

Informationsstandardernas ABC

Inlägg vid Stiftelsen för rättsinformations workshop på temat
Informationsstandarder och rättslig informationsförsörjning

2007-04-24

Anders Sundström
anders.sundstrom@sunstone.se

Sunstone Systems AB

Bakgrund – Struktur v.s. Layout

I ett dokument tryckt på papper används layout för att:

- Det skall se snyggt ut (t.ex. högerjustering)
- Lyfta fram dokumentets struktur (t.ex. **rubriker**)
- Uttrycka betydelse (t.ex. *kursivering*)

Inom olika fackområden finns formella eller informella ”regler” som gör att ”alla vet” vad olika layout betyder. Detta gäller inte minst beträffande rättsinformation.

Vid elektronisk hantering av dokument behöver dessa formella och informella regelverk dels anpassas för det elektroniska mediets ”krav och möjligheter”, dels göras begripliga för maskiner så att den elektroniska hanteringen kan ge ett mervärde.

Det är detta som dagens workshop handlar om!

Lösning – Struktur v.s. Layout

Många fördelar ges av:

- Uppmärkning vid författandet
 - av stukturen
 - inte layouten
- Layout påförs vid distribution / publicering
 - Anpassad för distributionskanalen

Möjliggör korrekt publicering av texten på många olika sätt (tryck, webb, mobil, databaser, ...) men gör att författaren tappar kontrollen över det slutliga utseendet.

Det sistnämnda är en svårighet som inte skall underskattas!

Stor utmaning för dokumentförfattaren att fokusera på uppmärkning av struktur och betydelse i stället för utseende.

Även stor utmaning att ta fram och etablera lämpligt arbetsverktyg för författaren (jfr Microsoft Word).

XML – självklart men endast ett (litet) steg på vägen ...

XML (eXtensible Markup Language) är idag förmodligen det självklara valet, f.f.a. därför att det fått sådant tekniskt genomslag.

(Men ”den stora grejen” är e.g. att över huvud taget använda ett märkspråk och med dess hjälp strukturmarkera sin text. XML är ”bara” ett lämpligt verktyg för att göra detta.)

Men, att säga att ”det skall vara XML” är ungefär som att säga att ”det skall vara på papper”, dvs man är långt från mål vad gäller att ta fram kompletta ”elektroniska skrivregler”.

Många IT-system kan hantera XML – i alla fall på något sätt – men därmed inte säkert på det sätt man som användare önskar! Även här säger XML inte mer än ”på papper” ...

XML – med det finns alternativ såklart ...

Lagring av textfragment i databas (t.ex. SQL) som håller reda på all struktur och alla relationer mellan fragmenten. Detta är dock främst en metod som kan vara lämplig för att hantera (f.f.a. producera) dokument internt inom ett system eller en organisation, men inte så lämpligt för publicering av dokument, som görs bättre i XML.

Microsoft Word 2003 kan hantera XML dels enligt dess egen struktur (WordML, som har sin utgångspunkt i det ”vanliga” doc-formatet), dels med ”custom XML schema”.

PDF – Portable Document Format – Bra på många sätt och vis, men INTE vad gäller att hantera strukturerad text eftersom fokus ligger på hantering av utseende.

XML & HTML – återblick

SGML - Standard Generalized Markup Language

- Optimerat för att underlätta manuell “stansning” direkt i SGML från texteditor...
- ... vilket dock gör det krångligare för maskinen att läsa och tolka
- Många avancerade finesser som krånglar till det till ganska liten nytta

*XML förenkling av SGML ...
... men och utvidgning!*

HTML – HyperText Markup Language

- Bygger (eller snarare byggde) på SGML
- HTML var tänkt för strukturmärkning men ”spårade ur” och landade i layoutmärkning
- Avsaknad av strikt validering (i praktiken) har gjort att HTML divergerat till massor med dialekter vilket gör det svårt att bygga såväl webbläsare som applikationer eftersom bådadera måste vara ”dialektresistent”.
- Webbläsare dessutom mycket förlåtande mot syntaxfel vilket är bra på många sätt men också väldigt dåligt

