

# **Publishing XML with XSL A Survey of Tools and Techniques**

May 13, 2003



## ***Today's Agenda***

1. Introduction
2. History of XSL
3. The Business Case for XSL
4. Transformation and Presentation
5. Sample Architectures
6. Survey of Current Toolsets
7. Sample Applications
8. Trends
9. Conclusion



# ***1. Introduction***

## **Publishing XML involves:**

- Assembly of information for dissemination
  - Applying Formatting and Typesetting properties
- ⇒ Ultimately gives a publication its "look and feel".
- ⇒ Key factors in the effectiveness of written communication.
- ⇒ Consider these aspects of publishing as they relate to your own organization.



# ***1. Introduction***

## **Challenges:**

- It has always been difficult to produce high quality print output from XML and SGML
- Historically, has required specialized typesetting tools, (e.g. Advent 3B2, Datalogics Composer, Arbortext Publisher, Xyvision XPP, etc.)
- These tools have a steep learning curve due to often cryptic, proprietary stylesheet languages or typesetting codes.



# ***1. Introduction***

## **The Extensible Stylesheet Language (XSL):**

- A standard language for transforming and specifying formatting for XML documents
- Addresses the need to produce quality printed output from XML data without the need for proprietary tools.
- An ideal tool for batch-pagination applications and the “print on demand” sector of the publishing industry.



## ***2. History of XSL***

Starts with FOSI and DSSSL in the early nineties

- FOSI – MIL-PRF 28001C “Formatting Output Specification Instance”
- DSSSL – ISO 10179:1996 “Document Style Semantics and Specification Language”



## ***2. History of XSL***

Problems with FOSI and DSSSL:

- Syntax hard to learn and understand
- Difficult to implement the entire specifications due to their size, complexity
- As a result, not widely implemented

*(Technically, they are very strong standards.)*



## ***2. History of XSL***

With the arrival of XML:

- A styling language was one of the original three XML standards - XML, XSL, XLINK
- Work began early on XSL
- Progress was slow due to the focus on XML as a data interchange format (e.g. EDI, database integration, etc.), not as a data publishing format



## ***2. History of XSL***

- The focus of W3C standards development became "transformation", due to its application to data interchange issues.
- XSL was split into two parts, "XSL Transformations" and "XSL Formatting Objects"
- XSLT was completed March 1999 ([www.w3c.org/TR/xslt/](http://www.w3c.org/TR/xslt/))
- XSL-FO was completed October, 2001 ([www.w3c.org/TR/xsl/](http://www.w3c.org/TR/xsl/))



## ***2. History of XSL***

### **Status of XSLT:**

- Many implementations in many programming languages
- Becoming a standard feature of many enterprise software applications and frameworks.

### **Status of XSL-FO:**

- At least 2 commercial implementations and 2 free implementations available (not quite fully compliant).



## ***2. History of XSL***

### **Expectation:**

XSL-FO will gain widespread adoption for a number of reasons, mainly:

- It provides the necessary functionality
- It has an easier syntax
- It has better marketing



## **3. The Business Case for XSL**

### **What you want:**

- High quality, consistent output ⇒ “look and feel”, “branding”
- Automated production of information products ⇒ “print on demand”
- Supports re-purposing of content ⇒ “single-source publishing”
- Supports the investment in content rather than presentation ⇒ “separation of content from formatting”
- Independent of any particular application ⇒ “portability”
- Availability of resources ⇒ “non-proprietary skills, languages”



### **3. The Business Case for XSL**

#### **Some options:**

- ? High-end solutions (Datalogics Composer, 3B2, Miles33, Xyvision, Life\*Type)
  
- ? Use or convert to proprietary solutions (XML->RTF, XML->MIF, XML->TeX, etc.)
  
- ✓ Open standards (DSSSL, XSL)



## ***3. The Business Case for XSL***

### **Impact to the Organization:**

- ⇒ Increase quality of products, customer satisfaction
- ⇒ Reduce cost of production
- ⇒ Support the creation of new revenue streams by re-purposing/re-using existing assets
- ⇒ Reduce cost of system development and maintenance



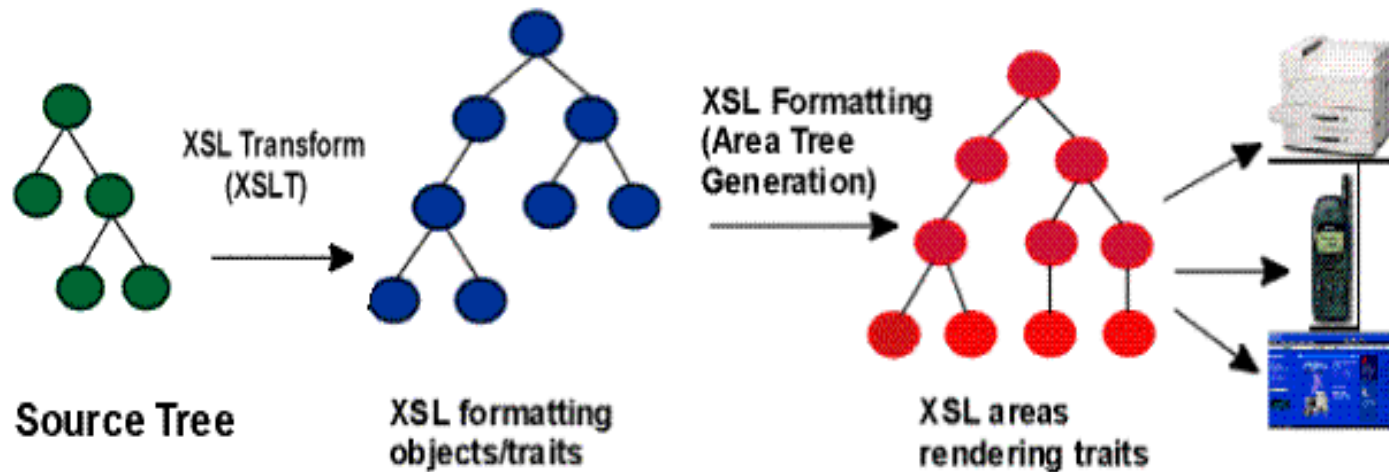
## ***3. The Business Case for XSL***

### **Impact to the Technical Writer/Information Architect:**

- ⇒ Can focus on creating quality content
- ⇒ Style Guides are automatically applied
- ⇒ The design of information products can be re-used
- ⇒ Opportunity to apply skill sets and knowledge in a new way

## 4. Transformation and Presentation

**Publishing = Transformation + Presentation**



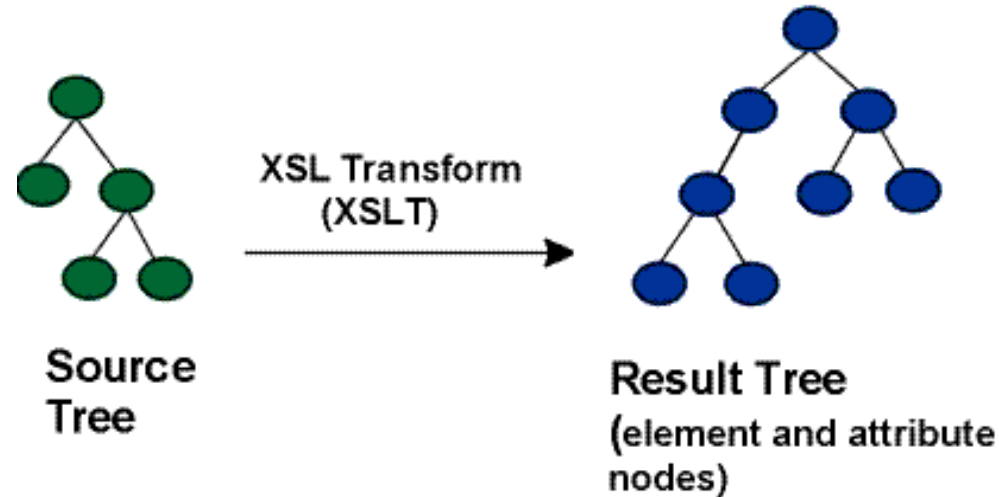
Source: Extensible Stylesheet Language Version 1.0, <http://www.w3c.org/TR/xsl/>



## **4. Transformation** *and Presentation*

- XSLT provides a standard method for specifying tree transformations.
- It includes many facilities for working with XML data, including
  - node selection
  - id generation and resolution (related to linking)
  - access to ancestral information (parents, grandparents, etc.)
  - Information about child elements

## 4. Transformation and Presentation



Source: Extensible Stylesheet Language Version 1.0, <http://www.w3c.org/TR/xsl/>

- With XSLT, you write transformation "templates" which an XSLT processor applies to an XML instance to generate a resulting data set.

