uenig i konklusionen, der er  
        ingen tvivl om ..." </kommentar:kommentar>  
</rdf:Description>  
</rdf:RDF>
```

Figur 1-12 RDF beskrivelsen

Eksemplerne illustrerer også, at RDF'er der beskriver en bestemt ressource, ikke alle behøver at være placeret på samme sted, i stedet kan de være distribueret ud over hele nettet.

1.3 Opbygning af Vokabularer

RDF leverer altså en model for metadata og en syntaks så uafhængige deltagere kan udveksle og bruge denne metadata. Det leverer ikke egne udtryk såsom forfatter, instruktør eller forretningskategorier. At få opbygget disse vokabularer, vil være en opgave for alle og det må forventes at primært industriorganisationer, standardorganisationer og lignende vil levere disse udtryk. IFX kan levere udtryk til finansverdenen, Papinet til papirindustrien osv.

1.4 RDF Schema

RDF tilbyder en måde at udtrykke simple sætninger omkring ressourcer, ved at bruge navngivne egenskaber og værdier. Det er dog også nødvendigt at kunne fortælle hvilken type eller klasse ressource, det drejer sig om.

RDF Schema ([RDF Schema Candidate Recommendation](#)) er designet til at være en simpel datatype model for RDF. Ved brug af RDF Schema kan man sige at "Fido" er af typen "Hund" og at "Hund" er en underklasse til "dyr".

RDF Schema giver mulighed for at definere klasser og egenskaber til disse klasser. F.eks. kan man definere en klasse, der hedder *rapport* eller *artikel* og bruge egenskaber såsom *titel*, *forfatter*, *emne*.

RDF Schema tilbyder ikke selv vokabularer, men tilbyder mekanismene til at opbygge klasser og egenskaber i vokabularer og til at fortælle hvilke klasser og egenskaber, der skal bruges sammen.

"rdfs:" bruges ofte som alias for RDF Schema, hvilket også vil ske i denne rapport.

De tre vigtigste koncepter som RDF og RDF schema giver er "Ressource" (rdfs:Resource), "klassen" (rdfs:Class), og "egenskab" (rdf:Property). Disse er alle klasser, fordi udtryk kan tilhører disse. F.eks. er alle udtryk i RDF af typen ressource. For at kunne sige at noget er en type af noget andet, bruges rdf:type egenskaben.

```
rdfs:Resource rdf:type rdfs:Class .
rdfs:Class rdf:type rdfs:Class .
rdf:Property rdf:type rdfs:Class .
rdf:type rdf:type rdf:Property .
```

Eksempel 1-6 Brug af rdf:type

Der er rimeligt simpelt at lave ens egne klasser. Hvis vi f.eks. ønsker at oprette en klasse "Idræt", som vil indeholde alle sportsgrene i verdenen:

```
kum:Idræt rdf:type rdfs:Class
```

Nu er det muligt at sige at "Fodbold" er af typen "Idræt"

```
DBU:Fodbold rdf:type kum:Idræt
```

Man kan lave egenskaber ved at give det en URI og definere det til at være af typen rdf:Property. Derefter kan disse egenskaber bruges i ens RDF.

```
Eks:kaldes rdf:type rdf:Property
DBU:Fodbold eks:kaldes "Fodbold"
```

Læg mærke til at det er nødvendigt at fortælle at DBU:Fodbold kaldes "Fodbold". Dette skyldes at DBU:Fodbold jo kun er en URI, ganske vist udvalgt med omhu, så den er lettere at huske for mennesker, men for en computer kunne der lige så godt have stået DBU:hklsaher. Så selv om mennesker kan gætte, at vi beskæftiger os med noget der kaldes fodbold, kan computer det ikke. Det bliver vi nødt til at fortælle den.

Rdfs:subClassOf er en anden relevant egenskaber. Man kan for eksempel sige at klassen ”Idræt” er en underklasse af klassen ”Kulturområde”. Dette gøres ved:

Kum:Kulturområde	rdf:type	rdfs:Class
kum:Idræt	rdfs:subClassOf	kum:Kulturområde

Det betyder, at når man siger, at fodbold er en idræt siger man også at fodbold er et kulturområde. Det er også muligt at angive andre underklasser af Kulturområde:

kum:Teater	rdfs:subClassOf	kum:Kulturområde
kum:Film	rdfs:subClassOf	kum:Kulturområde

Hvis man har dannet følgende instanser og egenskaber:

Cpr:123456-1234	rdf:type	eks:menneske
http://www.kgl-teater.dk/	rdf:type	kum:Teater
eks:besøgt	rdf:type	rdf:Property
DBU:spilleri	rdf:type	rdf:Property
DBU:superliga	rdf:type	DBU:fodbold

kan man registrere yderligere information:

cpr:123456-1234	eks:besøgt	http://www.kgl-teater.dk/
http://www.kgl-teater.dk/	eks:kaldes	”Det Kongelige Teater”
cpr:123456-1234	eks:aktiv	DBU:Superliga
cpr:123456-1234	eks:kaldes	”Per Jensen”

Som det kan ses er RDF Schema rimeligt simpelt, men tillader alligevel at opbygge vidensbaser af data i RDF meget hurtigt.

1.4.1. Beskrivelse af egenskaber

RDF Schema tilbyder også vokabular til beskrivelse af, hvorledes egenskaber og klasser skal bruges sammen i RDF data. Den vigtigste information af denne type er leveret ved de RDFS definerede egenskaber rdfs:range og rdfs:domain. Det giver mulighed for at fortælle hvilke klasser subjektet og objektet til hver egenskab må tilhører.

Alle egenskaber i RDF er beskrevet som værende af klassen rdf:Property. Så en ny egenskab som *dommer* defineres som

DBU:dommer	rdf:type	rdf Property
------------	----------	--------------

Eksempel 1-7 Brug af rdf:Property

rdfs:domain fortæller hvilken klasse subjektet i en triple, som bruger den egenskab, hører til

rdfs:range fortæller hvilken klasse objektet af en triple, der bruger den egenskab, hører til.

Figur 1-13 rdfs:domain og rdfs:range

Vi kan altså definere

Eks:menneske	rdf:type	rdfs:Class
DBU:dommer	rdfs:range	eks:menneske
Eks:alder	rdfs:range	xsd:integer

Eksempel 1-8 Brug af rdfs:range

Datatypen xsd:integer er identificeret ved dets URIref (<http://www.w3.org/2001/XMLSchema#integer>). Denne URIref kan bruges uden eksplisit at angive i RDF Schema, at det angiver en datatype. Det er ofte

hensigtsmæssigt eksplisit at angive at en given URIref identifierer en datatype. Dette kan gøres ved at bruge den RDFS definerede klasse rdfs:Datatype. For at fortælle at xsd:integer er en datatype, kan man skrive følgende RDF sætning.

xsd:integer	rdf:type	rdfs:Datatype
-------------	----------	---------------

Eksempel 1-9 Brug af rdfs:Datatype

Hvis man vil fortælle, at egenskaben DBU:dommer hører til instanser af klassen DBU:fodbold, vil man skrive følgende RDF sætning

DBU:dommer	rdfs:domain	DBU:fodbold
------------	-------------	-------------

Eksempel 1-10 Brug af rdfs:domain

1.5 Opsummering på RDF

Vi har med ovenstående gennemgang forstået, at for fremtiden skal vi ikke mere bruge udtrykket ”farve” brug i stedet udtrykket

http://www.pantomime.com/2002/std6#color

eller i N3

min:Bil pan:color "blue246"

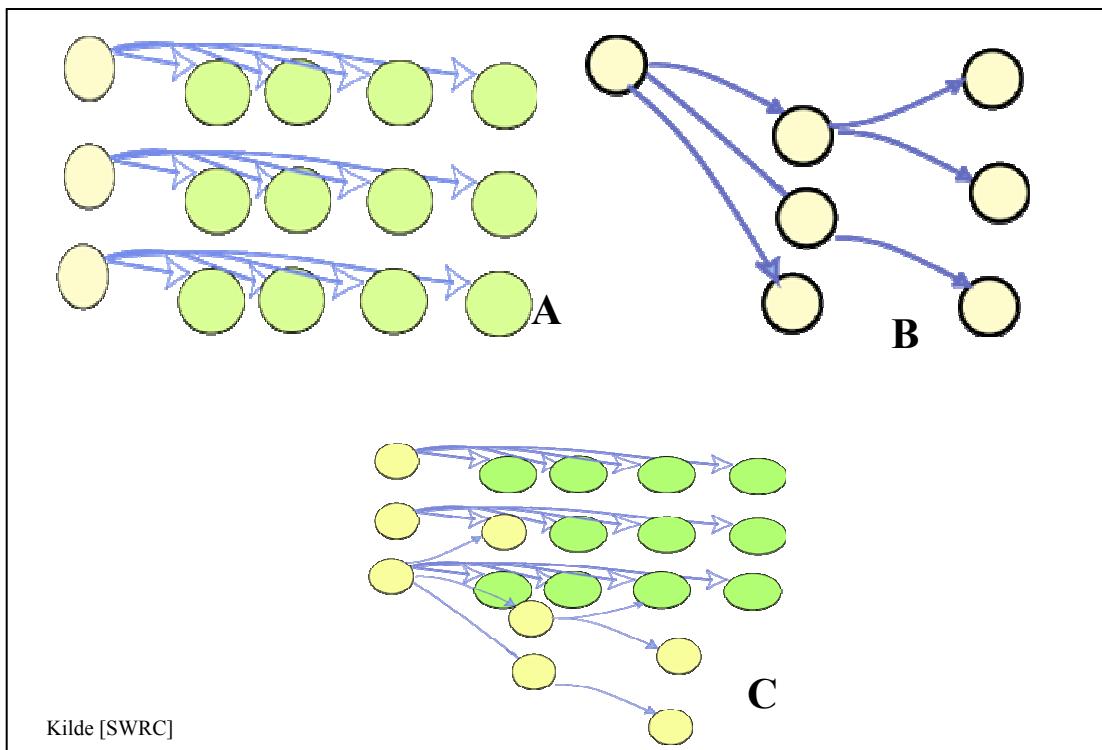
Eksempel 1-11 Beskrivelse af farven på ens bil

Vi har med RDF et udtryk der ligesom relationelle databaser indeholder emne, egenskab og værdi. Det kan kodes i XML og er simpelt og matematisk konsistent.

Semantic Web indeholder tabeller og træer alt kan beskrives i RDF Figur 1-15.

	Egenskab	
Emne	Værdi	

Figur 1-14 En relationel database indeholder også emne, egenskab og værdi



Figur 1-15 Semantic Web indeholder A:Tabeller, B:Træer, C:Alt

Man har med RDF opnået et standardformat til interoperabilitet mellem applikationer. RDF Schema laget giver os en minimalistisk men brugbar model, som indeholder alle de funktionaliteter, der er nødvendige for at kunne give interoperabilitet af den store mængde af data, der eksisterer på internettet. RDF Schema laget giver mulighed for at danne vokabularer definitioner og schema for eksisterende og nye applikationer