JURNAL MATEMATIKA DAN KOMPUTER
Vol. 6. No. 3, 128 - 136, Desember 2003, ISSN : 1410-8518

Transformasi Dokumen XML

Aris Puji Widodo
Jurusan Matematika FMIPA UNDIP Semarang
E-Mail : masarisdong@yahoo.com

Abstrak

Dokumen Extensible Markup Language (XML) merupakan dokumen standar yang memiliki sifat menstrukturkan informasi, portable dan tidak tergantung pada satu platform tertentu, sehingga memberikan efisiensi dalam melakukan proses manipulasi informasi pada suatu dokumen. Pada makalah ini dibahas mengenai transformasi dokumen XML ke format dokumen lainnya dengan memanfaatkan Cocoon Framework, Parser dan sejumlah Prosesor XML Stylesheets Language (XSL) yang merupakan hasil open source project. Dokumen XML dibangkitkan melalui Generator, kemudian dokumen XML ditransformasikan ke dokumen lainnya melalui Transformer dengan menggunakan stylesheet untuk masing-masing format dokumen hasil transformasi. Dokumen XML dikumpulkan pada Aggregator dan selanjutnya dilakukan rendering terhadap dokumen XML melalui serializer untuk menghasilkan format dokumen lainnya, untuk dapat ditampilkan ke berbagai persentasi yang bervariasi.

Kata Kunci : XML, platform, cocoon framework, prosesor, dan stylesheet.

1. PENDAHULUAN

XML adalah *markup language* yang dikembangkan oleh *World Wide Web Consortium* (W3C), dengan tujuan utamanya adalah untuk mengatasi sejumlah keterbatasan yang terdapat pada *Hyper Text Markup Language* (HTML). XML dan HTML merupakan subset dari *Structured Generalized Markup Language*) (SGML) (W3C, 2002), (Marchal & Benoit, 2000). Secara aktual XML lebih mirip SGML dibandingkan dengan HTML, karena HTML hanya digunakan untuk mendiskripsikan *web pages*. Tetapi XML adalah *language* yang digunakan untuk mendiskripsikan dan memanipulasi struktur dokumen, serta menawarkan beberapa mekanisme untuk memanipulasi informasi yang bebas *platform*. Sebagai contoh, XML digunakan oleh StarOffice dan AbiWord untuk salah satu format penyimpanan dokumen dan XML digunakan untuk menyimpan obyek persisten dalam dokumen perkantoran (Widyani, 2001).

XML berkonsentrasi pada struktur informasi, tetapi tidak berkonsentrasi untuk menampilkan dokumen informasi. Untuk menampilkan dokumen XML

dibutuhkan suatu format atau *style* dari dokumen XML, dimana format atau *style* tersebut secara langsung dihubungkan dan merupakan suatu turunan dari struktur dokumen XML yang diorganisasikan dalam *stylesheets*. *Stylesheets* yang direkomendasikan oleh W3C diantaranya adalah XSL (Adler et. Al, 2002), (Clark & James, 2002). XSL digunakan untuk menstrasformasikan dokumen XML ke format dokumen HTML, Text, *Rich Text Format* (RTF), XHTML, *Portable Data Format* (PDF), dan *PostScrip* (PS) (Holzner & Steven, 2002). Proses transformasi dokumen XML tersebut diatas membutuhkan suatu prosesor XSL yang sesuai dengan persentasi untuk masing-masing dokumen hasil transformasi.

Untuk melakukan efisiensi dalam pengorganisasian manajemen dokumen, maka dokumen dari satu sumber harus dapat ditampilkan ke berbagai persentasi yang bervariasi. Hal ini dapat dilakukan dengan cara membuat suatu dokumen sumber dalam bentuk format standar, sehingga dari format tersebut akan dapat ditampilkan ke berbagai format persentasi yang bervariasi. Mekanisme ini dapat dilakukan dengan memanfaatkan kelebihan-kelebihan yang ditawarkan XML dan XSL.

