Zachycení lingvistické anotace na několika rovinách pomocí PML

Jan Štěpánek

ÚFAL

30. 10. 2006, Praha

PML – Prague Markup Language

- Požadavky na obecný datový formát
- Schéma a instance
- PML
  - datové typy
  - role
  - linky
  - modularizace

Požadavky na obecný datový formát

Požadavky na obecný datový formát

- **Uniformita**

  podobné nebo analogické konstrukce by měly být reprezentovány jednotně nebo analogicky.

Požadavky na obecný datový formát

- Uniformita
- **Principy stand-off anotace**
  - *oddělení anotace od holých dat*
  - *rozdělení anotace do několika rovin*

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- **Jednotný systém odkazů**
  - *odkazy v rámci jedné roviny*
  - *odkazy mezi rovinami*
  - *odkazy na datové zdroje a externí dokumenty*

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- **Lineární a strukturní data**
  - *lineární: pořadí slov nebo vět, časová následnost apod.*
  - *strukturní: stromy, multidominance, cykly.*

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- Lineární a strukturní data
- **Strukturované atributy**

  komplexní datové struktury odpovídající feature-structures.

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- Lineární a strukturní data
- Strukturované atributy
- **Víceznačnost**

  jednotná reprezentace alternativních anotací.

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- Lineární a strukturní data
- Strukturované atributy
- Víceznačnost
- **Čitelnost pro člověka**

  poškození dat, vývojové fáze, snazší rozvoj a přizpůsobení.

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- Lineární a strukturní data
- Strukturované atributy
- Víceznačnost
- Čitelnost pro člověka
- **Rozšiřitelnost**

  celého obecného formátu i jakéhokoliv odvozeného specifického formátu.

Požadavky na obecný datový formát

- Uniformita
- Principy stand-off anotace
- Jednotný systém odkazů
- Lineární a strukturní data
- Strukturované atributy
- Víceznačnost
- Čitelnost pro člověka
- Rozšiřitelnost
- **XML**

široce rozšířeno, mnoho existujících nástrojů...

PML (1)

Každá rovina má svůj formát specifikován pomocí *PML-schématu*.

PML (1)

Každá rovina má svůj formát specifikován pomocí *PML-schématu*.

- **PML-schéma**

  je formalizací abstraktního konceptu anotačního schématu pro danou rovinu. Definuje strukturu anotace společně s rolemi, které jsou přiřazeny některým prvkům anotace.

PML (1)

Každá rovina má svůj formát specifikován pomocí *PML-schématu*.

- PML-schéma

  je formalizací abstraktního konceptu anotačního schématu pro danou rovinu. Definuje strukturu anotace společně s rolemi, které jsou přiřazeny některým prvkům anotace.

- **struktura anotace**

  Datová struktura vybudovaná z abstraktních datových typů.

PML (1)

Každá rovina má svůj formát specifikován pomocí *PML-schématu*.

- PML-schéma

  je formalizací abstraktního konceptu anotačního schématu pro danou rovinu. Definuje strukturu anotace společně s rolemi, které jsou přiřazeny některým prvkům anotace.

- struktura anotace

  Datová struktura vybudovaná z abstraktních datových typů.

- **abstraktní datové typy**

  pokrývají nejčastější typy datových struktur: atomické hodnoty, dvojice typu atribut–hodnota, seznamy, alternativy, apod.

PML (1)

Každá rovina má svůj formát specifikován pomocí *PML-schématu*.

- PML-schéma

  je formalizací abstraktního konceptu anotačního schématu pro danou rovinu. Definuje strukturu anotace společně s rolemi, které jsou přiřazeny některým prvkům anotace.

- struktura anotace

  Datová struktura vybudovaná z abstraktních datových typů.

