INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu *Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP* reg. č. CZ.1.07/2.2.00/29.0011

Pojmenované vzory

Pojmenované vzory (1)

- nezbytné pro složitější jazyky – opakované použití
- **XML syntaxe:**
  - <define name="*jméno*">*obsah*</define>
  - použití <ref name="*jméno*"/>
  - XML vyžaduje jeden kořenový prvek – schéma zabaleno do <grammar>...</grammar>
  - <grammar> obsahuje právě jeden prvek <start> určující výchozí bod schématu

Příklad – XML

```
<grammar>

<start>
    <element name="osoba">
        <ref name="jmeno"/>
        <element name="adresa"><text/></element>
    </element>
</start>

<define name="jmeno">
    <element name="krestni"><text/></element>
    <element name="prijmeni"><text/></element>
</define>

</grammar>
```

Pojmenované vzory (2)

- **kompaktní syntaxe:**
  - definice: *jméno = obsah*
  - při použití se jednoduše uvede *jméno*
  - není XML, přesto je i zde formálně definován obalový grammar { ... }, je ale implicitní – nemusí se uvádět
  - uvnitř grammar je právě jedna definice start = *obsah* určující výchozí bod schématu

Příklad – kompaktní

```
grammar {

start =
    element osoba {
        jmeno,
        element adresa { text }
    }

jmeno =
    element krestni { text },
    element prijmeni { text }

}
```

Definice jazyka

Obvyklé návrhy schémat

- **matrjoška**
  - přímé vkládání vzorů, jen pro jednoduché jazyky
- **à la DTD**
  - samostatná definice pro každý prvek, určující jeho obsah; odkazuje se na definice ostatních prvků
  - definice prvku snadno k nalezení
- **orientovaný na obsah**
  - definuje obsah každého prvku jako samostatný vzor
  - nejsnadněji rozšiřitelné

Ceník jako matrjoška

```
element cenik {
    element zbozi {
        element nazev { text },
        element cena { xsd:decimal }
    }+
}
```

Ceník à la DTD

```
grammar {

start = prvek-cenik

prvek-cenik = element cenik { prvek-zbozi+ }

prvek-zbozi = element zbozi {
      prvek-nazev,
      prvek-cena
      }

prvek-nazev = element nazev { text }

prvek-cena = element cena { xsd:decimal }

}
```

Ceník podle obsahů

```
grammar {

start = element cenik { cenik-obsah }

cenik-obsah = element zbozi { zbozi-obsah }+

zbozi-obsah =
    element nazev { nazev-obsah },
    element cena { cena-obsah }

nazev-obsah = text

cena-obsah = xsd:decimal

}
```

Pokročilé konstrukce

Vkládání definic

- do grammar lze vložit další z externího souboru – vloží se všechny jeho definice
  - XML: <include href="*lokátor-souboru*"/>
  - kompaktní: include "*lokátor-souboru*"
- umožňuje vytvářet knihovny „předvařených" definic

```
<grammar>
    <include href="typedef.rng"/>
    ...
</grammar>
```

```
grammar {
    include "typedef.rnc",
    ...
}
```

Úprava vložených definic

- lze změnit vybrané definice z vloženého souboru
  - XML:

```
<include href="lokátor-souboru">
    <define name="jméno">
        nová-definice
    </define>
</include>
```

  - kompaktní:

```
include "lokátor-souboru" {
    jméno = nová-definice
}
```

Odkazy na externí definice

- lze využívat vzory uložené v jiných souborech (obsahem souboru musí být jeden vzor)
  - XML: <externalRef href="*lokátor-souboru*"/>
  - kompaktní: external "*lokátor-souboru*"

```
<element name="prodejna">
    <externalRef href="cenik.rng"/>
    <element name="vedouci">
        <text/>
    </element>
</element>
```

```
element prodejna {
    external "cenik.rnc",
    element vedouci { text }
}
```

Kombinování definic

- schéma může obsahovat několik stejnojmenných definic – kombinují se
- nutno definovat způsob kombinace
  - XML: atribut combine, hodnoty choice a interleave
  - kompaktní: znaky |= a &= místo =
  - všechny definice pro stejné jméno musí mít totožný způsob kombinování
  - u jedné lze vynechat – výhodné pro úpravy knihoven

