Y36XML – Technologie XML

Přednáší:

Irena Mlýnková (mlynkova@ksi.mff.cuni.cz)

Martin Nečaský (necasky@ksi.mff.cuni.cz)

ZS 2009

Stránka přednášky:
http://www.ksi.mff.cuni.cz/~mlynkova/Y36XML/

Osnova předmětu

- Úvod do principů formátu XML, přehled XML technologií, jazyk DTD
- Datové modely XML, rozhraní DOM a SAX
- Úvod do jazyka XPath
- Úvod do jazyka XSLT
- XPath 2.0, XSLT 2.0
- Úvod do jazyka XML Schema
- Pokročilé rysy jazyka XML Schema
- Přehled standardních XML formátů
- Úvod do jazyka XQuery
- Pokročilé rysy jazyka XQuery, XQuery Update
- Úvod do XML databází, nativní XML databáze, číslovací schémata, structural join
- Relační databáze s XML rozšířením, SQL/XML

Jmenné prostory

- xml =
  http://www.w3.org/XML/1998/namespace
- xs =
  http://www.w3.org/2001/XMLSchema
- xsi =
  http://www.w3.org/2001/XMLSchema-instance
- fn =
  http://www.w3.org/2005/04/xpath-functions
- xdt =
  http://www.w3.org/2005/04/xpath-datatypes
- local =
  http://www.w3.org/2005/04/xquery-local-functions

Jmenné prostory - speciální

- specifikace XQuery využívají speciální jmenné prostory
  - *dm* = přístup přes datový model
  - *op* = pro operátory XQuery
  - *fs* = pro funkce a typy definované ve formální sémantice XQuery
  - nemají explicitní URI
  - konstrukce z těchto jmenný prostorů nejsou přístupné v XPath/XQuery/XSLT

Datový model XQuery

- XQuery 1.0 and XPath 2.0 Data Model
- definuje informaci obsaženou ve vstupu pro XSLT či XQuery procesor
- definuje všechny povolené hodnoty výrazů v jazycích XSLT, XQuery a XPath

Datový model XQuery

- jazyk je uzavřený vzhledem k datovému modelu, pokud hodnota každého jeho výrazu je v tomto datovém modelu
- XPath, XSLT a XQuery jsou uzavřené vzhledem k XQuery 1.0 and XPath 2.0 Data Model

Datový model XQuery

- založený na datovém modelu XML (XML Infoset)
- vyžaduje další vlastnosti vycházející z požadavků na sílu jazyků XQuery a XSLT
  - nereprezentujeme jen XML dokumenty ale i výsledky dotazů
  - podpora typovaných atomických hodnot a uzlů
- typy založeny na XML Schema
  - podpora uspořádaných sekvencí
    - atomických hodnot
    - smíšených, tj. složených jak z uzlů (i dokumentů) tak z atomických hodnot

Datový model XQuery

- sekvence je uspořádaná kolekce položek
  - nemůže obsahovat jinou sekvenci
- položka je buď uzel a nebo atomická hodnota
  - může existovat pouze v sekvenci
  - v jedné sekvenci může být vícekrát
  - musí mít přiřazen typ

➔ jazyk založený na datovém modelu XQuery je silně typovaný (!)

➔ výsledkem dotazu je sekvence

Datový model XQuery
Atomické hodnoty

- atomická hodnota je hodnotou z domény atomického typu a má přiřazen název tohoto typu
- atomický typ je
  - primitivní jednoduchý typ
  - odvozený z jiného atomického typu restrikcí (viz. XML Schema)

Datový model XQuery
Atomické hodnoty

- jednoduché datové typy
  - 19 XML Schema vestavěných datových typů
  - xs:untypedAtomic
    - označuje, že atomická hodnota nemá žádný typ
  - xs:anyAtomicType
    - atomická hodnota má přiřazen nějaký atomický typ, ale není určeno jaký
    - zahrnuje všechny atomické typy
  - xs:dayTimeDuration
  - xs:yearMonthDuration

Datový model XQuery
Uzly

- 7 typů uzlů
- `document`, `element`, `attribute`, `text`, `namespace`, `processing instruction`, a `comment`
  - méně než v XML Infoset
  - nereprezentuje např. DTD a notace
- každý uzel má svoji jednoznačnou identitu
- každý uzel má svůj typ obsahu
  - odvozování typů obsahu převzato z XML Schema
  - xs:untyped označuje, že uzel nemá žádný typ

Datový model XQuery
Uzly

- přístup k hodnotě uzlu přetypované na `xs:string`
  - `fn:string`
- přístup k hodnotě uzlu s původním datovým typem
  - `fn:data`

Datový model XQuery
Výsledek dotazu

- výsledkem dotazu v dotazovacím jazyce založeném na datovém modelu XQuery je instance tohoto modelu
- instancí může být pouze sekvence
  - položky (tj. atomické hodnoty nebo uzly) mohou existovat pouze v sekvencích
- pokud je položkou uzel, pak je to kořen stromu
  - pokud je to `document` tak strom reprezentuje XML dokument
  - pokud to není `document` tak strom reprezentuje fragment XML dokumentu

Datový model XQuery
Příklad – XML data

```
<?xml version="1.0"?>
<katalog>
  <kniha rok=”2002”>
    <titul>Šéfkuchař bez čepice</titul>
    <autor>Jamie Oliver</autor>
    <isbn>80-968347-4-6</isbn>
    <kategorie>kuchařky</kategorie>
    <stran>250</stran>
    <recenze>
      Během posledních sedmi let se stal <autor>Jamie
   Oliver</autor> díky svým zábavným televizním pořadům o vaření
   <titul>Šéfkuchař bez čepice</titul> pro BBC či <titul>Jamieho
   kuchyně</titul> pro Channel 4 jednou z nejoblíbenějších
   celebrit v Británii.
    </recenze>
  </kniha>
```

Datový model XQuery
Příklad – XML data

```
  <kniha rok="2007">
    <titul>Modrá, nikoli zelená planeta</titul>
    <podtitul>Co je ohroženo: klima nebo svoboda?</podtitul>
    <autor>Václav Klaus</autor>
    <isbn>978-80-7363-152-9</isbn>
    <kategorie>společnost</kategorie>
    <kategorie>ekologie</kategorie>
    <stran>176</stran>
    <recenze>
      Oproti <titul>Nepříjemná pravda</titul> jejímž autorem je
   <autor>Al Gore</autor> ... .
    </recenze>
  </kniha>
</katalog>
```

Datový model XQuery
Příklad – reprezentace (z Infoset)

```
element catalog of type xs:untyped {
  element kniha of type xs:untyped {
    attribute rok of type xs:untypedAtomic {“2002”},
    element titul of type xs:untyped {
      text of type xs:untypedAtomic {“Šéfkuchar bez čepice”}
    },
    element autor of type xs:untyped {
      text of type xs:untypedAtomic {“Jamie Oliver”}
    },
    ...
```

Datový model XQuery
Příklad – reprezentace (z Infoset)

```
    element recenze of type xs:untyped {
      text of type xs:untypedAtomic {
        "Během posledních sedmi let se stal "
      },
      element autor of type xs:untyped {
        text  of type xs:untypedAtomic {"Jamie Oliver"}
      },
      text of type xs:untypedAtomic {
        " díky svým zábavným televizním pořadům o vaření "
      },
      ...
    },
  },
  ...
}
```

Datový model XQuery
Příklad – schéma XML dat

```
<?xml version="1.0"?>
<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”
  <xs:element name=”katalog” type=”KatalogType” />
  <xs:complexType name=”KatalogType”
    <xs:sequence>
      <xs:element name=”kniha” type=”KnihaType”
                  maxOccurs=”unbounded”/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name=”KnihaType”>
    <xs:sequence>
      <xs:element name=”titul” type=”xs:string” />
      <xs:element name=”autor” type=”xs:string” />
      ...
      <xs:element name=”recenze” type=”RecenzeType” />
    </xs:sequence>
    <xs:attribute name=”rok” type=”gYear” />
  </xs:complexType>
```

Datový model XQuery
Příklad – schéma XML dat

```
  <xs:complexType name="RecenzeType" mixed="true">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="titul" type="xs:string" />
      <xs:element name="autor" type="xs:string" />
    </xs:choice>
  </xs:complexType>
</xs:schema>
```

- XML dokumentu přiřadíme toto XML schéma v rámci jmenného prostoru bk

Datový model XQuery
Příklad – reprezentace (z PSVI)

```
element catalog of type bk:KatalogType {
  element kniha of type bk:KnihaType {
    attribute rok of type xs:gYear {“2002”},
    element titul of type xs:string {
      text of type xs:untypedAtomic {“Šéfkuchar bez čepice”}
    },
    element autor of type xs:string {
      text of type xs:untypedAtomic {“Jamie Oliver”}
    },
    ...
```

Datový model XQuery
Příklad – reprezentace (z PSVI)

```
    element recenze of type bk:RecenzeType {
      text of type xs:untypedAtomic {
        "Během posledních sedmi let se stal "
      },
      element autor of type xs:string {
        text  of type xs:untypedAtomic {"Jamie Oliver"}
      },
      text of type xs:untypedAtomic {
        " díky svým zábavným televizním pořadům o vaření "
      },
      ...
    },
  },
  ...
}
```

Typová propagace

- při
  - funkčních voláních
  - v klauzulých order by
  - u operátorů s numerickými nebo řetězcovými operandy
- numerická typová propagase
  - `xs:float -> xs:double` (stejná hodnota)
  - `xs:decimal -> xs:float -> xs:double` (ztráta přesnosti)
- řetězcová typová propagace
  - `xs:anyURI -> xs:string`

Konverze typů ve funkcích

- argument je atomický typ
  - vstupní parametr (tj. posloupnost) atomizován
- položka typu `xs:untypedAtomic` ve vstupní posloupnosti je přetypována na odpovídající typ argumentu