Pada makalah ini menguraikan tentang transformasi dokumen XML ke format dokumen lainnya dengan memanfaatkan *parser*, prosesor-prosesor XSL dan *cocoon framework* yang secara keseluruhan merupakan hasil *open source project*.

2. XML

XML adalah sebuah standar yang digunakan untuk menstrukturkan informasi dalam sebuah dokumen menjadi sejumlah bagian dan untuk mengidentifikasi bagian tersebut. Dokumen bukan hanya berupa dokumen teks, tetapi termasuk juga data gambar, persamaan matematika, rumus bangun kimia, dan berbagai jenis informasi yang dapat distrukturkan (Anderson et. Al., 2000), (W3C, 2002), (Marchal & Benoit, 2000).

XML menstrukturkan informasi dalam bentuk sekumpulan elemen dan atribut. Sebuah dokumen XML minimal mempunyai sebuah elemen, yaitu *root element*. Sebuah elemen bisa mempunyai elemen lain sebagai elemen anak. Selain itu, setiap elemen juga bisa mempunyai atribut sebagai penjelas elemen tersebut.

Setiap elemen di dalam dokumen XML dibatasi dengan *markup* yang berbentuk sebagai pasangan *tag*.

Contoh bentuk sebuah dokumen XML diberikan pada kode 1.

```
<page>
<title>Memakai file pageOne.xml</title>
<s1 title="Section one : Aris Puji Widodo Make Webapp With Cocoon version 2.0.2-dev">
    <p>Dosen Matematika</p>
    <p>Fakultas MIPA</p>
    <p>Universitas Diponegoro</p>
    <p>Semarang</p>
    <p>2003</p>
    <p>text pada bagian satu</p>
</s1>
</page>
```

Kode 1. Dokumen XML.

Maksud dari pembuatan XML adalah sebagai format universal untuk menstrukturkan dokumen dan data pada *web* (W3C, 2002), walaupun pada kenyataannya dokumen tersebut tidak harus ditampilkan di *web* dan aplikasi yang menggunakannya tidak harus berbasis *web*. XML dapat digunakan untuk aplikasi seperti *word processor*, *spreadsheet, database*, digunakan di beberapa bidang antara lain kesehatan, kimia, bisnis, hukum, matematika, dan tidak tergantung kepada sistem operasi tertentu.

3. PEMROSESAN XSL

XSL adalah bahasa yang digunakan untuk mengekpresikan suatu *stylesheet*. XSL terdiri dari *XSL Transformations* (XSLT) dan *XSL Formating Objects* (XSL-FO) (Holzner & Steven, 2002). XSLT kebanyakan digunakan untuk mentransformasikan dokumen XML ke HTML, sedangkan XSL-FO digunakan untuk melakukan konversi dokumen XML ke format PDF.

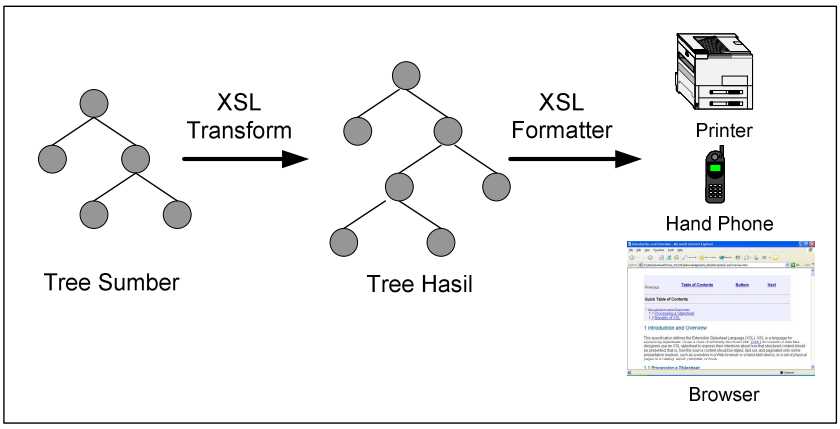
Stylesheets diproses menggunakan XSL *stylesheet* prosesor, yang dapat menerima dokumen atau data dalam bentuk XML dan XSL *stylesheet* akan menghasilkan presentasi isi dokumen XML sumber yang dirancang dalam

stylesheet. Terdapat 2 aspek dalam proses pembentukan presentasi (Adler et. al. 2002), (Clark & James, 2002) yaitu :