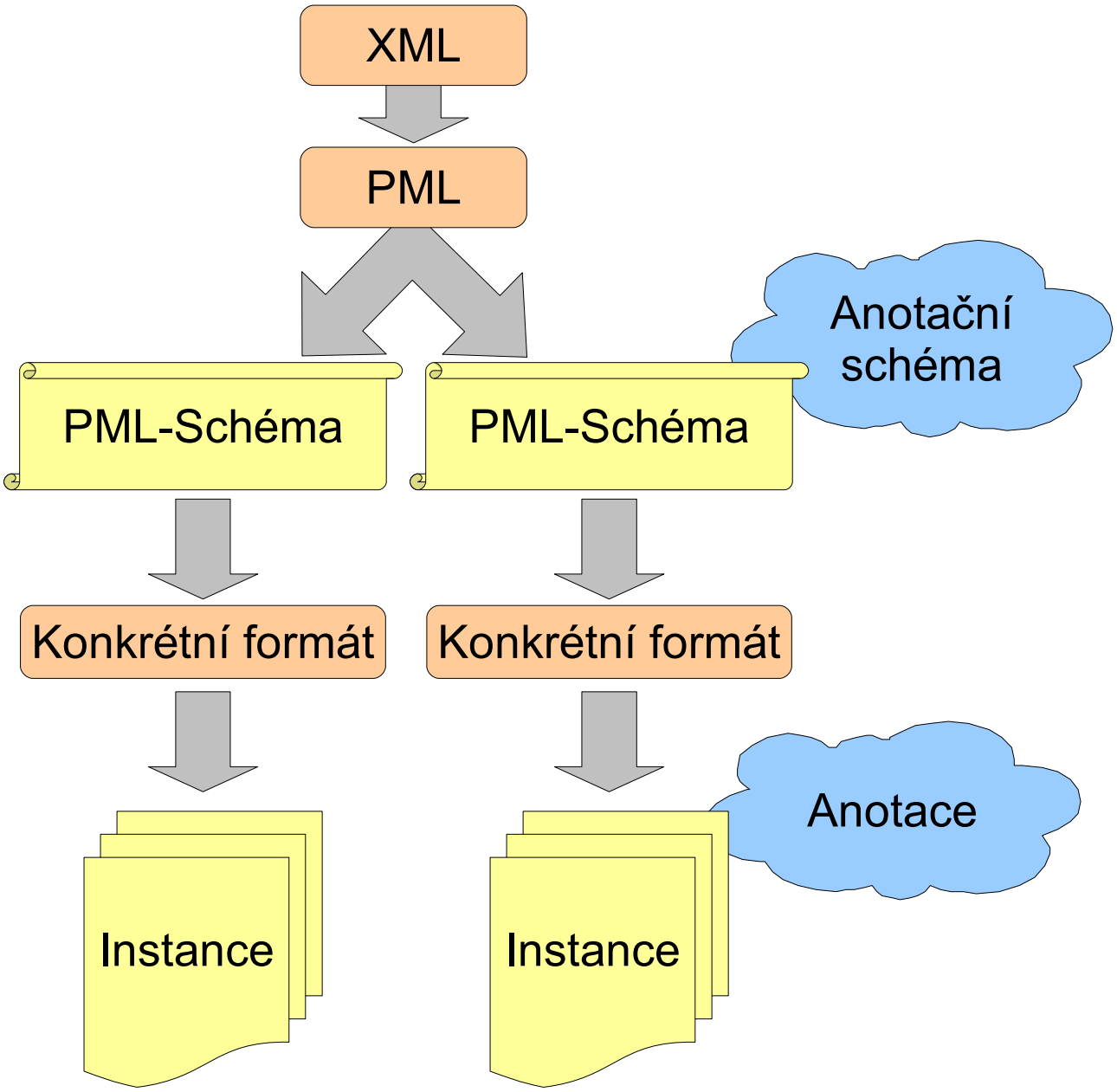
- abstraktní datové typy

  pokrývají nejčastější typy datových struktur: atomické hodnoty, dvojice typu atribut–hodnota, seznamy, alternativy, apod.

- **PML role**

  Role označuje prvek anotace za nositele vlastnosti vyššího řádu, např. uzel stromu, jednoznačný identifikátor, apod.

PML (2)


XML
PML
Anotační schéma
PML-Schéma
PML-Schéma
Konkrétní formát
Konkrétní formát
Anotace
Instance
Instance

PML (3)

XML dokument obsahující anotaci a odpovídající PML schématu se nazývá *instancí PML*.

PML instance mohou být zpracovávány:

PML (3)

XML dokument obsahující anotaci a odpovídající PML schématu se nazývá *instancí PML*.

PML instance mohou být zpracovávány:

- libovolnými XML nástroji (pomocí DOM, XPath, atd.)

PML (3)

XML dokument obsahující anotaci a odpovídající PML schématu se nazývá *instancí PML*.

PML instance mohou být zpracovávány:

- libovolnými XML nástroji (pomocí DOM, XPath, atd.)
- nástroji vytvořenými speciálně pro daný formát (znalost formátu přímo v kódu)

PML (3)

XML dokument obsahující anotaci a odpovídající PML schématu se nazývá *instancí PML*.

PML instance mohou být zpracovávány:

- libovolnými XML nástroji (pomocí DOM, XPath, atd.)
- nástroji vytvořenými speciálně pro daný formát (znalost formátu přímo v kódu)
- inteligentními obecnými nástroji, které umějí zpracovávat PML-schéma:
  - deklarace datových typů $\rightarrow$ optimální vnitřní reprezentace
  - přiřazení rolí $\rightarrow$ odpovídající prezentace anotace uživateli, poskytnutí dalších vhodných funkcí (indexace apod.)

