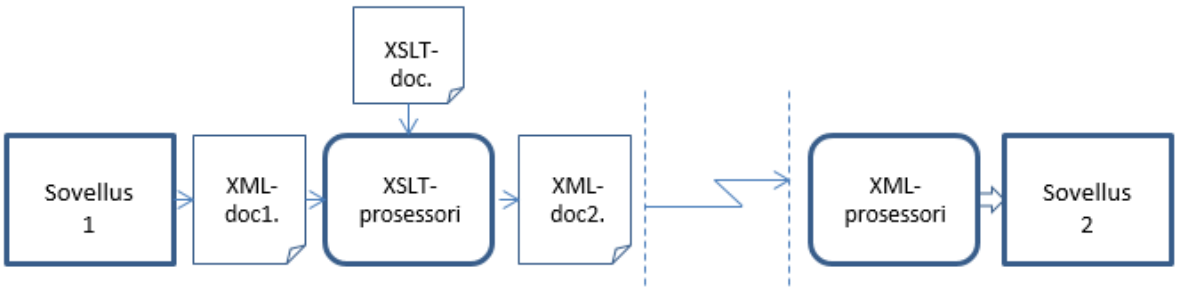
XML_XSLT_teoria_1.docx
ELEC-C8203 - Automaatiojärjestelmät 2
XML Tietointegraatio

XSLT-kieli muunnoksien määrittelyyn - teoria



Kuva 1. 'XSLT-prosessori' (XML API) muuntaa dokumentin (doc1.xml) toiseen muotoon (doc2.xml) lukemansa XSLT-dokumentin sääntöjen ohjaamana.

1. XSLT:n käyttö tietointegroinnissa

Järjestelmien tietointegraatiossa tarvitaan muunnoksia kielestä toiseen. Eri organisaatioiden tietojärjestelmät käyttävät usein erilaista XML-esitystapaa samankin sovellusalueen tietojen esittämiseen.
Sovelluksen kutsumana 'XSLT-prosessori' (API) muuntaa lähtödokumentin (doc1.xml) uuteen muotoon tulosdokumentiksi (doc2.xml) XSLT-dokumentin sisältämien sääntöjen ohjaamana. Yleensä tavoitteena on lähtödokumentin tietosisällön esittäminen jossain uudessa rakennemuodossa.

- Uuteen dokumenttiin voidaan luoda aivan toisenlainen rakenne uusine elementteineen, attribuutteineen ja tekstisisältöineen
- Toisaalta uuteen dokumenttiin voidaan myös kopioida lähdedokumentin elementtejä ja attribuutteja tietosisältöineen

Uuden dokumentin rakenne ja tietosisältö voi siis olla (melkein) minkälainen tahansa. Sen ei tarvitse olla edes XML-dokumentti; se voi olla html-dokumentti tai pelkkä tekstidokumentti.

2. XSLT-prosessoinnin vaiheet

Muunnoksen prosessoinnin vaiheet ovat seuraavat:

1. XSLT-prosessori lukee XML-dokumentin ja XSLT-tyylisivun (*XSLT-stylesheet*)
2. Se analysoi ja jäsentää XML-dokumentin solmupuuksi (*node-tree*)
3. Se käy läpi solmupuuta solmu kerrallaan aloittaen juurisolmusta (*/*)
4. Ohjeet/säännöt ko. solmun käsittelyyn se hakee XSLT-tyylisivun ko. solmun käsittelyyn tarkoitetusta sapluunasta/mallinteesta (*template*)
5. Sapluunan *match*-attribuutin *XPath*-lauseke määrittää sen solmun/solmujoukon, jonka käsittelyyn sapluunan säännöt on tarkoitettu
6. Prosessori aloittaa sääntöjen lukemisen juuri-sapluunasta (*root template*), joka on oltava jokaisessa XSLT-dokumentissa*: <xsl:template match="/">*