1. Mengkonstruksikan hasil berupa *tree* dari *tree* dokumen XML sumber.
2. Menginterprestasikan hasil yang berupa *tree* untuk menghasilkan suatu format hasil yang sesuai dengan presentasi pada *display* atau media yang akan digunakan untuk menampilkan dokumen hasil transformasi.

Untuk aspek yang pertama disebut *tree transformation*, sedangkan aspek yang kedua disebut *formatting* yang dijalankan oleh *formatter*.

Untuk detail model konseptual pemrosesan terhadap XSL diberikan pada gambar 1 sebagai berikut :



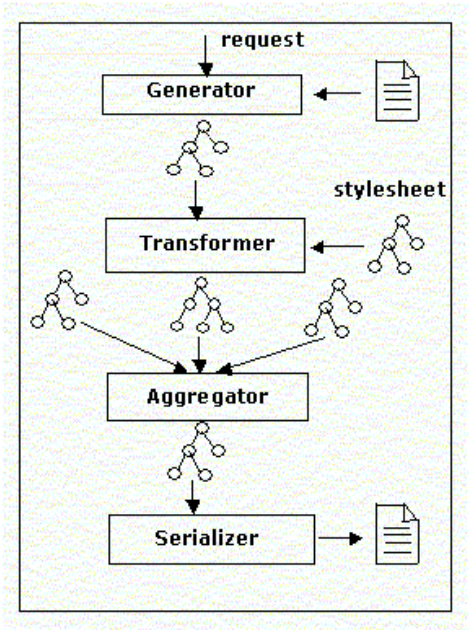
Gambar 1. Model Konseptual Pemrosesan XSL.

4. MEKANISME TRANFORMASI DOKUMEN XML

Untuk melakukan transformasi dokumen XML digunakan *Cocoon framework* yang merupakan *open source project* dikembangkan oleh (Mazzocchi, 2002). *Cocoon framework* adalah *framework* XML *publish* untuk meningkatkan penggunaan teknologi XML dan XSLT pada *server application*. Adapun untuk mekanisme dasar cocoon dalam melakukan pemrosesan dokumen XML diberikan pada gambar 2.

Dokumen XML dibangkitkan melalui *Generator*, kemudian dokumen XML ditransformasikan ke dokumen lainnya melalui *Transformer* dengan menggunakan *stylesheet*. Dokumen XML dikumpulkan pada *Aggregator* dan

selanjutnya dilakukan *rendering* terhadap dokumen XML melalui *serializer* untuk menghasilkan format dokumen yang lain. Pemrosesan yang dilakukan bersifat *pipeline processing*, dengan menggunakan konfigurasi *sitemap* yang dapat dilakukan *setting* secara dinamik terhadap *pipeline processing* yang terdiri dari *generator, multiple transformer* dan *serializer*.



Gambar 2. Mekanisme Pemrosesan Dokumen XML pada *Cocoon Framework*.

5. IMPLEMENTASI TRANSFORMASI DOKUMEN XML

Untuk contoh implementasi dilakukan transformasi dokumen XML ke format dokumen HTML dan PDF. Informasi dokumen distrukturkan menggunakan XML sebagai dokumen sumber yang akan ditransformasikan ke format dokumen HTML dan PDF. Struktur dokumen XML sumber menggunakan dokumen XML pada kode 1.