PML (3)

XML dokument obsahující anotaci a odpovídající PML schématu se nazývá *instancí PML*.

PML instance mohou být zpracovávány:

- libovolnými XML nástroji (pomocí DOM, XPath, atd.)
- nástroji vytvořenými speciálně pro daný formát (znalost formátu přímo v kódu)
- inteligentními obecnými nástroji, které umějí zpracovávat PML-schéma:
  - deklarace datových typů → optimální vnitřní reprezentace
  - přiřazení rolí → odpovídající prezentace anotace uživateli, poskytnutí dalších vhodných funkcí (indexace apod.)

Validace:

- pomocí běžných validačních nástrojů pro XML jako `xmllint` nebo `jing`.
  (PML-schéma se dá na RelaxNG-schéma přeložit pomocí XSLT)

Datové typy

Datové typy

- **struktura hodnot (AVS)**

  množina dvojic atribut–hodnota. Hodnota může být libovolného typu.

Datové typy

- **struktura hodnot (AVS)**

  množina dvojic atribut–hodnota. Hodnota může být libovolného typu.

Deklarace PML-schématu

```
<type name="m.type">
<structure>
  <member name="form">
    <cdata format="token"/>
  </member>
  <member name="lemma">
    <cdata format="token"/>
  </member>
  <member name="tag"
     type="tagset.type"/>
</structure>
</type>
```

$\Longrightarrow$

příklad instance

```
<...>
  <form>walking</form>
  <lemma>walk-1</lemma>
  <tag>VBG</tag>
</...>
```

Datové typy

- struktura hodnot (AVS)
- **kontejner**

  umožňuje přiřadit jedné hodnotě daného typu sadu atributů (tedy množinu dvojic atribut–hodnota, kde všechny hodnoty jsou řetězce).

Datové typy

- struktura hodnot (AVS)
- **kontejner**

  umožňuje přiřadit jedné hodnotě daného typu sadu atributů (tedy množinu dvojic atribut–hodnota, kde všechny hodnoty jsou řetězce).

Deklarace PML-schéma

```
<type name="word.type">
  <container>
    <attribute name="id" role="#ID">
      <cdata format="ID"/>
    </attribute>
    <cdata format="token"/>
  </container>
</type>
```

příklad instance

$\Longrightarrow$ `<... id="w-23">Walking</...>`

Datové typy

- struktura hodnot (AVS)
- kontejner
- **seznam**

  seskupení několika hodnot stejného typu. Seznam může být uspořádaný i neuspořádaný.

Datové typy

- struktura hodnot (AVS)
- kontejner
- **seznam**

  seskupení několika hodnot stejného typu. Seznam může být uspořádaný i neuspořádaný.

Deklarace PML-schéma

```
<type name="sent.type">
  <list ordered="1"
        type="word.type">
</type>
```

$\Longrightarrow$

Příklad instance

```
<...>
  <LM id="w-33">Flies</LM>
  <LM id="w-35">like</LM>
  <LM id="w-36">an</LM>
  <LM id="w-37">arrow</LM>
  <LM id="w-38">.</LM>
</...>
```

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- **alternativa**

  paralelní seskupení několika hodnot stejného typu, např. alternativní anotace.

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- **alternativa**

paralelní seskupení několika hodnot stejného typu, např. alternativní anotace.

Deklarace PML-schéma

```
<type name="morph.type">
  <alt type="m.type">
</type>
```

$\Longrightarrow$

Příklad instance

```
<AM>
  <form>flies</form>
  <lemma>fly-1</lemma>
  <tag>VBZ</tag>
</AM>
<AM>
  <form>flies</form>
  <lemma>fly-2</lemma>
  <tag>NNS</tag>
</AM>
```

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- **sekvence**

  seskupení hodnot různých typů. Jména členů určují jejich datový typ. Uspořádání členů, počet jejich výskytů apod. může být specifikováno pomocí regulárního výrazu.

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- **sekvence**

Deklarace PML-schéma

```
<type name="chapter.type">
  <sequence
    content_pattern
      ="para*, sect+">
    <element name="para"
        type="para.type"/>
    <element name="sect"
        type="sect.type">
  </sequence>
</type>
```

$\Longrightarrow$

Příklad instance

```
<...>
  <para>
    In this chapter...
  </para>
  <sect>...</sect>
  <sect>...</sect>
</...>
```

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- sekvence
- **výčtový typ**

  předem daná množina možných hodnot řetězcového typu

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- sekvence
- **výčtový typ**

předem daná množina možných hodnot řetězcového typu

Deklarace PML-schéma

```
<type name="boolean.type">
  <choice>
    <value>TRUE</value>
    <value>FALSE</value>
  </choice>
</type>
```

$\Longrightarrow$

Příklad instance

```
<...>TRUE</...>
```

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- sekvence
- výčtový typ
- **cdata**