Typové operátory

- *expr* `instance of` *sequenceType*
  - vrací `true`, pokud hodnota *expr* má typ *sequenceType* (uvažuje se propagace typů)

```
1 instance of xs:integer            => true
1 instance of xs:decimal            => true
(1,2) instance of xs:integer        => false
(1,2) instance of xs:integer+       => true
<x/> instance of element ()         => true
<x/> instance of xs:string          => false
<x>{5}</x> instance of xs:integer => false
<x>{5}</x> instance of element()  => true
```

Typové operátory

- `typeswitch`
  - umožňuje výběr možnosti podle typu argumentu

```
typeswitch($a)
  case $a as element(*, USAddress)
          return $a/state
  case $a as element(*, CanadaAddress)
          return $a/province
  case $a as element(*, JapanAddress)
          return $a/prefecture
  default return "unknown"
```

Typové operátory

- `cast`
  - umožňuje přetypování
- `castable`
  - true, pokud je přetypování možné

```
if ($x castable as hatsize)
   then $x cast as hatsize
   else if ($x castable as IQ)
   then $x cast as IQ
   else $x cast as xs:string
```

Typové operátory

- `treat`
  - přetypování proti směru typové hierarchie (down-casting)
  - cílový typ musí být odvozen od původního

`1.0 treat as xs:integer => 1.0` (xs:decimal)

`1.8 treat as xs:integer =>` chyba při běhu (není celé číslo)

`"1" treat as xs:integer =>` chyba při překladu

XQuery Algebra

- “databázová” algebra (hnízděná relační)
  - zpracovávání n-tic
  - zaměřená na operace spojení, seskupení, uspořádání
  - práce nad indexy
  - částečná paralelizace (pipelining)
- streamované operace
  - zpracování událostí
  - efektivita nad soubory a sockety
  - přímá podpora paralelizace

XQuery Optimalizace

- “relační” optimalizace
  - selekce co nejdříve
  - eliminace hnízděných bloků dotazu
  - paralelizace spojení n-tic
  - ...
- optimalizace funkcionálního jazyka
  - odstranění rekurze
  - vkládání funkcí
  - optimalizace cyklů
  - ...

XQuery Optimalizace - Problémy

- podmíněné výrazy a zpracování chyb

```
if ($k/rok-vydani = 2004)
then ns:WS(<novinka>{$xx/titul}</novinka>)
else ns:WS(<titul>{$xx/titul}</titul>)
```

- problém
  - pouze jedna větev smí vyvolat vyjímku
  - omezená možnost paralelizmu a přeplánování

XQuery Optimalizace - Problémy

- zpracování chyb

```
for $x in (1 to 10)
return if $x = 100 then ($z idiv 0) else $x
vs.
let $y := $z idiv 0
for $x in (1 to 10)
return if $x = 100 then $y else $x
```

- problém
  - možné pouze při striktním “líném” vyhodnocování

XQuery Optimalizace - Problémy

- neplatná pravidla některých operátorů
  - porovnání nejsou tranzitivní
    - implicitní existenciální kvantifikace
    - dynamická přetypování
  - neplatí klasické booleaovské zákony (distributivita, ...)
    - `fn:not($x = $y)` jiné než `$x != $y`

XQuery Optimalizace - Problémy

- dvouhodnotová logika
  - `()` je přeloženo na `fn:false()`
- nedeterminisimus
  - platí zkrácené vyhodnocování logických výrazů
  - důsledek: program může dávat více různých výsledků v závislosti na vyhodnocení a všechny jsou platné

```
if (($x castable as xs:string) and ($x cast as
   xs:string = “xxx”)) then ...
```

XQuery – streamované XML

- problém se zpětnou navigací
  - řešeno pouze částečně
  - náhrada dopřednou navigací pokud lze
  - jinak ukládání na zásobník
    - budujeme vlastně DOM struktury

XQuery – materializace dat

- maximalizace využití operací nad datovým proudem
- materializace
  - pro relační operátory
  - pokud je proměnná použita vícekrát
  - pokud je proměnná použita v cyklu
  - pokud je obsah proměnné předáván rekurzivní funkci
  - při zpětné navigaci

XQuery – materializace dat

- datový model vyžaduje u každého uzlu jednoznačný identifikátor
  - náročné na prostor i čas
- identifikátory uzlů jsou generovány jen pro operace, které to skutečně vyžadují
  - relační operátory, navigace, id(), pořadí, ...
  - není třeba např. pro prostou serializaci
- překladač musí podporovat poměrně silné prostředky pro analýzu datového toku

XQuery – příklady použití

- datová integrace
  - kratší ale složité dotazy
  - velká datová báze, perzistentní
  - zachovává hierarchii
- centralizovaná XML data
  - pouze ke čtení
  - velmi velké objemy
  - selektivní dotazy
  - široce indexovaná data

XQuery – příklady použití

- distribuovaná data
  - XML zdroje na webu
- XML pro webové služby a prezentaci
  - role transormačního prostředku
  - dlouhé a složité dotazy, vícenásobně používané
  - streamovaná data
- viz. případy použití na W3C
  http://www.w3.org/TR/xquery-use-cases/

Konec