Kemudian dibuat *stylesheet* untuk masing-masing format hasil transformasi ke dokumen HTML dan PDF. Untuk transformasi ke format dokumen HTML digunakan XSLT, sedangkan untuk transformasi ke format dokumen PDF digunakan XSL-FO. yang diberikan pada kode 2 dan 3.

```
<!-- Pendefinisian Stylesheet menggunakan XSLT -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<!-- generate HTML skeleton pada elemen root -->
<xsl:template match="/">
  <html>
     <!-- Body untuk halaman HTML -->
  </html>
</xsl:template>
<!-- Melakukan konversi ke heading HTML -->
</xsl:stylesheet>
```

Kode 2. *Stylesheet* untuk Format Dokumen HTML.

```
<!-- Pendefinisian Stylesheet menggunakan XSL-FO -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
                xmlns:fo="http://www.w3.org/1999/XSL/Format">
<!-- Melakukan generate struktur halaman PDF -->
    <xsl:template match="/">
        <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
            <!-- Body loyout halaman PDF -->
        </fo:root>
    </xsl:template>
<!-- Pemrosesan terhadap paragraphs -->
<!-- Melakukan konversi ke heading XSL-FO -->
    </xsl:template>
</xsl:stylesheet>
```

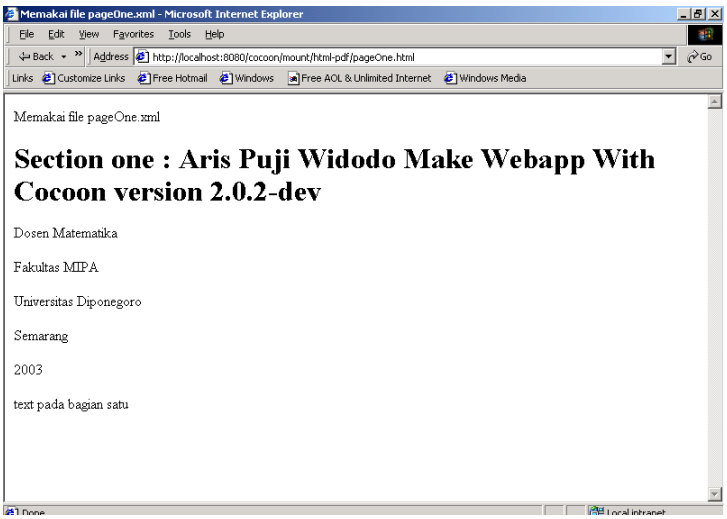
Kode 3. *Stylesheet* untuk Format Dokumen PDF.

Kemudian untuk pemrosesan secara *pipeline processing* dilakukan dengan membuat *sitemap*, yang diberikan pada kode 4. Untuk melakukan transformasi ke format dokumen HTML digunakan prosesor XSL JTidy, sedangkan untuk format PDF digunakan prosesor XSL FOP. *Sitemap* dapat memilih *pipeline* tertentu berdasarkan pada *request* yang terjadi melalui *browser*.

```
<!-- Pendefinisian sitemap -->
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
    <map:pipelines>
        <map:pipeline>
            <!-- memberikan responds request PDF dengan memproses stylesheet PDF -
->
            <map:match pattern="*.pdf">
                <map:generate src="{1}.xml"/>
                <map:transform src="doc2pdf.xsl"/>
                <map:serialize type="fo2pdf"/>
            </map:match>
        </map:pipeline>
    </map:pipelines>
</map:sitemap>
```

Kode 4. *Sitemap* untuk *Pipeline Processing*.

Untuk hasil transformasi dokumen XML ke format dokumen HTML dan PDF yang dijalankan pada *browser* diberikan pada gambar 3 dan 4. Informasi yang terdapat pada masing-masing hasil transformasi menggunakan dokumen sumber pada kode 1, dimana dengan satu format dokumen sumber dapat ditampilkan pada persentasi yang berbeda.



ERROR: ioerror
OFFENDING COMMAND: image

STACK: