FACULTEIT INGENIEURSWETENSCHAPPEN
EN ARCHITECTUUR

Gedistribueerde toepassingen

Bert De Saffel

Master in de Industriële Wetenschappen: Informatica Academiejaar 2018–2019

UNIVERSITEIT
GENT

Inhoudsopgave

Deel I

Theorie

Hoofdstuk 1

Extensible Stylesheet Language Family

1.1 Inleiding

XSL staat voor *Extensible Stylesheet Language Family* en is een standaard om XML-documenten te presenteren en te transformeren. De drie componenten van XSL zijn:

- **XSLT** (Extensible Stylesheet Language Transformations) : dit is een XML-taal dat XML-documenten kan omvormen naar opnieuw XML, maar kan ook HTML, LaTeX, JSON, enz... zijn.
- **XPath** : dit is een taal dat waarmee bepaalde stukken van een XML-document kunnen gemanipuleerd worden aan de hand van het opbouwen van paden.
- XSL-FO (XSL Formatting Objects)

1.2 XSLT

1.2.1 Structuur

Een XSLT-bestand moet altijd voldoen aan volgende structuur:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:output method="html"/>
        <xsl:template match="patroon">
        ...
        </xsl:template>
        ...
</xsl:stylesheet>
```

1. `xsl:stylesheet` is het basiselement

2. `xsl:output` is het element dat het type transformatie vastlegt in het attribuut *method*
3. `xsl:template` bevat wat er moet gebeuren voor elk element dat aan *patroon* voldoet.

1.2.2 Instructie-elementen

- **xsl:value-of** : Bepaalt de waarde van een XPath uitdrukking.

  ```
  <xsl:value-of select='uitdrukking'/>
  ```

- **apply-templates** : Zal de templates van de onmiddelijke kindelementen van het huidig element toepassen. Indien er geen templates zijn, zal de inhoud van het element naar de uitvoer uitgeschreven worden. Het optionele *select* attribuut zal enkel de templates uitvoeren voor elk element die aan de uitdrukking voldoet.

  ```
  <xsl:apply-templates [select='uitdrukking']/>
  ```

- **xsl:for-each** : Voert een actie uit voor alle knopen die door het *select*-attribuut bepaalt wordt. Binnen de for-each kunnen meerdere instructie-elementen voorkomen.

  ```
  <xsl:for-each select'uitdukking'>...</xsl:for-each>
  ```

- **xsl:sort** :Dit is een kindelement van *xsl:apply-templates* of een *xsl:for-each* element. Dit element zal de knopen sorteren die aan de waarde van het select attribuut voldoen. Optionele attributen zijn

  - data-type : legt het datatype vast (*text* of *number*). Dit heeft een invloed op de sorteervolgorde (bij *text* is 10 kleiner dan 2).
  - order : bepaalt de sorteervolgorde (*ascending* of *descending*).

  ```
  <xsl:sort select='uitdrukking' [data-type='text|number'] [order='ascending|descending]/>
  ```

- **xsl:if** : Dit is een typische selectiestructuur. Het bevat als enig attribuut *test* met als waarde een predicaat.

  ```
  <xsl:if test='logische uitdrukking'> ...  </xsl:if>
  ```

- **xsl:choose** : Een alternatieve selectiestructuur equivalent zoals een switch in programeertalen zoals Java en C#. Dit element wordt gevolgd door één of meerdere **xsl:when** elementen. Een xsl:when element is equivalent met een xsl:if element en heeft ook als enig attribuut *test*.

  ```
  <xsl:choose>
          <xsl:when test="logische uitdrukking 1">
                  ...
          </xsl:when>
          <xsl:when test="logische uitdrukking 2">
                  ...
          </xsl:when>
          ...
  </xsl:choose>
  ```

- **xsl:text** : Wanneer tekst spaties bevat is het aan te raden om xsl:text te gebruiken zodat deze ook opgenomen worden in het resultaat. Spaties die niet in een xsl:text element staan worden genegeerd.

  ```
  <xsl:text>stukje tekst</xsl:text>
  ```

- **xsl:variable** : Legt een constante variabele vast. Het verplichte attribuut is *name*, wat de naam van de variabele vastlegt. Het invullen van deze waarden kan op twee manieren :

  1. via het select attribuut `<xsl:variabele name='varnaam' select='uitdrukking'/>`
  2. via de inhoud van het element: `<xsl:variabele name ="varnaam"> ... </xsl:variable>` In dit geval kan de waarde ook opgebouwd worden uit meerdere instructies.

- **xsl:param** : Indien dit element gedeclareerd wordt in het begin van een xls:template element, dan is het mogelijk om parameters mee te geven aan dit template. De declaratie kan op dezelfde manier gebeuren als bij xsl:variabele.

- **xsl:with-param** : Dit element is een kindobject van xsl:apply-templates. Dit element vult voor een bepaalde *name* de waarde die in *select* staat.

```
<xsl:apply-templates ...>
        <xsl:with-param name='varnaam' select='uitdrukking'>
</xsl:apply-templates>
```

Hoofdstuk 2

XML Path Language (XPath)

XPath is een taal om knopen te selecteren van een XML-document. Het resultaat van een XPath-uitdrukking kan een getal, string of logische waarde zijn.

2.1 Paden

Belangrijke XPath uitdrukkingen zijn paden. Een pad identificeert nul, één of meerdere knopen in een XML-document. Een pad bestaat uit *location steps* die verbonden zijn met een /. Bij elk pad zijn volgende wildcards mogelijk:

- *: selecteert elk elementknoop in de huidige context.
- node(): selecteert alle knopen, dus niet enkel elementen
- @*: selecteert alle attributen in de huidige context.

2.1.1 Basispad (Root Location Path)

Het eenvoudigste pad is `/` en selecteert het root-element van het XML-document. Verder is dit een absoluut pad, dus deze uitdrukking zal **altijd** het root-element teruggeven.

2.1.2 Kinelement-stap (Child Element Location Steps)

Dit pad bestaat uit de naam van een element en selecteert alle elementen met deze naam in de huidige context. De context is afhankelijk van de ouder.

```
<xsl:template match ="author">
        <xls:apply-templates select="name" />
</xls:template>
```

In dit voorbeeld wordt de XPath uitdrukking van het selectattribuut `/author/name/`.

2.1.3 Attribuut-stap (Attribute Location Steps)

Een attribuut selecteren begint met een @ en wordt gevolgd door de naam van het attribuut.

```
<xsl:template match ="query">
        <xls:value-of select="@isbn"/>
</xls:template>
```

2.1.4 Voorwaarden

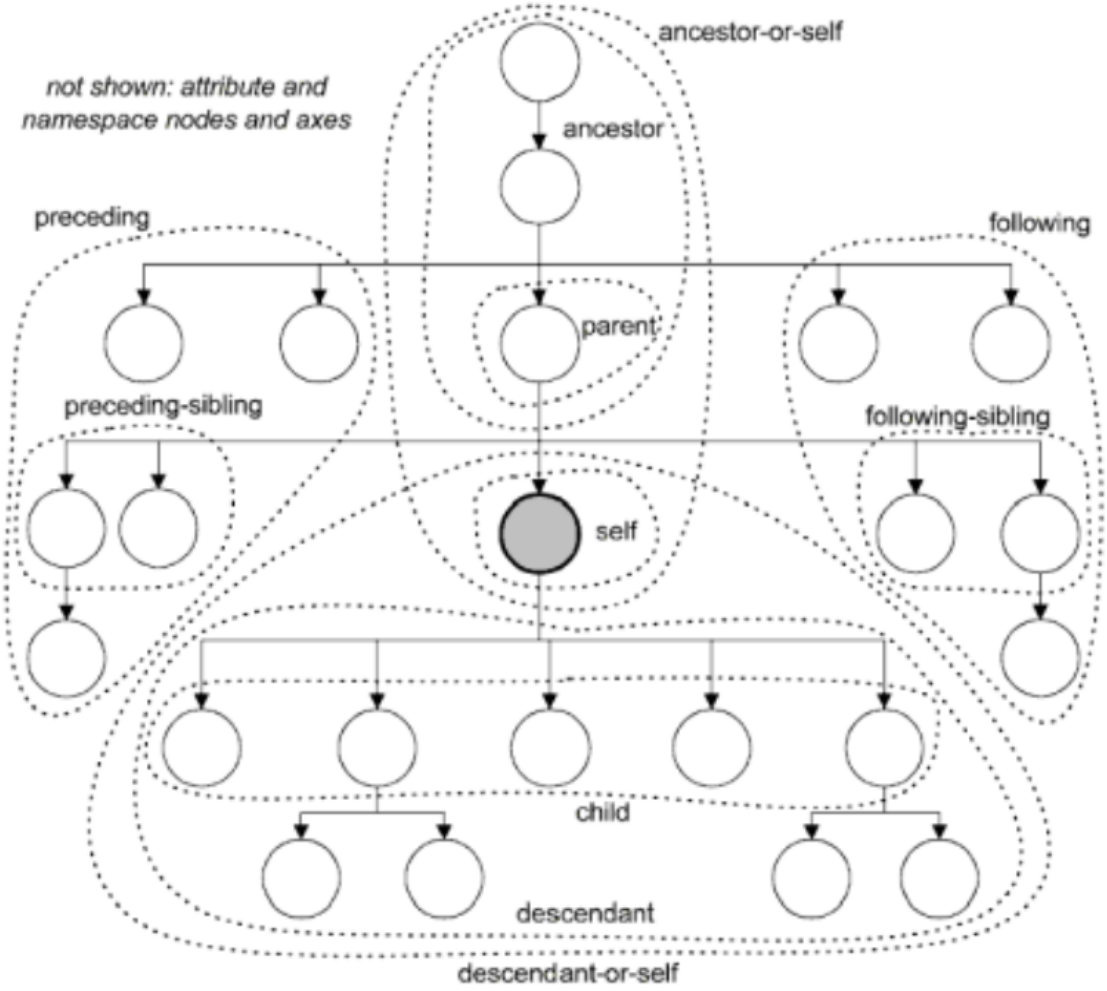
```
<xsl:apply-templates select="//book[title='XML']//name[.='Ongenae]"/>
```

```
<xsl:apply-templates select="//person[@born<=1976]"/>
```

2.1.5 Lange padnamen

In elke XPath uitdrukking wordt elke stap voorafgegaan door de child:: as.
`books/book/isbn => child::books/child::book/child::isbn`
andere assen:

- parent (of ..)
- self (of .)
- descendant-or-self (of //)
- ancestor
- ancestor-or-self
- namespace
- descendant
- following-sibling
- preceding-sibling
- following
- preceding



2.2 Andere XPath-uitdrukkingen

XPath kan ook getallen, logische waarden en strings manipuleren.

2.2.1 Datatypes

- **Getallen** : Stel dat *born* het geboortejaar van een persoon is, dan zal de volgende uitdrukking de eeuw uitschrijven van dit jaar. `<xls:value of select="(@born - (@born mod 100)) div 100 + 1"`
- **Strings** : Strings staan tussen enkele of dubbele aanhalingstekens. Strings vergelijken kan met = en != operatoren.
- **Logische waarden** : De sleutelwoorden true en false bestaan niet in XPath, wel worden ze

respectievelijk voorgesteld door de functies true() en false(). Op logische waarden kunnen de operatoren *and* en *or* toegepast worden. Met not(...) wordt de negatie toegepast.

2.3 Functies

XPath kent een aantal functies.

- **Knopenfuncties** zoals *position()*, *last()*, *count(...)* en *id(...)* bepalen respectievelijk
  - het volgnummer van de huidige knoop in de context,
  - het aantal knopen in de huidige context (volgnummer van de laatste knoop),
  - telt het aantal knopen van het argument, dat een verzameling knopen is en
  - bepaalt een verzameling knopen. Deze verzameling bevat alle elementen van een XML-document met één van de gespecifieerde ID's. Het argument is een string bestaande uit ID's gescheiden door spaties.
- **Stringfuncties** zoals *concat(...)*, *contains(...)*, *starts-with(...)*, *string(...)* (conversie naar string) en *string-length(...)*.
- **Logische functies** zoals *true()*, *false()*, *not(...)*, *boolean(...)* (conversie naar bool waarde).
- **Numerieke functies** zoals *ceiling(...)*, *floor(...)*, *number(...)*, *round(...)*.