




6 listopada 2003



XSL

Extensible Stylesheet Language



XPath – XML Path Language

Problem:

- jednoznaczne adresowanie fragmentów struktury dokumentu XML.


Rozwiązanie:

- drzewiasty model struktury dokumentu,
- normalizacja zawartości dokumentu (ten sam byt, choć różnie zakodowany, jest tak samo reprezentowany w modelu),
- język ścieżek w drzewie struktury.

Status: Rekomendacja W3C z 16 listopada 1999 r.

Zastosowania:

- XSL,
- XPointer,
- ...




XPath data model

Modelowanie dokumentu przy pomocy drzewa:

- węzeł root,
- węzły elementów,
- węzły atrybutów,
- węzły tekstowe,
- węzły instrukcji przetwarzania,
- węzły komentarzy,
- węzły przestrzeni nazw.

Własności węzłów:

- string-value,
- normalizacja odwołań do encji i sekcji CDATA,
- expanded-name.



Wyrażenia XPath


Typy wyrażeń:

- node-set,
- boolean,
- number,
- string.

Węzeł bieżący (context node).

Poruszanie się w hierarchii elementów:

- /
- /book/section
- section/para



Location paths


Ścieżka XPath złożona z kroków.

Opis kroku:

- oś,
- test węzła,
- predykaty.

Przykłady:

- /child::book/child::section
- child::para[attribute::type="warning"]



Osie (axes)

"Kierunki" poruszania się po modelu dokumentu:

- child
- descendant
- parent
- ancestor
- following-sibling
- preceding-sibling
- following
- preceding
- attribute
- namespace
- self
- descendand-or-self
- ancestor-or-self



Testy węzłów

Podstawowy typ węzła:

- dla osi `attribute`: atrybut,
- dla osi `namespace`: przestrzeń nazw,
- dla pozostałych osi: element.

Testy:

- nazwa węzła,
- * - wszystkie węzły podstawowego typu
- `node()`
- `text()`
- `comment()`
- `processing-instruction()`
- `processing-instruction(target-name)`

Zapis skrócony

Skróty:

- `child::` można pominąć,
- `@` `attribute::`
- `//` `/descendant-or-self::node()/`
- `.` `self::node()`
- `..` `parent::node()`

Zapis pełny vs. skrócony – przykład:

- `./para`
- `self::node()/descendant-or-self::node()/child:para`

Predykaty

Dowolne wyrażenie.

Interpretacja:

- `number` – prawda, gdy równy pozycji węzła w kontekście,
- `string` – prawda, gdy niepusty,
- `node-set` – prawda, gdy niepusty.

Przykłady

```
para
*
*/para
@name
@*
/doc/chapter[5]/section[2]
chapter//para
chapter[title]
chapter[title="Introduction"]
employee[@secretary and @assistant]
```

Ważniejsze funkcje

Operatory: + - * / > >= < <= and or ...

```
last()
position()
count(node-set)
name(node-set?)
string(object?)
concat(string, string, string*)
contains(string, string)
not(boolean)
```

Języki formatowania dokumentów strukturalnych

SGML:

- FOSI (Formatting Output Specification Instance):
 - specyfikacja MIL-PRF-28001,
 - zbyt małe możliwości dla ogólnych zastosowań.
- DSSSL (Document Style Semantics and Specification Language):
 - ISO/IEC 10179:1996
 - oparty na podzbiore języka Scheme bez efektów ubocznych.

XML:

- CSS (Cascading Style Sheets), stosowane m. in. w HTML-u,
- XSL (Extensible Stylesheet Language):
 - język wysokopoziomowy,
 - deklaratywny, bez efektów ubocznych.

XSL – części składowe

XSLT (XSL Transformations):

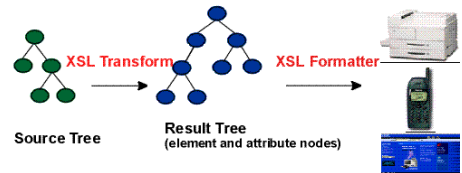
- język opisu przekształceń dokumentów XML,
- składnia XML,
- oparty na dopasowywaniu wzorców,
- przestrzeń nazw: <http://www.w3.org/1999/XSL/Transform>,
- rekomendacja W3C z 16 listopada 1999 r.