Příklad: alternativy pro jméno

```
<define name="jmeno">
    <element name="jmeno">
        <text/>
    </element>
</define>

<define name="jmeno" combine="choice">
    <element name="krestni">
        <text/>
    </element>
    <element name="prijmeni">
        <text/>
    </elemen>
</define>
```

```
jmeno =
    element jmeno { text }

jmeno |=
    element krestni { text },
    element prijmeni { text }
```

Jmenné prostory (1)

- Relax NG podporuje jmenné prostory
- XML syntaxe:
  - atribut ns="*lokátor-jmenného-prostoru*" u prvku element
  - dědí se, stačí uvést u kořenového prvku
  - atribut ns lze uvést i u prvků attribute; implicitní hodnotou je prázdný řetězec
  - jmenné prostory lze deklarovat i standardně a používat prefixy v hodnotě atributu name

Jmenné prostory (2)

- kompaktní syntaxe:
  - na začátku je třeba deklarovat prefixy jmenných prostorů
    namespace *prefix* = "*lokátor-jmenného-prostoru*"
  - deklarace se nemohou vnořovat, nepřenášejí se z vkládaných souborů
  - lze deklarovat implicitní jmenný prostor:
    default namespace = "*lokátor-jmenného-prostoru*"
    (opět se nevztahuje na atributy)
  - nemá-li vložený soubor (include, external) vlastní implicitní jmenný prostor, zdědí jej od vkládajícího

Příklad s implicitním prostorem

```
<element name="cenik"
    ns="http://www.kdesi.cz/relax/cenik">
    <oneOrMore>
        <ref name="zbozi"/>
    </oneOrMore>
</element>

<define name="zbozi">
    <element name="zbozi">
        <element name="nazev">
            <text/></element>
        <element name="cena">
            <text/></element>
    </element>
</define>
```

```
default namespace =
    "http://www.kdesi.cz/
     relax/cenik"

element cenik {
    zbozi+
}

zbozi = element zbozi {
    element nazev { text },
    element cena { text }
}
```

Obsah podle kontextu

- Relax NG umožňuje definovat strukturu dokumentu v závislosti na jeho obsahu
- hodnota prvku či atributu může ovlivnit, jaké další prvky/atributy dokument bude obsahovat
- technická realizace: výběr ze skupin, do nichž patří prvek/atribut s danou hodnotou a věci na něm závislé
- příklad: u hudby nás zajímá médium, u SW licence

Příklad závislosti na kontextu

```
<element name="cd">
    <element name="nazev">
        <text/></element>
    <choice>
        <group>
            <attribute name="druh">
                <value>hudba</value>
            </attribute>
            <element name="interpret">
                <text/></element>
        </group>
        ...
    </choice>
</element>
```

```
element cd {
    element nazev { text },
    (   attribute druh { "hudba" },
        element interpret { text } )
    |
    (   attribute druh { "sw" },
        element licence { text } )
}
```

Dokumentace (1)

- nemá speciální dokumentační prvky, lze vložit cokoli
- **XML syntaxe:**
  - prvky z jiných jmenných prostorů než http://relaxng.org/ns/structure/1.0 jsou ve schématu ignorovány
  - lze využít pro vkládání dokumentace a komentářů

```
<element name="zbozi">
    <html:p><html:strong>zbozi:</html:strong>
    jedna položka ceníku.</html:p>
    ...
</element>
```

Dokumentace (2)

- **kompaktní syntaxe:**
  - dokumentace bývá v XML
  - pro ohraničení se používají [...]
  - přesná syntaxe závisí na umístění (před/za/mimo prvek)

```
[  html:p  [  html:strong  [ zbozi  ]
jedna položka ceníku.  ]  ]
element zbozi {
    ...
}
```

Dokumentace (3)

- **kompaktní syntaxe:**
  - připouští komentáře, zahájeny znakem #
  - specifikace kompatibility s DTD zavádí prvek documentation; zkrácená syntaxe ## *text*

```
## jedna položka ceníku
element zbozi {
    ...
}
```

```
namespace a =
"http://relaxng.org/ns/compatibility/annotations/1.0"

[ a:documentation [
    jedna položka ceníku ] ]
element zbozi {
    ...
}
```