7. Säännöt voivat ohjata joko kirjoitusta tulosdokumenttiin tai käsiteltävän solmujoukon jatkoprosessointia ali-sapluunakutsuineen: *<xsl:apply-templates select="sub-node">*

3. XSLT-tyylisivun rakenne

XSLT-tyylisivu koostuu joukosta määrityksiä ja *sapluunoita (template)*

- Määrityksillä muokataan tyylisivun tuottamaan dokumenttiin liittyviä asioita tai määritellään muuttujia myöhempää käyttöä varten.
- Sapluunat *(template)* sisältävät dokumentin muunnossa käytettävät säännöt, sisään luetun dokumentin käsittelyohjeita ja tuotetun dokumentin elementtejä.

Osa tuotettavan dokumentin elementeistä on määritelty pysyvästi ja näiden elementtien väliin tuotetaan lisää sisältöä käsittelyohjeiden pohjalta.

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
   <xsl:template match="/">
      <html><head><title>Wonders of the World</title></head>
       <body>
            <h1>Wonders of the World</h1>
        The <xsl:value-of select="ancient_wonders/wonder/name"/>
        is located in <xsl:value-of select="ancient_wonders/wonder/location"/>.
       </body>
       </html>
   </xsl:template>
</xsl:stylesheet>
```

Kuva 2. Esimerkki XSLT-tyylisivusta, joka generoi ancient_wonders.xml dokumentista html-dokumentin (web-sivu).

Tyylisivun kaikki elementit ovat juuri-elementin <xsl:stylesheet> sisällä (ks. Kuva 2).
Tyylisivun tulostusmuoto määritellään <xsl:output> elementillä, jossa tulostuksen metodiksi voidaan määritellä: *xml, html tai text*
Indent-attribuutilla voidaan tulostus sisentää
ESIM: *< xsl:output method="html" indent="yes"/>*
Muunnettavan XML-dokumentin alussa voidaan viitataan tyylisivuun prosessointi-ohjeella
ESIM: *<?xml-stylesheet type="text/xsl" href="ihmeet_lec2ex1.xsl"?>*

4. Template eli sapluuna

Tyylisivujen sisältämät sapluunat määritellään <xsl:template > elementeillä
ESIM: *<xsl:template match="Xpath-expression">*
XSLT muunnos aloitetaan juuri-sapluunan (root template) prosessoinnilla.
<xsl:template match="/"> … muunnossääntöjä ….</xsl:template >
Se on oltava jokaisessa xslt-dokumentissa ja se määrittää miten xml-dokumentin juurisolmua prosessoidaan. Muunnossääntöjen prosessointi jatkuu viimeiseen juuri-sapluunan sääntöön asti. Se voi sisältää muiden sapluunoiden soveltamiskäskyjä:
ESIM: *<xsl:apply-templates select="xpath-exp"/>*

Sapluuna koostuu kahdesta osasta

1. sapluunan laukaiseva sääntö
2. sapluunan sisältö

Sääntö määritetään XPath-lausekkeella.
Sapluunan sisällön rakenteen määrää tuotettavan dokumentin rakenne ja sisältö:

- sapluuna voi tuottaa uusia elementtejä ja attribuutteja (ja tekstisisältöä) tuotettuun dokumenttiin
- sapluuna voi valita sisään luetun dokumentin elementtejä
    - kutsua uusia mallinteita/templateja elementtien perusteella (*<xsl:apply-templates>***)**
    - käsitellä elementit itse
- sapluuna voi sisältää ehtoja ja toistoja

(Ks. Online tutoriaali: http://www.w3schools.com/xsl/default.asp)

5. Sapluunoiden valinnan ohjaus

Sapluuna voi käynnistää sapluunan haun mille tahansa XML-dokumentin osalle (solmulle tai solmujoukolle), joka määritellään Xpath-lausekkeella

<xsl:apply-templates select=”Xpath-lauseke”/>

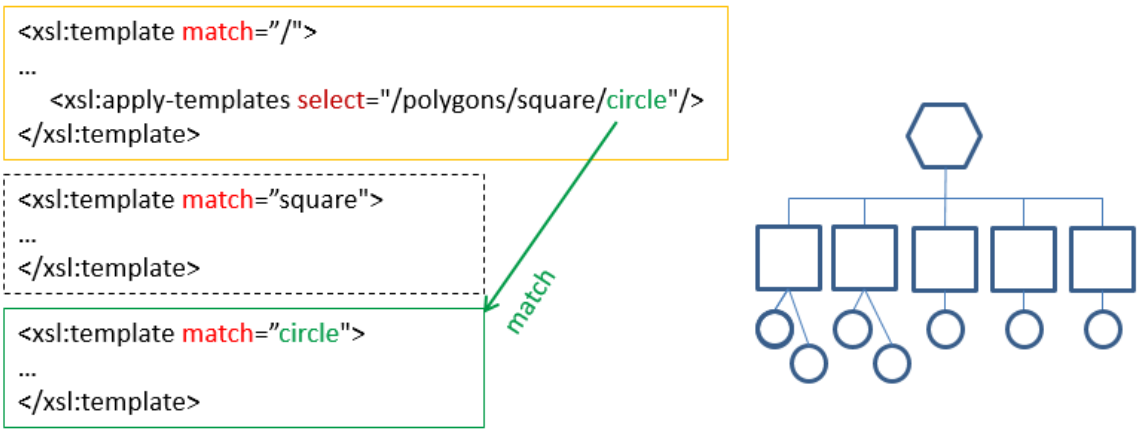
Prosessori käsittelee solmut dokumenttijärjestyksessä (document order)
Myös attribuutti, tekstisirpale tai muu XML-dokumentin osa voidaan valita

- voidaan valita esimerkiksi vain tietyt solmut käsittelyyn:
  <xsl:apply-templates select="name|street|city|postal-code"/>
- valinta voi kohdistua muualla oleviin solmuihin:
  <xsl:apply-templates select="//distant"/>
- Valinta voidaan tehdä myös ehdolliseksi <xsl:apply-templates select="/ancient_wonders/wonder[@build &lt 800]"/>
- (Voidaan myös käsitellä nykyisen solmun alla (sisällä) olevat solmut yhdellä kutsulla: <xsl:apply-templates/> (*Huom: Oletus-sapluuna: Jos kutsutaan sapluunoita ilman select-valintaa, prosessori hakee kullekin solmulle sopivimman löytämänsä sapluunan)

Huomaa, että xslt-prosesori käy läpi kaikki Xpath-lausekkeella (select) valitut solmut kutsuen niihin sopivaa sapluunaa (match) ja toteuttaa sapluunan sisältämät toiminnot (tulosdokumenttiin kirjoittamisen) jokaiselle solmujoukon solmulle (eli xsl:for-each luuppeja ei tarvitse erikseen kirjoittaa)

ESIMERKKI (ks. Kuva 3.)
<xsl:apply-templates komennon *select*:n XPath-lauseke valitsee kutsuttavan templaten:
<xsl:template match=”circle”>
Tämän templaten sääntöjä sovelletaan kaikkien *square*-elementtien kaikkiin *circle*-elementteihin (7 kpl)

(Ks. Online tutoriaali: http://www.w3schools.com/xsl/xsl_apply_templates.asp)



Kuva 3. Esimerkki sapluunoiden valinnasta

6. Rakenteiden luontisäännöt

Elementtien Kopiointi

XSLT mahdollistaa myös kopioinnin sisään luettavasta dokumentista tuotettavaan dokumenttiin kopiointi suoritetaan elementillä <xsl:copy-of select="XPath-ex"/>, jossa XPath-lauseke määrää kopioitavat elementit. Jos lauseketta ei määritellä, kopioidaan nykyinen solmu <xsl:copy-of>.

Uusien elementtien ja attribuuttien luonti

XSLT mahdollistaa uusien elementtien luonnin dynaamisesti seuraavasti:

<xsl:element name="new-element-name">

<!— Uuden elementin tekstisisällön, lapsielementtien ja/tai attribuuttien luonti -->

</xsl:element>

Vastaavasti myös attribuutteja voidaan luoda lennossa seuraavasti:

<xsl:attribute name="new-attribute-name"> Attribuutin arvo *</xsl:attribute>*

Attribuutin luonti liitetään viimeksi määritellyn elementin sisällä kuten seuraavassa esimerkissä:

TEMPLATE ESIMERKKI: Kaikkia lähtödokumentin circle-elementtejä vastaten luodaan 'square' elementit, ja niiden 'color'- attribuutien arvoksi asetetaan arvo 'red'.

```
<xsl:template match="circle">
   <xsl:element name="square">
      <xsl:attribute name="color">red</xsl:attribute>
   </xsl:element>
</xsl:template>
```

7. Tekstisisältöjen käsittelysäännöt

Arvojen poimiminen

XSLT-tyylisivuissa yleensä poimitaan suuri joukko arvoja sisään luettavasta dokumentista ja sijoitetaan tuotettavaan dokumenttiin. Arvojen poiminta tapahtuu elementillä
<xsl:value-of select="xpath-lauseke"/>
elementti poimii XPath-lausekkeen mukaisen arvon ja sijoittaa sen tuotettavaan dokumenttiin Huomaa, että Xpath-lausekkeen määrittämän solmujoukon ainoastaan ensimmäisen solmun arvo tulostetaan. Mikäli arvoa ei ole, elementti ei tuota mitään.

Toistorakenne

XSLT sisältää hyvin yksinkertaisen toistorakenteen, jonka avulla voidaan toistaa joukko operaatioita kaikille tietyllä XPath-lausekkeella valituille solmuille. Toisto toteutaan elementillä
<xsl:for-each select="XPath-exp">
Elementissä oleva XPath-lauseke (select:ssä) määrittää solmujoukon ja elementin sisällä oleva XSLT-koodi toistetaan kaikille säännön valitsemille elementeille. Toistojen määrä riippuu siis aina sisään luettavasta dokumentista XPath lausekkeella valittujen solmujen lukumäärästä.

ESIM: Kaikkien 'name' elementtien attribuuttien arvoista muodostetaan lista email-osoitteita:

```
<xsl:for-each select="name">
   <xsl:value-of select="@first"/>.
   <xsl:value-of select="@last"/>@aalto.fi<br/>
</xsl:for-each>
```

8. Muita hyödyllisiä xsl:elementtejä ja vinkkejä

Muita hyödyllisiä käsittelysääntöjä ovat mm. seuraavat (Katso tarkemmin XML_XSLT_extra.pdf)

SORT

<xsl:sort select="..."> XSLT tarjoaa mahdollisuuden järjestää sisään luettavan dokumentin elementit uuteen järjestykseen ennen sapluunan valintaa

IF ja CHOOSE

XSLT:ssa on kaksi rakennetta ehdollisten osioiden laatimiseksi, joissa molemmissa käytetään samaa testirakennetta, jossa testinä on XPath-lauseke. If-lauseke on yksinkertaisempi ehdoista. Sen sisällä oleva mallinteen osa suoritetaan ehdon toteutuessa. ESIM:

```
<xsl:if test="@target &gt; 10">
  <xsl:attribute name="target">
    <xsl:value-of select="@target"/>
  </xsl:attribute>
</xsl:if>
```

TEXT

Tekstisisältö kannattaa usein syöttää *<xsl:text>elementin sisällä</xsl:text>*, jolloin varmistetaan, että syöte on varmasti string-muodossa ja ettei syötteeseen tule mukaan mitään ylimääräisiä merkkejä kuten välilyöntejä tai rivin vaihtoja:

```
 <xsl:attribute name="email">
   <xsl:value-of select="@firstname"/>
   <xsl:text>.</xsl:text>
   <xsl:value-of select="@lastname"/>
   <xsl:text>@aalto.fi</xsl:text>
 </xsl:attribute
```

MUUTTUJAT ja CONCAT XPath funktio

Edellä olevan email-attribuutin arvon voisi syöttää myös seuraavasti käyttäen concat funktiota, joka yhdistää erilliset merkkijonot (concatenate). Attribuutin nimi 'Address' ja domain nimi '@aalto.fi' on määritelty dokumentin alussa muuttujien avulla (xsl:variable). Muuttujiin viitataan XPath lausekkeessa *$muuttuja* ja muualla *{$muuttuja}*

```
<xsl:variable name="email">Address</xsl:variable>
<xsl:variable name="domain">@aalto.fi</xsl:variable>
…
 <xsl:attribute name="{$email}">
   <xsl:value-of select="concat(@firstname,'.', @lastname,$domain)"/>
 </xsl:attribute
```

9. LIITTEET: XSLT muunnosesimerkit

Tutki seuraavia kolmea XSLT-muunnosesimerkkiä ja yritä ymmärtää miten niissä XSLT muunnos (kuva B) tuottaa tulosdokumentin (kuva C) lähtödokumentista (kuva A). Sinun tulisi ymmärtää jokaisen niissä olevan xsl: alkuisen elementin/rivin merkitys muunnoksen toteutuksessa. Jos jonkin xsl-elementtin merkitys on epäselvä, tarkista se *XML_XPath_extra.pdf* kalvosetistä.

XSLT MUUNNOS ESIMERKKI 1: HTML-sivun generointi

```
<?xml version="1.0"?>
<!-- ihmeet_lec2ex1.xml -->
<?xml-stylesheet type="text/xsl" href="ihmeet_lec2ex1.xsl"?>

<ancient_wonders>
  <wonder>
    <name language="English">Colossus of Rhodes</name>
    <location>Greece</location>
  </wonder>
</ancient_wonders>
```

A) Muunnettava lähtödokumentti

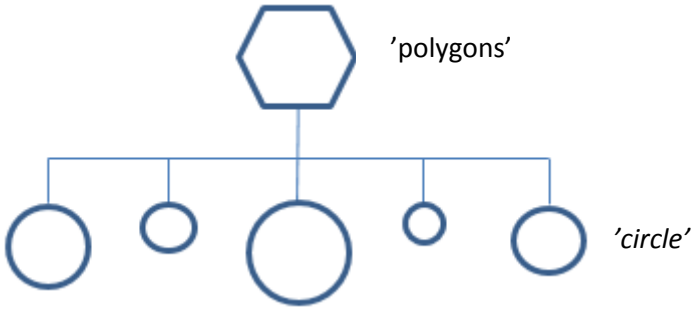
```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
    <xsl:template match="/">
        <html><head><title>Wonders of the World</title></head>
         <body>
                <h1>Wonders of the World</h1>
         The <xsl:value-of select="ancient_wonders/wonder/name"/>
         is located in <xsl:value-of select="ancient_wonders/wonder/location"/>.
        </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