XPath (XML Path Language).

XSL:FO (XSL Formatting Objects):

- słownik XML-owy pozwalający definiować formatowanie,
- przestrzeń nazw: <http://www.w3.org/1999/XSL/Format>,
- opisany w rekomendacji XSL 1.0 z 15 października 2001 r.

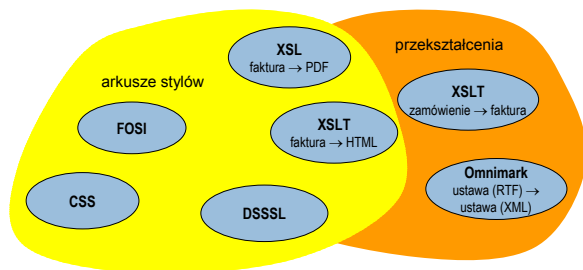
XSL a XSLT



Result XML tree is the result of XSLT processing.

Źródło: Extensible Stylesheet Language (XSL) Version 1.0, W3C Recommendation 15 October 2001 (<http://www.w3.org/TR/xsl/>)

Arkusze stylów a przekształcenia



Zasada działania przekształcenia XSLT

Reguła XSLT:

- ścieżka XPath określająca węzły, dla których reguła obowiązuje,
- treść "wykonywana" w przypadku uruchomienia reguły:
 - tekst i elementy wypisywane na wyjście,
 - instrukcje XSLT.

Sposób przetwarzania:

- wykonaj regułę dla węzła /,
- reguła może rekurencyjnie wywołać reguły dla innych węzłów.

Arkusze stylów/przekształcenie XSLT

Element główny:

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:import href="..." />
  <xsl:output-method="html" />
  <xsl:param name="..." />
  ...
</xsl:stylesheet>
```

Output methods: xml, html, text.

Określanie arkusza stylów dla dokumentu:

```
<?xml-stylesheet type="text/xsl" href="..."?>
```

Podstawy składni – przykład

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="wiersz">
    <poemat>
      <tytul><xsl:value-of select="@tytul" /></tytul>
      <xsl:apply-templates />
    </poemat>
  </xsl:template>

  <xsl:template match="zwrotka">
    <xsl:apply-templates /><odstep />
  </xsl:template>

  <xsl:template match="wers">
    <p><xsl:apply-templates /></p>
  </xsl:template>
</xsl:stylesheet>
```

Przekształcenie – przykład

```
<wiersz tytul="****"> <poemat><tytul>****</tytul>
<zwrotka>
  <wers>aaa</wers> <p>aaa</p>
  <wers>bbb</wers> <p>bbb</p>
</zwrotka> <odstep/>
<zwrotka>
  <wers>ccc</wers> <p>ccc</p>
  <wers>ddd</wers> <p>ddd</p>
</zwrotka> <odstep/>
</wiersz> </poemat>
```

Wbudowane reguły

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

<xsl:template match="text()|@*">
  <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="processing-instruction()|comment()"/>
```

Generowanie dokumentu wyjściowego

Elementy i tekst literalnie podane w przekształceniu.

Instrukcje generujące:

- `<xsl:value-of select=string-expression/>`
- `<xsl:element name=.../>`
- `<xsl:attribute name=.../>`
- `<xsl:text/>`
- `<xsl:processing-instruction name=.../>`
- `<xsl:comment/>`
- `<xsl:copy/>`

Przetwarzanie warunkowe: if

```
<xsl:template match="item">
  <tr>
    <xsl:if test="position() mod 2 = 0">
      <xsl:attribute name="bgcolor">yellow</xsl:attribute>
    </xsl:if>
    <xsl:apply-templates/>
  </tr>
</xsl:template>
```

Przetwarzanie warunkowe: choose

```
<xsl:template match="orderedlist/item">
  <xsl:variable name="level"
    select="count(ancestor::orderedlist) mod 3"/>
  <xsl:choose>
    <xsl:when test='$level=1'>
      <xsl:number format="i"/>
    </xsl:when>
    <xsl:when test='$level=2'>
      <xsl:number format="a"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:number format="1"/>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:text> . </xsl:text>
  <xsl:apply-templates>
</xsl:template>
```

Pętla

```
<xsl:template match="index">
  <h1>Index</h1>
  <xsl:for-each select="//keyword">
    <p><xsl:value-of select="text()"/></p>
  </xsl:for-each>
</xsl:template>
```

Rodzaje przetwarzania XSLT (1)

Przetwarzanie sterowane strukturą dokumentu źródłowego:

- przechodzimy po strukturze dokumentu źródłowego,
- generujemy fragmenty struktury dokumentu wyjściowego.

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



Rodzaje przetwarzania XSLT (2)

Przetwarzanie sterowane strukturą dokumentu wyjściowego:

- jedna duża reguła dla węzła root,
- generujemy strukturę dokumentu docelowego,
- wyciągamy odpowiednie wartości z dokumentu źródłowego.

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns="http://www.w3.org/TR/xhtml1/strict">
  <xsl:template match="/">
    <html><head><title>Expense Report Summary</title></head>
    <body>
      <h1>Company: <xsl:value-of select="company/name"/></h1>
      <p>Total Amount:
        <xsl:value-of select="expense-report/total"/></p>
    </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```



Zmienne

Jak w funkcyjnych językach programowania:

- brak instrukcji przypisania,
- brak efektów ubocznych.

Deklaracja:

- <xsl:variable name=.../>

wartość:

- atrybut select → wyrażenie odpowiedniego typu,
- zawartość elementu → fragment drzewa wynikowego,

Użycie:

- w wyrażeniach: $\$name$,
- <xsl:copy-of select=expression/>



Zaawansowane możliwości XSLT

Sortowanie węzłów.

Wzorce nazwane:

- wywoływane jak podprogramy (procedury),
- przekazywanie parametrów,
- rekursja.

Tryby przetwarzania (modes):

- przełączanie między trybami,
- niezależnie definiowane wzorce dla każdego trybu.



Wykorzystanie rekursji w XSLT (1)

Sposób na brak "prawdziwych" zmiennych i pętli iteracyjnych.

Przykład:

```
<xsl:template name="Books">
  <xsl:param name="cnt" select="1"/>
  <xsl:if test="$cnt > 0">
    <book>
      <tr><xsl:apply-templates/></tr>
    </book>
    <xsl:call-template name="Books">
      <xsl:with-param name="cnt" select="$cnt - 1"/>
    </xsl:call-template>
  </xsl:if>
</xsl:template>
```



Wykorzystanie rekursji w XSLT (2)

Przykład – c.d.:

```
<xsl:template match="book">
  <xsl:choose>
    <xsl:when test="@count">
      <xsl:call-template name="Books">
        <xsl:with-param name="cnt" select="@count"/>
      </xsl:call-template>
    </xsl:when>
    <xsl:otherwise>
      <xsl:call-template name="Books"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```



Generowanie przekształceń XSL (1)

Problem:

- źródło przekształcenia nie zawiera metainformacji o strukturze dokumentu,
- metainformacje pojawiają się na wyjściu.

```
<wniosek-urlopowy>
<wniosek>
<pracownik>Szymon Ziolo</pracownik>
<rodzaj>wypoczynkowy</rodzaj>
<od>2003-06-20</od>
<do>2003-06-27</do>
<dni-roboczych>6</dni-roboczych>
</wniosek>
<decyzja>
<zgoda>1</zgoda>
<zastepca>Jan Kowalski</zastepca>
</decyzja>
</wniosek-urlopowy>
```

Wniosek urlopowy

Wniosek

Pracownik:	Szymon Ziolo
Rodzaj urlopu:	wypoczynkowy
Od dnia:	2003-06-20
Do dnia:	2003-06-27
Dość dni roboczych:	6
Decyzja przelozonego:	
Zgoda przelozonego:	tak
Zastepca:	Jan Kowalski



Generowanie przekształceń XSL (2)

Rozwiązanie:

- zapisanie metainformacji w szablonie,
- generowanie przekształcenia z szablonu.

```
<dokument nazwa="wniosek-urlopowy"
etykieta="Wniosek urlopowy">
<sekcja nazwa="wniosek" etykieta="Wniosek">
<tr><td><xsl:value-of select="@etykieta" /></td></tr>
<td><xsl:value-of select="@etykieta" /></td></tr>
<td><xsl:value-of select="@etykieta" /></td></tr>
<td><xsl:value-of select="@etykieta" /></td></tr>
<td><xsl:value-of select="@etykieta" /></td></tr>
</sekcja>
<sekcja nazwa="decyzja" etykieta="Decyzja przelozonego">
<td><xsl:value-of select="@etykieta" /></td></tr>
<td><xsl:value-of select="@etykieta" /></td></tr>
</sekcja>
</dokument>
```



Generator – przykład (1)

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:res="http://www.w3.org/1999/XSL/TransformAlias">
<xsl:namespace-alias stylesheet-prefix="res"
result-prefix="xsl"/>

<xsl:template match="/">
<res:stylesheet version="1.0">
<res:output method="html"/>
<xsl:apply-templates/>
</res:stylesheet>
</xsl:template>

<xsl:template match="sekcja">
<res:template match="{@nazwa}">
<p><b><xsl:value-of select="@etykieta" /></b></p>
<table><res:apply-templates/></table>
</res:template>
<xsl:apply-templates/>
</xsl:template>
```



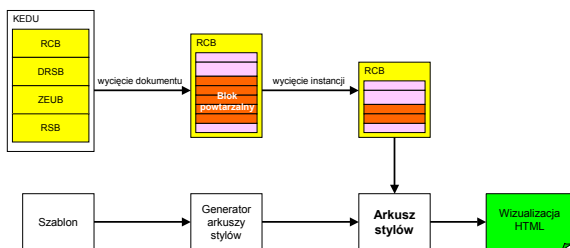
Generator – przykład (2)

```
<xsl:template match="pole">
<res:template match="{@nazwa}">
<td><b><xsl:value-of select="@etykieta" /></b></td>
</td></tr>
<xsl:choose>
<xsl:when test="@typ='boolean'">
<res:choose>
<res:when test="text()='1'">tak</res:when>
<res:otherwise>nie</res:otherwise>
</res:choose>
</xsl:when>
<xsl:otherwise>
<res:value-of select="text()" />
</xsl:otherwise>
</xsl:choose>
</b></td></tr>
</res:template>
<xsl:apply-templates/>
</xsl:template>

</xsl:stylesheet>
```



Zastosowanie w projekcie: KEDU ZUS



KEDU ZUS – przykład wizualizacji

IV. B. ZESTAWIENIE NALEŻNYCH SKŁADEK NA UBEZPIECZENIA SPOŁECZNE I UBEZPIECZENIE ZDROWOTNE	02	34	03	04	05	06	07	08
1) Wynik K, wsiłachowy NIE jest przesłany								
2) Wynik K, wsiłachowy NIE jest przesłany								
3) Wynik K, wsiłachowy NIE jest przesłany								
4) Wynik K, wsiłachowy NIE jest przesłany								
5) Wynik K, wsiłachowy NIE jest przesłany								
6) Wynik K, wsiłachowy NIE jest przesłany								
7) Wynik K, wsiłachowy NIE jest przesłany								
8) Wynik K, wsiłachowy NIE jest przesłany								
9) Wynik K, wsiłachowy NIE jest przesłany								
10) Wynik K, wsiłachowy NIE jest przesłany								
11) Wynik K, wsiłachowy NIE jest przesłany								
12) Wynik K, wsiłachowy NIE jest przesłany								
13) Wynik K, wsiłachowy NIE jest przesłany								
14) Wynik K, wsiłachowy NIE jest przesłany								
15) Wynik K, wsiłachowy NIE jest przesłany								
16) Wynik K, wsiłachowy NIE jest przesłany								
17) Wynik K, wsiłachowy NIE jest przesłany								
18) Wynik K, wsiłachowy NIE jest przesłany								
19) Wynik K, wsiłachowy NIE jest przesłany								
20) Wynik K, wsiłachowy NIE jest przesłany								
21) Wynik K, wsiłachowy NIE jest przesłany								
22) Wynik K, wsiłachowy NIE jest przesłany								
23) Wynik K, wsiłachowy NIE jest przesłany								
24) Wynik K, wsiłachowy NIE jest przesłany								
25) Wynik K, wsiłachowy NIE jest przesłany								
26) Wynik K, wsiłachowy NIE jest przesłany								
27) Wynik K, wsiłachowy NIE jest przesłany								
28) Wynik K, wsiłachowy NIE jest przesłany								
29) Wynik K, wsiłachowy NIE jest przesłany								
30) Wynik K, wsiłachowy NIE jest przesłany								
31) Wynik K, wsiłachowy NIE jest przesłany								
32) Wynik K, wsiłachowy NIE jest przesłany								
33) Wynik K, wsiłachowy NIE jest przesłany								
34) Wynik K, wsiłachowy NIE jest przesłany								
35) Wynik K, wsiłachowy NIE jest przesłany								
36) Wynik K, wsiłachowy NIE jest przesłany								
37) Wynik K, wsiłachowy NIE jest przesłany								
38) Wynik K, wsiłachowy NIE jest przesłany								
39) Wynik K, wsiłachowy NIE jest przesłany								
40) Wynik K, wsiłachowy NIE jest przesłany								
41) Wynik K, wsiłachowy NIE jest przesłany								
42) Wynik K, wsiłachowy NIE jest przesłany								
43) Wynik K, wsiłachowy NIE jest przesłany								
44) Wynik K, wsiłachowy NIE jest przesłany								
45) Wynik K, wsiłachowy NIE jest przesłany								
46) Wynik K, wsiłachowy NIE jest przesłany								
47) Wynik K, wsiłachowy NIE jest przesłany								
48) Wynik K, wsiłachowy NIE jest przesłany								
49) Wynik K, wsiłachowy NIE jest przesłany								
50) Wynik K, wsiłachowy NIE jest przesłany								
51) Wynik K, wsiłachowy NIE jest przesłany								
52) Wynik K, wsiłachowy NIE jest przesłany								
53) Wynik K, wsiłachowy NIE jest przesłany								
54) Wynik K, wsiłachowy NIE jest przesłany								
55) Wynik K, wsiłachowy NIE jest przesłany								
56) Wynik K, wsiłachowy NIE jest przesłany								
57) Wynik K, wsiłachowy NIE jest przesłany								
58) Wynik K, wsiłachowy NIE jest przesłany								
59) Wynik K, wsiłachowy NIE jest przesłany								
60) Wynik K, wsiłachowy NIE jest przesłany								
61) Wynik K, wsiłachowy NIE jest przesłany								
62) Wynik K, wsiłachowy NIE jest przesłany								
63) Wynik K, wsiłachowy NIE jest przesłany								
64) Wynik K, wsiłachowy NIE jest przesłany								
65) Wynik K, wsiłachowy NIE jest przesłany								
66) Wynik K, wsiłachowy NIE jest przesłany								
67) Wynik K, wsiłachowy NIE jest przesłany								
68) Wynik K, wsiłachowy NIE jest przesłany								
69) Wynik K, wsiłachowy NIE jest przesłany								
70) Wynik K, wsiłachowy NIE jest przesłany								
71) Wynik K, wsiłachowy NIE jest przesłany								
72) Wynik K, wsiłachowy NIE jest przesłany								
73) Wynik K, wsiłachowy NIE jest przesłany								
74) Wynik K, wsiłachowy NIE jest przesłany								
75) Wynik K, wsiłachowy NIE jest przesłany								
76) Wynik K, wsiłachowy NIE jest przesłany								
77) Wynik K, wsiłachowy NIE jest przesłany								
78) Wynik K, wsiłachowy NIE jest przesłany								
79) Wynik K, wsiłachowy NIE jest przesłany								
80) Wynik K, wsiłachowy NIE jest przesłany								
81) Wynik K, wsiłachowy NIE jest przesłany								
82) Wynik K, wsiłachowy NIE jest przesłany								
83) Wynik K, wsiłachowy NIE jest przesłany								
84) Wynik K, wsiłachowy NIE jest przesłany								
85) Wynik K, wsiłachowy NIE jest przesłany								
86) Wynik K, wsiłachowy NIE jest przesłany								
87) Wynik K, wsiłachowy NIE jest przesłany								
88) Wynik K, wsiłachowy NIE jest przesłany								
89) Wynik K, wsiłachowy NIE jest przesłany								
90) Wynik K, wsiłachowy NIE jest przesłany								
91) Wynik K, wsiłachowy NIE jest przesłany								
92) Wynik K, wsiłachowy NIE jest przesłany								
93) Wynik K, wsiłachowy NIE jest przesłany								
94) Wynik K, wsiłachowy NIE jest przesłany								
95) Wynik K, wsiłachowy NIE jest przesłany								
96) Wynik K, wsiłachowy NIE jest przesłany								
97) Wynik K, wsiłachowy NIE jest przesłany								
98) Wynik K, wsiłachowy NIE jest przesłany								
99) Wynik K, wsiłachowy NIE jest przesłany								
100) Wynik K, wsiłachowy NIE jest przesłany								



Zastosowanie w projekcie: Era DCO

Document Collection Office:

- system obiegu dokumentów strukturalnych,
- edycja przy pomocy formularzy HTML,
- zastosowanie: obieg protokołów z przeglądów stacji bazowych.

Szablon dokumentu:

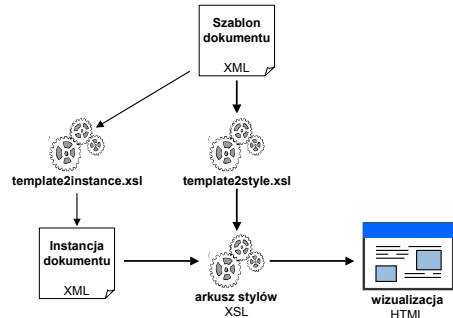
- struktura (nazwy pól, typy, kontrolki edycyjne),
- pola automatycznie obliczane,
- reguły walidacji,
- role i uprawnienia,
- proces obiegu dokumentu.

Generatory:

- schematu XML Schema,
- pustej instancji dokumentu,
- arkuszy stylów,
- ewaluatorów pól automatycznie obliczanych,
- walidatorów.



Era DCO – schemat przetwarzania



Formatting Objects – przykład drzewa wynikowego

```
<?xml version="1.0" encoding="utf-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
...
  <fo:page-sequence>
    <fo:flow>
      <fo:block font-size="18pt" font-weight="bold"
        text-align="center">Preface</fo:block>
      <fo:block font-size="12pt" space-before="1pc"
        text-align="justified"> This is a simple
        test document. It shows a
        <fo:inline font-style="italic">partial</fo:inline>
        fo-result tree (page layout missing).</fo:block>
    </fo:flow>
  </fo:page-sequence>
...
</fo:root>
```



Formatting Objects – przykłady reguł

```
<xsl:template match="chapter">
  <fo:flow><xsl:apply-templates/></fo:flow>
</xsl:template>

<xsl:template match="chapter/title">
  <fo:block font-size="12pt" font-weight="bold"
    text-align="center">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="para">
  <fo:block font-size="12pt" space-before="1pc"
    text-align="justified">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="emphasis">
  <fo:inline font-style="italic"><xsl:apply-templates/>
</fo:inline>
</xsl:template>
```



Narzędzia

Procesory XSLT:

- XT, James Clark (Java),
- Oracle XML Parser for Java / C / PL-SQL,
- Xalan (Java, C++),
- SAXON, Michael H. Kay (Java),
- Sablotron (C++, open source),
- Microsoft XML Core Services (MSXML 4.0).

Procesor XSL:FO:

- FOP (open source; generuje dokumenty w formacie PDF).

Edytory XSLT:

- xmilsy 2004,
- Xselerator XSL Editor/Debugger,
- xslide – Emacs Major Mode for XSL Stylesheets.



Rozwój XSL-a

XSLT 2.0 Working Draft:

- wsparcie dla XML Schema,
- grupowanie węzłów (for-each-group),
- definiowanie i wywoływanie własnych funkcji,
- generowanie wielu dokumentów wyjściowych,
- ...

XPath 2.0 Working Draft

- wyrażenia warunkowe (if i inne),
- kwantyfikatory,
- typ "sekwencja", pętla,
- ...



Gdzie szukać dalej

The Extensible Stylesheet Language Family:

www.w3.org/Style/XSL

XSLT Tutorial:

www.zvon.org/xxl/XSLTutorial/Output

TopXML:

www.topxml.com/xsl

www.topxml.com/xsltstylesheets

Sławomir Tyszko, Rekurencyjne szablony w XSLT

Software 2.0, nr 6/2002, Wydawnictwo Software

Szymon Ziolo, XSLT do kwadratu

Software 2.0, nr 6/2003, Wydawnictwo Software