## 4. Transformation and Presentation

XSLT can be used in two ways:

### 1. Event driven processing

- The stylesheet contains templates (rules) to match element occurrences in context, for example:

```
...  
<xsl:template match="chapter">  
    <xsl:apply-templates/>  
</xsl:template>  
  
<xsl:template match="title">  
    <fo:block space-before="1cm">  
        <xsl:apply-templates/>  
    </fo:block>  
</xsl:template>  
...
```



## 4. **Transformation** and Presentation

### 2. A “direct access” or “pull” approach

- using XPATH expressions to directly select elements and attributes from an XML instance
- Same approach as in server pages

```
...  
<xsl:for-each select="//title">  
  <fo:block space-before="1cm">  
    <xsl:value-of select="."/>  
  </fo:block>  
</xsl:for-each>  
...
```



## **4. Transformation** *and Presentation*

### **Advantages of XSLT:**

- Applies processing and formatting consistently and automatically
- Formatting rules are stored separately from the data, in a non-proprietary, human-readable format
- Can be less complicated than programming in Java, Perl, etc.
- Stylesheets can be used with different XSLT processors
- Easily integrates with other XML standards compliant applications



## **4. Transformation** *and Presentation*

### **Disadvantages of XSLT:**

- Performance - can run slower than a well designed and coded application
- XSLT is not a full programming language, so it isn't a general purpose tool
- Data driven, so sensitive to “garbage in, garbage out”



## **4. Transformation** *and Presentation*

### **Bottom Line:**

- If you're manipulating XML, then XSLT is a good tool to use.

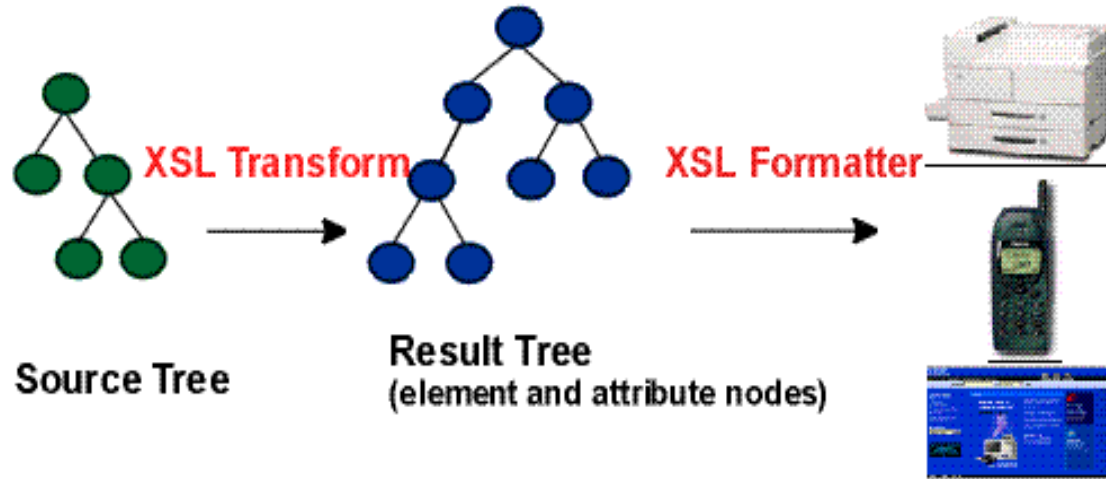


## **4. Transformation and *Presentation***

### **Extensible Stylesheet Language -Formatting Objects**

- The standard syntax for specifying the presentation of XML information.
- XSL includes XSLT by reference.
- Builds on concepts from DSSSL and Cascading Stylesheets v2.0 (CSS2)
- Strong support for internationalization, e.g. different “writing modes”, Unicode

## 4. Transformation and **Presentation**



Source: Extensible Stylesheet Language Version 1.0, <http://www.w3c.org/TR/xsl/>

- XSLT is used to transform the input document into an XSL-FO instance
- An XSL-FO engine interprets the FO instance and translates it into printed pages (PostScript, PDF, RTF, etc.)



## 4. Transformation and **Presentation**

### **Extensible Stylesheet Language -Formatting Objects**

- The stylesheet must specify:
  - page areas and sets of pages to be used to compose a document for paper.
  - "Flows", areas on pages which are "filled" with text and graphics
  - "block" areas and "inline" areas which are used to contain the content taken from an input XML document.
- The blocks are placed in the flows on the pages



## 4. Transformation and **Presentation**

### **Extensible Stylesheet Language -Formatting Objects**

Stylesheets may also:

- Generate additional portions of the final output not present in the input (such as ToCs, lists of figures, etc.)
- Suppress certain portions of information from the output (such as editorial content, non-applicable material, etc.).
- Re-order or sort content intended for output



## **4. Transformation and Presentation**

### **XSL-FO supports:**

- portrait and landscape pages
- multiple page sizes
- multiple writing directions
- headers/footers
- generated page numbering
- graphics
- floating areas
- Hyphenation and justification
- recto-verso pages
- multiple columns of text on a page
- Unicode
- generated ToCs
- re-sorting items for output
- tables, including running heads, row and column spans
- widow/orphan control



## 4. Transformation and **Presentation**

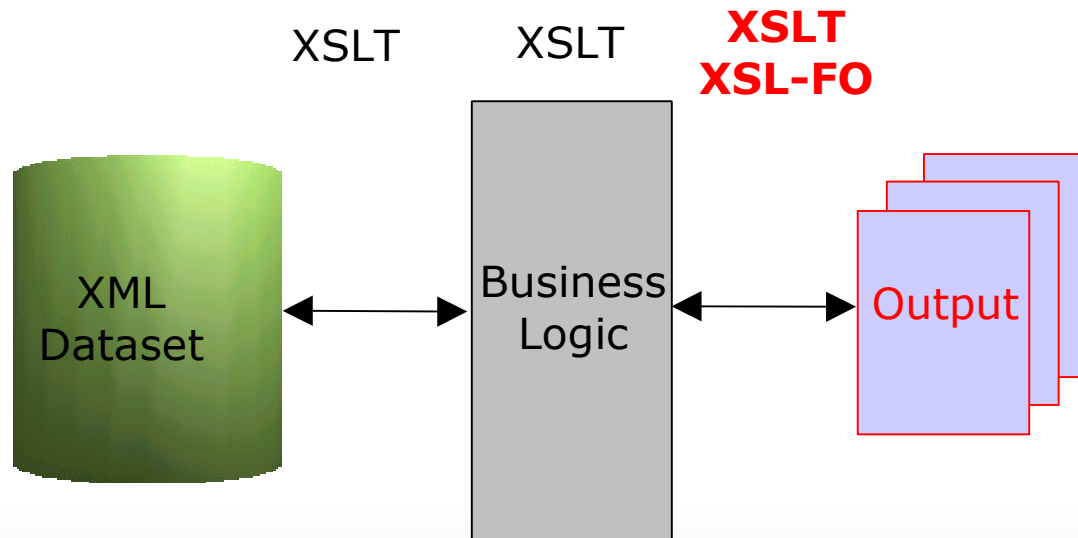
### **XSL-FO does not support (yet?):**

- loose leaf publishing
- Generation of revision markings by comparing previous and current versions of documents
- no "feedback" between the XSL-FO formatter and the stylesheet, limits handling of special formatting cases
- limited support for back of book indexes

Note: Some formatters may provide "extensions" to address these and other issues.

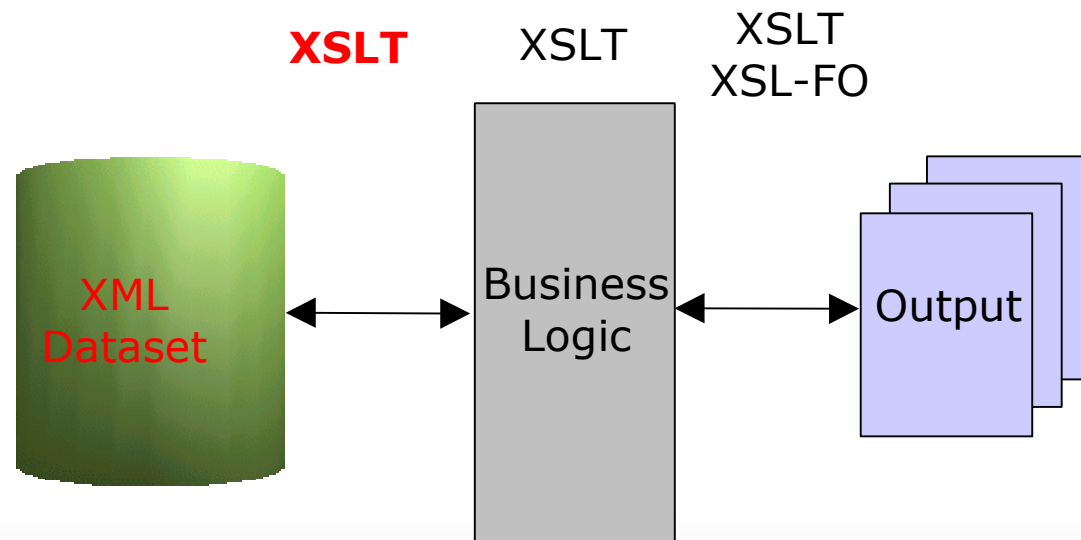
## 5. Sample Architectures

### Rendering Engines



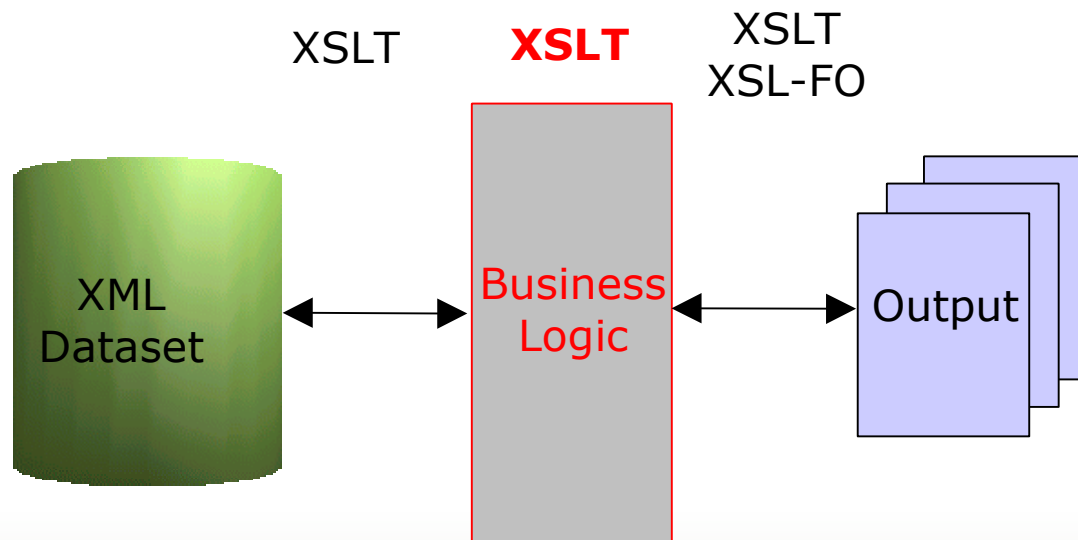
## 5. Sample Architectures

### Database Publishing



## 5. Sample Architectures

### “Middleware”





## ***6. Survey of Current Toolsets***

### **XSL Formatter, Antenna House** *([www.AntennaHouse.com](http://www.AntennaHouse.com))*

- A commercial product (v2.5)
- MS Windows/Unix (Solaris and Linux)
- Implements a large portion of current XSL-FO spec
- Graphical interface in addition to command line interface
- Can also be used in a server environment
- Produces PDF from XML input



## 6. Survey of Current Toolsets

### **XEP**, *RenderX* ([www.RenderX.com](http://www.RenderX.com))

- A commercial product (v.3.4)
- Implemented in Java
- Implements a large portion of current XSL-FO spec
- No graphical interface, suitable for integrating with other systems
- Produces PDF, PostScript output from XML input



## ***6. Survey of Current Toolsets***

**FOP**, Apache Foundation (<http://xml.apache.org/fop>)

- Free, open source application (v. 0.2.5)
- Implemented in Java
- Incomplete implementation of many portions of the current XSL-FO spec,
- No graphical interface, suitable for integrating with other systems
- Produces PDF, PostScript, RTF, etc.



## 6. Survey of Current Toolsets

### **PassiveTeX**, TEI ([www.tei-c.org.uk/Software/passivetex/](http://www.tei-c.org.uk/Software/passivetex/))

- Free, open source application
- An add-on for use with TeX typesetting software package
- Implemented as set of TeX macros
- Limited implementation of current XSL-FO spec
- Can be integrated with other systems via Unix "tool chain"
- Produces PDF, PS, dvi from XML input



## ***6. Survey of Current Toolsets***

### **Epic Editor v4.2** *Arbortext Inc. ([www.Arbortext.com](http://www.Arbortext.com))*

- A commercial application
- C++ implementation
- Integrated with Arbortext's XML content management system, E3
- Implements a large portion of current XSL-FO spec (Arbortext was an active participant in developing the XSL standard)



## ***7. Sample Applications***

- XSL Formatter
- FOP
- HTML Rendering of SGML
- Newbook Publisher



## **8. Trends**

### **Toolkits**

- Need a number of software components to do XSL development: e.g. XML parser, XSLT processor, XSL-FO engine, a Java Runtime Engine, possibly other programming/scripting languages (e.g. Python)
- Toolkits are available to reduce the complexity of getting all this set up and working correctly.

Examples:

Oracle's XDK, NIST XSL toolkit, others



## **8. Trends**

### **Industry Standard Stylesheets - Docbook**

- Docbook - an open standard, sponsored by O'Reilly publishers, supported by a number of dedicated users and developers
- Norm Walsh ([www.nwalsh.com](http://www.nwalsh.com)) has developed XSLT stylesheets that renders Docbook XML to HTML and print output. Freely available, maintained as an open source project.
- Highlights - CALS table implementation, index generation, ToC generation, standard page specifications
- An excellent source of re-usable templates and stylesheet components



## **8. Trends**

### **Integration with other applications**

#### Examples

- Oracle's XDK
- Adobe Document Server
- Cocoon (<http://xml.apache.org/cocoon/>)



## **8. Trends**

### **GUI Advances (still to come)**

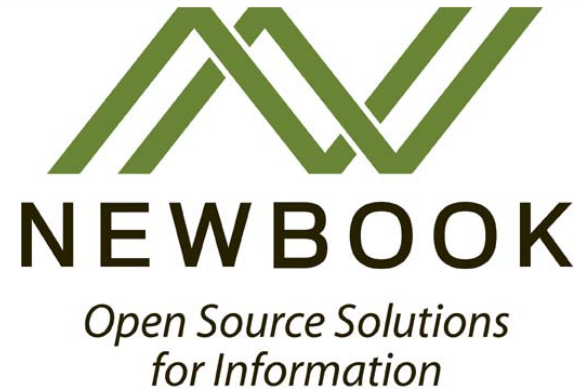
- Visual Stylesheet Development tools
- Browser support for XSL-FO (client side formatting)



## **9. Conclusion**

### **XSL**

- Solves the business problem of high quality print output from XML
- Addresses both transformation and presentation
- Can be used effectively in different areas of a publishing system
- Has growing vendor support
- Is an open standard



**Thank You!**

**Newbook Production Inc.  
1515 Matheson Blvd. E. Suite 200  
Mississauga, ON Canada L4W 2P5  
800.588.9334  
905.602.9136**

**[www.newbook.com](http://www.newbook.com)  
[info@newbook.com](mailto:info@newbook.com)  
[dmillar@newbook.com](mailto:dmillar@newbook.com)**