Processamento Estruturado de Documentos

LMCC & LESI, Universidade do Minho

Ano lectivo 2000/2001

Ficha Teórico-Prática Nº4

José Carlos Ramalho

7 de Novembro de 2000

1 Validação de Documentos usando o Omnimark

1.1 *Crie uma script Omnimark que permita se um documento XML está de acordo com um determinado DTD.*

Faça o download do poema e respectivo DTD da página da cadeira e utilize a script para fazer a respectiva validação. Introduza vários erros e analise as mensagens de erro.

```
;------------------------------------------------------------
; parser.xom
;
; Script que valida um documento XML com o respectivo DTD
;------------------------------------------------------------
down-translate with xml

element #implied
  suppress
```

2 Make-skeleton para Omnimark

2.1 *Crie uma script Omnimark de nome makeskel que recebendo um documento XML (com indicação dum DTD) vai criar uma nova script para transformar documentos desse tipo em documentos doutro formato.*

```
;------------------------------------------------------------------
; makeskel.xom
;------------------------------------------------------------------
```

```
GLOBAL COUNTER d1
GLOBAL STREAM CurrentFile
GLOBAL COUNTER df1 variable INITIAL-SIZE 0
GLOBAL COUNTER df2 variable INITIAL-SIZE 0
GLOBAL STREAM day
GLOBAL STREAM month
GLOBAL STREAM year
GLOBAL SWITCH light

process-start
  submit file #command-line-names
  set CurrentFile to #command-line-names
  set year to DATE "=xY"
  set month to DATE "=M"
  set day to DATE "=D"
  output "; this Omnimark script was generated: " ||
         "%g(day)-%g(month)-%g(year)%n;     by makeskel.xom%n%n" ||
         "down-translate with xml%n"

; handle comments
find "<!--" ((lookahead not "-->") any)* "-->"

; process empty elements
find "<" white-space* "!" white-space* ul "element"
      white-space* ul (letter ((lookahead not ul white-space) any)+) => dgn1
      white-space+ ul ("E" white-space* "M" white-space* "P"
        white-space* "T" white-space* "Y")
    do when df2 has key "%x(dgn1)"
      increment df2 key "%x(dgn1)"
    else
      set new df2 key "%x(dgn1)" to 1
    done

;handle elements that have content
find "<" white-space* "!" white-space* ul "element"
      white-space* ul (letter ((lookahead not ul white-space) any)+) => dgn1
      white-space+
    do when df1 has key "%x(dgn1)"
      increment df1 key "%x(dgn1)"
    else
      set new df1 key "%x(dgn1)" to 1
    done
```

```
find any

process-end
  repeat over df1
    do when df1 > 0
      local stream narf
      set narf to key of df1
        output "%nelement %g(narf)%n" ||
               "  output %"%%c%"%n%n"
    done
  again
  repeat over df2
    do when df2 > 0
      local stream narf
      set narf to key of df2
        output "%nelement %g(narf)%n" ||
               "  output %"%%c%"%n%n"
    done
  again
  output "element %#implied%n  suppress%n"
```

2.2 *Use a script makeskel.xom para gerar uma script de transformação para o poema. Altere o esqueleto gerado para tranformar poemas no formato XML em HTML.*

2.3 *Aplique agora o mesmo método ao DTD do relatório para gerar o esqueleto duma script de tranformação para LaTeX.*

3 XPath e Linguagens de Query

3.1 *Utilize o xw ("XML Workbench") para realizar as seguintes tarefas:*

- *Carregue em memória o poema e todos os outros documentos XML que tiver na sua directoria de trabalho.*
- *Execute a seguinte query: "Todos os elementos que tenham pelo menos um atributo instanciado".*
- *Execute a seguinte query: "Todos os elementos que são netos de alguém".*
- *Execute a seguinte query: "Todos os elementos na descendência de quadra".*
- *Execute a seguinte query: "Todos os elementos com um filho p".*

- *Execute a seguinte query: "Todos os atributos".*
- *Execute a seguinte query: "Todos os atributos de nome ident".*
- *Execute a seguinte query: "Todos os atributos pertencentes a elementos que ocorram do segundo nível para baixo na árvore documental".*