XML & HTML – exempel

HTML: Funkar defacto nästan hur illa man än skrivit, såväl i teorin (pga SGML) som i praktiken (pga förlåtande och "vilt gissande" browsers).

```
<HTML><body>  
Ett enkelt exempel med två stycken.<p>  
Hanteras olika i olika märkspråk.  
</html>
```

SGML: Avancerade regler gör att maskinen ser taggar som man inte skrivit ut.

```
<HTML><body>  
<p>Ett enkelt exempel med två stycken.</p>  
<p>Hanteras olika i olika märkspråk.</p>  
</body></html>
```

XML: Måste ALLTID vara välformad.

```
<HTML><body>  
<p>Ett enkelt exempel med två stycken.</p>  
<p>Hanteras olika i olika märkspråk.</p>  
</body></HTML>
```

XHTML: (Måste ALLTID vara välformad. Alla taggar i lower-case – och mycket mera!)

```
<html><body>  
<p>Ett enkelt exempel med två stycken.</p>  
<p>Hanteras olika i olika märkspråk.</p>  
</body></html>
```

XML – Olika sätt att beskriva strukturen

- **Well formed** – ”inte alls” utan XML-instansen (dokumentet) ”är som det är” och har en viss struktur som ett program kan förstå utan ytterligare information (eftersom alla taggar är kompletta).
- **DTD** - Document Type Definition – Metod som härstammar från SGML för att beskriva strukturen hos ett XML-dokument (även SGML, HTML) som gör det möjligt att validera strukturen. Dock begränsad funktionalitet vad gäller att styra ”fältinnehåll”. Många avancerade finesser som sällan används.
- **XML Schema** – ”Efterföljare” till DTD, kraftfullare ur många synvinklar men även vissa svagheter. Tillför bl.a. möjlighet att definiera syntax för ”fältinnehåll”.
- XML Schema beskrivs i XML (XSD). Jfr DTD som har ett ”eget” språk.
- Både DTD och XML Schema är möjliga att utvidga (om man förberett för detta).

Även om XML Schema är kraftfullt är risken ganska stor att ”människa skrivregler” INTE går att uttrycka fullt ut.

XHTML - eXtensible HyperText Markup Language

- HTML + XML => **XHTML** (men X betyder INTE XML!)
- Ansats att styra upp HTML genom att påföra XMLs formkrav
- Måste vara såväl ”välformad” som följa DTD
 - Gör det enklare att bygga applikationer – bra för t.ex. mobiler
- **XHTML 1.0 Transitional** – Nästan HTML, följer XMLs (+HTMLs) regelverk
- **XHTML 1.0 Strict** – Som ovan med UTAN layout-element
 - Layout görs istället med CSS, dvs skiljer struktur och layout!
- **XHTML 1.1** – Utgår från XHTML 1.0 Strict men delar upp taggarna i ”moduler”
 - Möjliggör utvidgning (eXtensible) med mer eller mindre ”allmänna” moduler
- **XHTML 2.0** – Snarlik HTML på många sätt men i defacto en helt ny standard.
 - Ej bakåtkompatibel med HTML, därmed har man sluppit en del ”fotbojor”.
 - ”Working Draft” sedan juli 2006.

XML / XHTML – Relaterade standarder och begrepp

- **CSS** - Cascading Style Sheets – Språk för att styra presentation av webbsidor
 - T.ex. `Bo Ek` blir *Bo Ek* (i.st.f. `<i>Bo Ek</i>`)
 - “Cascading” syftar på möjligheten att påverka utseendet i flera led, t.ex. både av författaren och läsaren (större text, skilja rött från grönt etc.)
- **XSLT** - Extensible Stylesheet Language Transformations – Språk för att översätta (transformera) ett XML-format till ett annat. (Ganska lätt att göra i praktiken bara det två formaten är ”kompatibla”. Ofta är så dock INTE fallet!)
- **XML Namespace** – Metod att unikt definiera namn på taggar (undvika krockar)
 - Bl.a. en förutsättning för att XHTML skall vara eXtensible.

```
<html xmlns="http://www.w3.org/1999/xhtml">                                <!-- Def av default namespace -->
<body xmlns:si="http://bimp.org/si-units">    <-- Def av alt namespace med alias "si" *) -->
  <p>Turbomotors laddtryck får maximalt vara <si:p unit="kPa">4711</si:p> kilo pascal.<p>
</body>
</html>    <!-- *) "si" är bara ett högst lokal alias, kunde lika väl varit "x", "units", etc. -->
```

Identifierare för XML (och RDF som är nästa ämne att ta upp)

Några begrepp som lätt förvirrar:

- **URI** - Uniform Resource Identifier (URL, URN eller bådadera)
 - URL - Uniform Resource Locator (t.ex. <http://www.sunstone.se/>)
 - URN- Uniform Resource NAME (t.ex. urn:isbn:0-395-36341-1)
- OBS: Ofta förvirrande med URI som är URN men ser ut som URL, t.ex. namn på XML Namespace eller RDF resource. (Webbsidan <http://www.w3.org/1999/xhtml/> finns t.ex., men förklarar bara att URIn INTE är tänkt som en URL utan som en URN...)
- Man kan fråga sig varför man gjort på detta sätt, och en av förklaringarna är att det är ett enkelt sätt att skapa garanterat unika namn. (T.ex. <http://x.sunstone.se/y>)
- Jfr programmeringsspråket java som använder en snarlik princip som dock är mindre förvirrande (t.ex. `se.sunstone.x.y.SampleClass`).

Bakgrund – Semantiska webben och Metadata

The **semantic web** is an **evolving** extension of the World Wide Web in which web content can be expressed not only in natural language, but also in a form that can be understood, interpreted and used **by software agents**, thus permitting them to **find, share and integrate information** more easily. (http://en.wikipedia.org/wiki/Semantic_Web)

Metadata är ”data om data”. (Jfr böcker i bibliotekets hyllor och metadata i kortlådor.)

T.ex. ”Anders Sundström har författat denna presentation” som kan uttryckas i HTML med hjälp av Dublin Core (DC) i html:

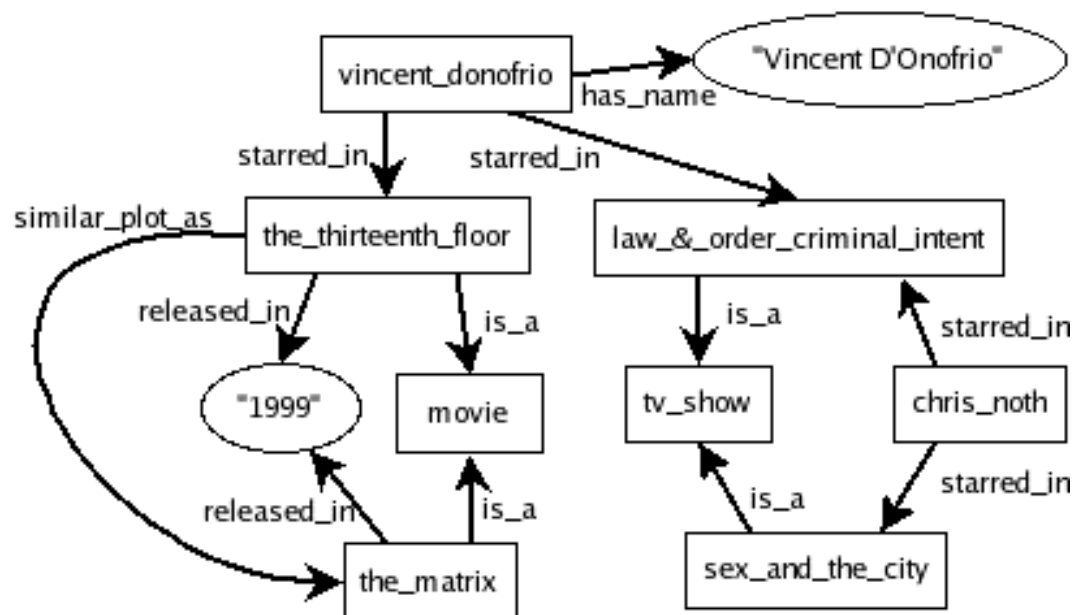
```
<html><head>
  <meta name="DC.creator" content="Anders Sundström" />
  <meta name="DC.date" scheme="DCTERMS.W3CDTF" content="2007-04-24" />
  <link rel="DC.relation" href="http://www.rattsinfo.se/" />
</head><body>...</body></html>
```

RDF - Resource Description Framework

En modell av ett “ramverk” för att beskriva resurser (bl.a. dokument).

Innan vi går in på hur RDF ”ser ut”, lite om vad RDF ”är” (vilket man lätt förbiser).

- Bygger på påståenden om en resurs (statements)
- På formen **subjekt** – **predikat** – **objekt**
- Och som flätas ihop till ett stort nät – den semantiska webben



(<http://www.xml.com/pub/a/2001/01/24/rdf.html>)

RDF - Resource Description Framework (forts)

Exempel:

- Subjekt http://en.wikipedia.org/wiki/Resource_Description_Framework
- Predikat **Publisher**
- Objekt **Wikipedia**

Kanske lättast att (som text) läsa på s.k. triplett-form:

<http://en.wikipedia.org/wiki/Resource_Description_Framework> -> **Publisher** -> **Wikipedia**

Eller mer precist...

<http://en.wikipedia.org/wiki/Resource_Description_Framework> -> <<http://purl.org/dc/elements/1.1/publisher>> -> <<http://wikipedia.org/>>

...men också allt mindre läsbart för det mänskliga ögat (vilket inte heller är meningen)

RDF – Olika sätt att uttrycka RDF

Antingen integrerat i resursen eller separat, spelar ingen roll rent principiellt.

RDF/XML:

```
<rdf:Description rdf:about="http://en.wikipedia.org/wiki/Resource_Description_Framework">
  <dc:publisher>Wikipedia</dc:publisher>
  <dc:language><dcterms:RFC1766><rdf:value>en</rdf:value></dcterms:RFC1766> </dc:language>
</rdf:Description>          <!--Där rdf, dc, dcterms är alias för XML Namespace(s) vars definitioner utelämnats -->
```

XHTML 2.0 med RDFa:

```
<p xmlns:dc="http://purl.org/dc/elements/1.1/"
  about="http://www.example.com/books/wikinomics">
  In his latest book <em property="dc:title">Wikinomics</em>,
  <span property="dc:author">Don Tapscott</span> explains deep changes in
  technology, demographics and business. The book is due to be published in
  <span property="dc:date" content="2006-10-01">October 2006</span>.
</p>          (http://en.wikipedia.org/wiki/RDFa)
```

RDF & Metadata – Relaterade standarder / begrepp

Efter denna genomgång av det grundläggande vad gäller metadata och RDF måste man också konstatera att det behövs något ”mer”:

- Ytterligare ordning och reda vad gäller de begrepp man använder
- Något som knyter ihop all information och ”ger den liv”

Hanteras bl.a. av...

- SPARQL (“sparkel”) - SPARQL Protocol and RDF Query Language
- SKOS - Simple Knowledge Organisation System
- OWL - Web Ontology Language

... vilket vi kommer att få höra mer om i nästa inlägg.