B) XSLT-dokumentti, joka generoi XML-dokumentista html-dokumentin

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Wonders of the World</title>
  </head>
  <body>
    <h1>Wonders of the World</h1>
    The Colossus of Rhodes is located in Greece.
  </body>
</html>
```

C) Muunnoksen tuloksena generoitu html-dokumentti

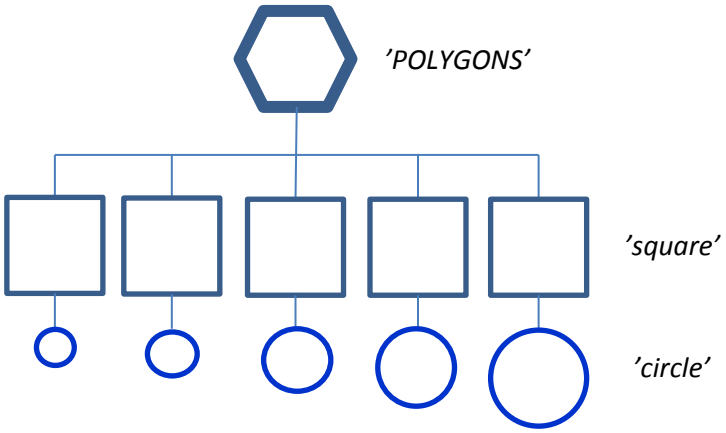
XSLT MUUNNOS ESIMERKKI 2: Uuden XML rakenteen generointi (yksi template)



A) Muunnettavan lähtödokumentin polygons_doc1.xml havainnollistava rakennekuva

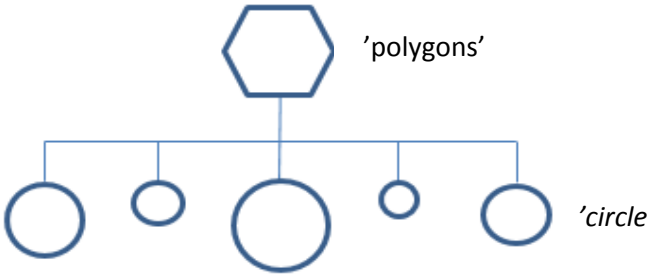
```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- XSL Ex1: to_square_sort_value-of.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="xml" indent="yes"/>
    <xsl:template match="/">
        <xsl:element name="POLYGONS">
            <xsl:for-each select="/polygons/circle">
                <xsl:sort select="@size" data-type="number" order="ascending"/>
                <xsl:element name="square">
                    <xsl:copy-of select="."/>
                </xsl:element>
            </xsl:for-each>
        </xsl:element>
    </xsl:template>
</xsl:stylesheet>
```

B) XSLT-dokumentti, joka generoi XML-dokumentista uuden XML-dokumentin



C) Muunnoksen tuloksena generoidun tulosdokumentin POLYGONS_doc2.xml rakennekuva

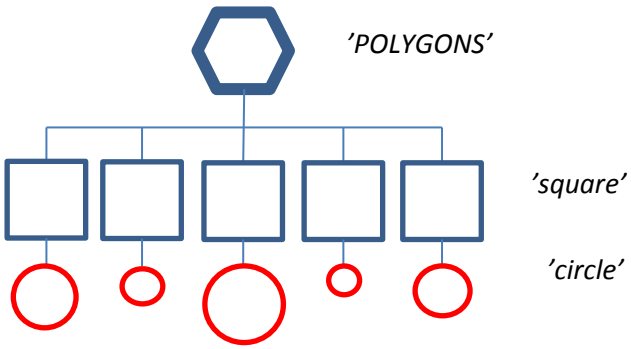
XSLT MUUNNOS ESIMERKKI 3: Uuden XML rakenteen generointi (kolme templatea)



A) Muunnettavan lähtödokumentin polygons_doc1.xml havainnollistava rakennekuva

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- to_square_simple_template.xsl  -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="xml" indent="yes"/>
    <xsl:template match="/">
        <xsl:apply-templates select="polygons"/>
    </xsl:template>
    <xsl:template match="polygons">
        <xsl:element name="POLYGONS">
            <xsl:apply-templates select="/polygons/circle"/>
        </xsl:element>
    </xsl:template>
    <xsl:template match="circle">
        <xsl:element name="square">
            <xsl:copy>
            <xsl:attribute name="color">red</xsl:attribute>
            <xsl:attribute name="size"><xsl:value-of select="@size"/>
            </xsl:attribute>
            </xsl:copy>
        </xsl:element>
    </xsl:template>
</xsl:stylesheet>
```

B) XSLT-dokumentti, joka generoi XML-dokumentista uuden XML-dokumentin



C) Muunnoksen tuloksena generoidun tulosdokumentin POLYGONS_doc2_B.xml rakennekuva