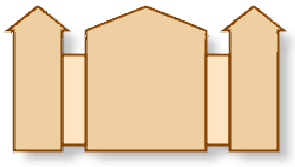
Structured Documents on the Web

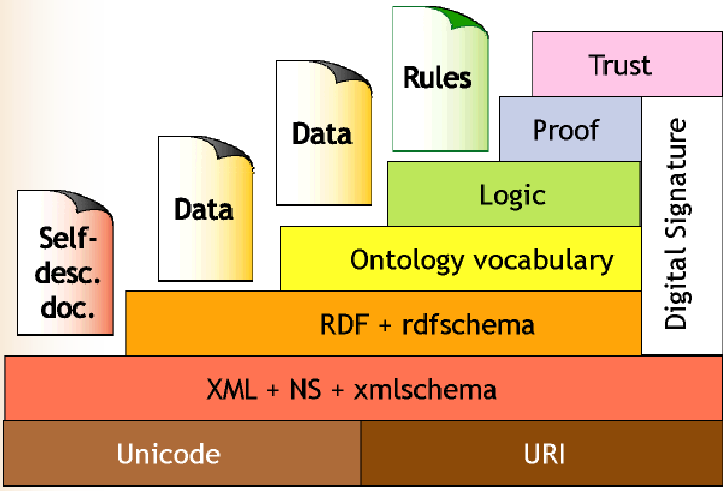
Jacco van Ossenbruggen
CWI Amsterdam



Talk overview

- Introduction and historical background
- Multiple delivery publishing (MDP)
- MDP on the Web: Style sheets
- Conclusion

TBL Layer Cake

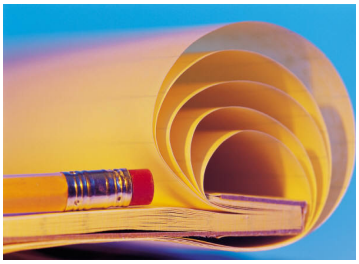


What is a "Document"?

Examples:

- –Book, poem
- –Article, paper, report
- –Memo, e-mail, letter, etc

My definition:



A document is a self-contained unit of information, intended to be communicated to a human interpreter

What isn't a document?

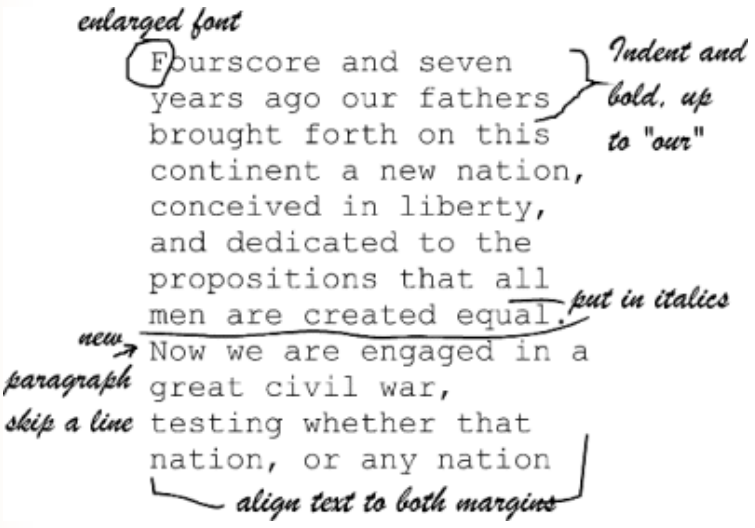
All data that is:

- Fragmentary
- Intended solely for further machine processing

Examples:

- Database records
- HTTP requests
- Software source code
- RDF metadata …

What is Markup?



Picture taken from "The XML Handbook" by Goldfarb and Prescod

Talk overview

- Introduction and historical background
- Multiple delivery publishing (MDP)
- MDP on the Web: Style sheets
- Conclusion

Electronic Documents (then)

- Goal (authoring/production):
  - More efficient/effective production by using WYSIWYG authoring interfaces (WP,DTP)
- Goal (final-form):
  - Obtain same typographic quality as traditional print
- Production electronic, dissemination and final-form still on paper
- Authoring & storage format:
  - Mimics final-form presentation format

Electronic Documents (now)

- Goal (authoring/production):
  - Efficient, industrial scale, full document life cycle
- Goal (final-form):
  - Improve communication by exploiting presentation potential of new media
    - Use of audio, video, animation, etc
    - Interactivity (hyperlinks, forms, etc.)
    - Dissemination over internet (WWW)
    - Use of document technology to access (legacy) information
- Both production & dissemination is electronic
- Authoring & storage format:
  - Differs radically from presentation format

Electronic Documents: Problems

- **Longevity** (many documents need to last >30 years)
- **Maintenance** & **reuse**
- **Flexibility** & **tailorability**

In general:

- Doc. formats can't cope with **changing environments**:
  - Hardware dependencies (use of printer/typesetter- specific control sequences)
  - Software dependencies (use of proprietary formats)
  - Presentation dependencies (layout and style)
- C.f. issues in software engineering

"Solution"

(Semi-automatically) convert all documents to new format or new layout

- –Expensive
- –Time consuming
- –Error prone (& pretty boring too!)
- –Loss of (implicit) information

Real solution

Multiple delivery publishing model

A.k.a.

- Cross-channel publishing
- Separation of content & presentation
- Separation of style & structure

Multiple delivery publishing (MDP)

- MDP distinguishes two formats
  - One for authoring and long term storage
  - Another one for final-form presentation
- Mappings from source to target format
- Source format can now abstract from all details that are likely to change in the target

Sounds pretty straightforward eh?

But it actually meant...

Revolution!

Software developers

No longer control their application's own file format

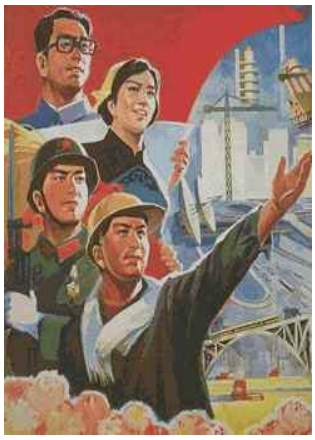
Document authors

No longer control style and layout of their documents

Tools

No longer used the "sacred" WYSIWYG paradigm

Multiple delivery publishing was not obvious at all!



MDP: Nothing new …

- This approach was already advocated by Goldfarb et al. in the 70's!
- Source documents encoded using IBM's Generic Markup Language (GML)
- GML was standardized by ISO in 1986 as SGML
- First publicly available parser developed here at the VU
  - Amsterdam SGML Parser by Warmer, Van Egmond and Van Vliet (late 80's)

MDP & SGML

- MDP and SGML remained highly controversial
  - People do not like to give up control or change the way they work
  - MDP could not always match the output quality of traditional tools
  - MDP is no silver bullet!
  - Primarily suited for *content-driven* applications
  - Not for *layout-driven* applications
- SGML standard is extremely complex
  - Still not fully implemented (?!)
  - Huge and inflexible
  - Mainly used in academic and large organizations

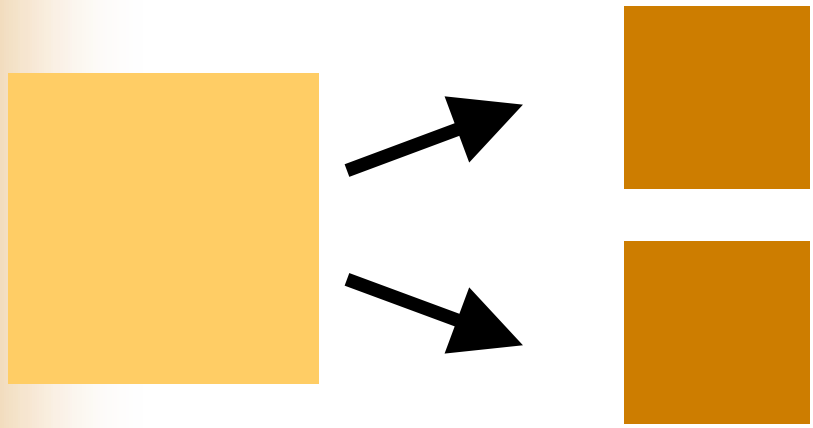
"SGML" revival due to the Web

- HTML already is an application of SGML *(eh... sort of)*
- XML is a stream-lined and simplified subset of SGML *(it really is, this time)*
- Published in 1998, XML already had more applications that year than SGML ever had!

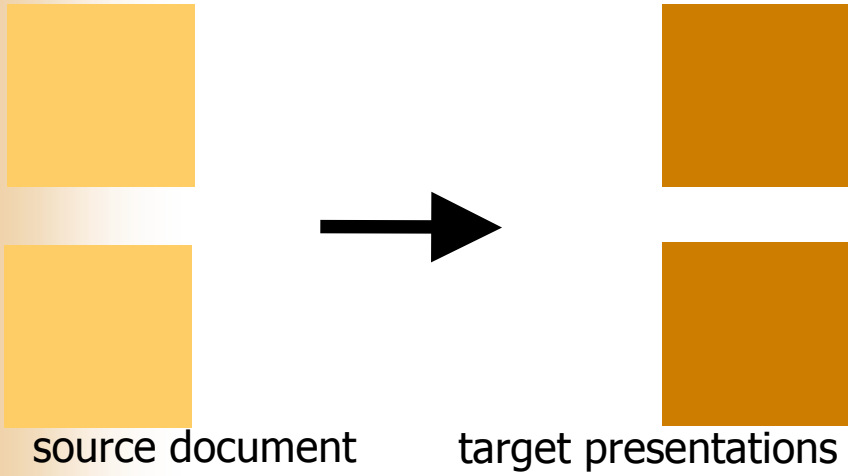
Talk overview

- Introduction and historical background
- Multiple delivery publishing (MDP)
- MDP on the Web: Style sheets
- Conclusion

MDP: easy reuse of source document



MDP: easy reuse of style specification



MDP: Document design dimensions:

- Content versus markup
  - what is in the tags, what is between the tags?
- Embedded versus external markup
  - What is encoded in the same file, what is stored elsewhere?
- Declarative versus procedural
  - Specify what or specify how
- Domain independent versus domain specific
  - <title> or <product-shelf-number>?
- Layout-driven versus content-driven applications
  - magazine cover or technical manual?
- Visual markup versus structured markup
  - <i> or <emph>?

Source vs. presentation format

- Source format:
  - Structured, declarative markup
  - Can be domain independent but...
  - ...is usually tailored to a specific domain
  - Provide sufficiently rich structure for style sheets and other processing
- Presentation format:
  - Visual, often procedural markup
  - Can be platform/medium independent but...
  - ... is usually tailored to a specific output medium/device
  - Provide sufficient information to obtain high quality output
- How do you classify your favourite document format?

Domain independent vs. domain specific

Domain independent:

- –Examples: HTML, Docbook, (LaTeX)
- –Wide deployment: easy to learn, many (cots) tools available
- –Poor semantics for automatic processing other than presentation
- –Tools only need to deal with predefined markup semantics

Domain specific:

- –Examples: product specific documents standards (e.g. automobile and aircraft industry)
- –Users need training, tailor-made tools might need to be developed
- –Rich (domain-specific) semantics for further processing (retrieval, screen scraping etc.)
- –Need tools tailored to domain-specific document formats or ...

Presentation of domain-specific document formats

- *Generic* tools that can process *user*-defined markup
  - –Software adapts to document structure
- No predefined (presentation) semantics
  - –Also need to be user-defined

Beyond presentation semantics

- Document-oriented semantics
  - –static: style and layout
    (e.g. style sheets, focus second half of this talk)
  - –interaction: linking & forms
  - –dynamic: scheduling & animation
- Other semantics:
  - –do not describe the document,
    but the *domain* of the document's *content*
  - –can still be related to document
    - annotations & meta data
  - –RDF(S), OWL, DAML+OIL, etc.

Talk overview

- Introduction and historical background
- Multiple delivery publishing (MDP)
- MDP on the Web: Style sheets
- Conclusion

Multiple delivery publishing on the Web

Function \ Bloodtype	W3C/HTML		
Markup	HTML		
Style	CSS		
Linking	<a href=		
Addressing	<a name=		

Multiple delivery publishing on the Web

Function \ Bloodtype	W3C/HTML		ISO/SGML
Markup	HTML		SGML
Style	CSS		DSSSL
Linking	<a href=		HyTime, TEI
Addressing	<a name=		HyTime, TEI

Multiple delivery publishing on the Web

Function \ Bloodtype	W3C/HTML	W3C/XML	ISO/SGML
Markup	HTML	XML	SGML
Style	CSS	CSS, XSLT, XSL FO	DSSSL
Linking	<a href=	XLink	HyTime, TEI
Addressing	<a name=	XPath, XPointer	HyTime, TEI

Style sheets: HTML & CSS

HTML with embedded visual markup:

```
<h3 align="center">
  <font color="black">
     The Need for Style Sheets
  </font>
</h3>
```

versus HTML with separate CSS style sheet:

HTML:

```
<h3>The Need for Style Sheets</h3>
```

CSS (optional!):

```
h3 { text-align: center; color: black }
```

Style sheets: XML & CSS

- Example fragment using MyOwnML (XML):

```
<product>
  <type>X112332</type>
  <color>dark blue</color>
  ...
</product>
```

- With XML, your style sheet needs to specify more than just the style (CSS2):

```
product { display: list-item; ...}
type    { display: none; ...}
color   { display: block; ...}
```


Style sheets: XML & CSS

- With XML, style sheets are no longer optional
- Information presented with CSS remains in the same order
- Source tree and target tree have similar structure (allows cascading)
- Style properties are inherited via the source tree (!)

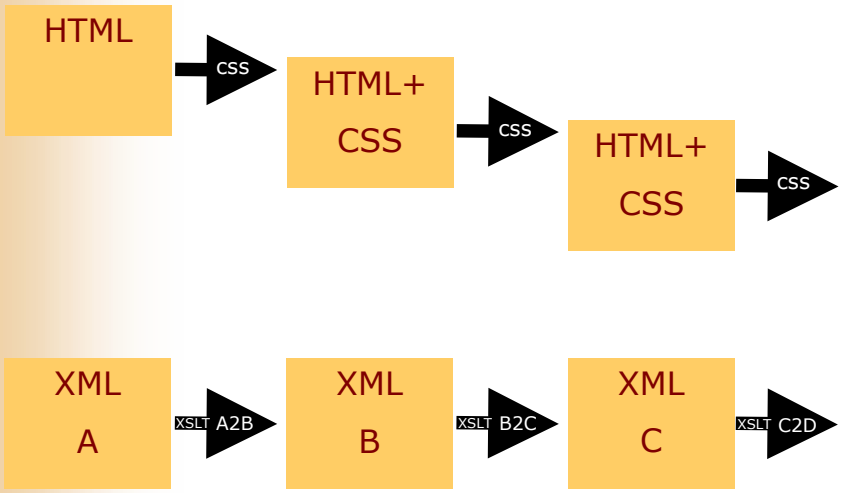
Transformations: XML and XSLT

- What if the desired target tree differs radically from the source tree?
  - assigning CSS properties will not suffice
  - need a language to describe XML (tree) transformations:
- XSL Transformations (XSLT)
  - more on XSL later!
  - XSLT transforms from XML to:
    - XML (including XHTML)
    - HTML (for legacy browsers, outputs "old" SGML syntax)
    - plain text (can be used to generate other text formats such as RTF, BibTeX, …)

Transformations: CSS vs XSLT

- XSLT itself also uses XML syntax (unlike CSS …)
  - so you can transform XSLT using XSLT…
  - … but it doesn't look really human friendly!
- The structure of the target tree and source tree can differ (unlike CSS):
  - XSLT style sheets can be chained, not cascaded

Cascading versus Chaining



XSLT template rules

- Transformations are described as a set of one or more template rules
- Each template rule consists of two parts:
  - A pattern that is matched against the source tree: the selector
  - A template to be filled in and added to the result tree
- XSLT selectors are based on XPath, e.g:

`product`	`/product`
`color|type`	`product/color`
`catalog//product`	`text()`
`id("W11")`	`product[1]`
`@class`	`/ * @*`

XSLT: Example (I)

A single template rule may be sufficient...

```
<xsl:template match="/">
  <html>
   <head>
    <title>Product Report Summary</title>
    </head><body>
    <p>...<table>
     <tr><td>
      <xsl:value-of select="product/type"/>
     </td><td>
      <xsl:value-of select="product/color"/>
     <td> ... </tr></table>...</body></html>
</xsl:template>
```


XSLT: Example (II)

... or a style sheet can contain many (smaller) template rules

```
<xsl:template match="/">
 <table>
  <xsl:apply-templates/>
 </table>
</xsl:template>

<xsl:template match="product">
 <tr>
  <xsl:apply-templates/>
 </tr>
</xsl:template>

<xsl:template match="color|type">
 <td>
  <xsl:apply-templates/>
 </td>
</xsl:template>
```

What about styling RDF?

1. Use XSLT on the RDF/XML serialization
   - Pro: Single step, no extra tools needed
   - Con: Different serializations of the same graph
   - Con: no RDF, RDFS or OWL semantics
2. Use XSLT on the XML serialization of the (sesame) query results
   - Pro: consistent source syntax
   - Pro: use built-in semantics of query engine
   - Con: two step process, hard to interact

What about styling RDF?

3. Extend XSLT with an RDF query language
   - Pro: Single step process
   - Pro: mix information form XML and RDF sources
   - Con: requires extension
   - Con: ugly syntactic mix of XML, XSLT and query language

What about styling RDF?

4. Fresnel: Design special purpose vocabulary in RDF
   - Pro:
     - RDF-only solution
     - Natural use of RDF syntax and semantics
     - Pure styling delegated to (embedded) CSS
   - Con:
     - Yet another language, yet another tool
     - Not widely supported (yet)

MDP wrap up: pros & cons

- Advantages:
  - Longevity
  - Reusability
  - Flexibility & tailorability
- Disadvantages:
  - Complexity
  - High dependency on tools (?!)
  - Training, high initial investment
- Works best for content-driven material
  - becomes cheaper due to massive use on the Web
  - free tool support
    - XML parsers/browsers, XSLT engines, XSL FO formatters, etc.
  - many "off-the-shelf" source & target formats to choose from
    - XHTML, SVG, SMIL, MathML, Docbook, PDF, ...
- Integration of style sheet & Semantic Web languages
  - still at its infancy

Links

- Fresnel:
  - http://www.w3.org/2005/04/fresnel-info/
- SeRQL-extended XSLT:
  - http://homepages.cwi.nl/~media/cuypers/java/
- Overview pages at www.w3.org:
  - http://www.w3.org/XML/
  - http://www.w3.org/Style/XSL/
  - http://www.w3.org/Style/CSS/
- Recommendations (+ drafts) at www.w3.org/TR/:
  - http://www.w3.org/TR/xslt
  - http://www.w3.org/TR/REC-xml
  - http://www.w3.org/TR/REC-CSS2
- Tutorials and more
  - http://www.xml.com
  - http://www.mulberrytech.com/
  - http://www.mulberrytech.com/quickref/
    (personal favourite)