  řetězec, který lze dále omezit formátem — většina typů definovaná v ***XML Schema***

Datové typy

- struktura hodnot (AVS)
- kontejner
- seznam
- alternativa
- sekvence
- výčtový typ
- **cdata**

Deklarace PML-schéma		Příklad instance
`<cdata format="token"/>`		`<...>hallo234</...>`
`<cdata format="float"/>`		`<...>12.7843E-2</...>`
`<cdata format="positiveInteger"/>`		`<...>17</...>`
`<cdata format="long"/>`	$\Longrightarrow$	`<...>-9223372054775808</...>`
`<cdata format="date"/>`		`<...>1999-05-31</...>`
`<cdata format="time"/>`		`<...>13:20:00.000</...>`
`<cdata format="any"/>`		`<...>ar6!7rar¥ d@t&</...>`
`...`		`...`

PML role

PML role

- **ID**

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

PML role

- ID

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

- **KNIT**

  přiřazena odkazům vhodným ke spojení dvou anotačních rovin (objekt, na nějž se odkazuje, je vložen do odkazujícího)

PML role

- ID

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

- KNIT

  přiřazena odkazům vhodným ke spojení dvou anotačních rovin (objekt, na nějž se odkazuje, je vložen do odkazujícího)

- **TREES**

  označuje seznam nebo sekvenci stromů

PML role

- ID

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

- KNIT

  přiřazena odkazům vhodným ke spojení dvou anotačních rovin (objekt, na nějž se odkazuje, je vložen do odkazujícího)

- TREES

  označuje seznam nebo sekvenci stromů

- **NODE**

  označuje uzel stromu

PML role

- ID

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

- KNIT

  přiřazena odkazům vhodným ke spojení dvou anotačních rovin (objekt, na nějž se odkazuje, je vložen do odkazujícího)

- TREES

  označuje seznam nebo sekvenci stromů

- NODE

  označuje uzel stromu

- **CHILDNODES**

  označuje prvek uzlu, který obsahuje seznam (sekvenci) synů

PML role

- ID

  přiřazena členům nebo atributům jednoznačně identifikujícím AVS nebo kontejner v rámci instance

- KNIT

  přiřazena odkazům vhodným ke spojení dvou anotačních rovin (objekt, na nějž se odkazuje, je vložen do odkazujícího)

- TREES

  označuje seznam nebo sekvenci stromů

- NODE

  označuje uzel stromu

- CHILDNODES

  označuje prvek uzlu, který obsahuje seznam (sekvenci) synů

- **ORDER**

  označuje číselnou hodnotu, která vyjadřuje pořadí na uzlech stromu

Odkazy

Odkazy

- V současnosti jsou podporovány pouze odkazy založené na identifikátorech.

Odkazy

- V současnosti jsou podporovány pouze odkazy založené na identifikátorech.

Odkazy pomocí ID:

- Odkaz uvnitř jedné PML-instance

Odkazy

- V současnosti jsou podporovány pouze odkazy založené na identifikátorech.

Odkazy pomocí ID:

- Odkaz uvnitř jedné PML-instance
  Příklad:

```
<coref.rf>t-node-232</coref.rf>
```

Odkazy

- V současnosti jsou podporovány pouze odkazy založené na identifikátorech.

Odkazy pomocí ID:

- Odkaz uvnitř jedné PML-instance
  Příklad:

```
<coref.rf>t-node-232</coref.rf>
```

- Odkazy na jiné instance
  Typicky mnoho odkazů jen do několika málo cílových instancí.

Odkazy

- V současnosti jsou podporovány pouze odkazy založené na identifikátorech.

Odkazy pomocí ID:

- Odkaz uvnitř jedné PML-instance
  Příklad:

```
<coref.rf>t-node-232</coref.rf>
```

- Odkazy na jiné instance
  Typicky mnoho odkazů jen do několika málo cílových instancí.

  Zbytečné opakování jména souboru — dvě části odkazu:
  - specifikace cílové instance — označení (ID) cílové instance v hlavičce odkazující instance
  - ID cílového objektu

Odkazy - příklady

Přiřazení cílových URL s identifikátorem v hlavičce odkazující instance:

```
<references>
  <reffile id="a" href="doc73.a"/>
  <reffile id="v" href="http://mysite/vallex.xml"/>
</references>
```

Odkazy - příklady

Přiřazení cílových URL s identifikátorem v hlavičce odkazující instance:

```
<references>
  <reffile id="a" href="doc73.a"/>
  <reffile id="v" href="http://mysite/vallex.xml"/>
</references>
```

Příklady odkazů

```
<val_frame.rf>v#f2234</val_frame.rf>

<lex.rf>a#doc73-w5</lex.rf>

<aux.rf>
  <LM>a#doc73-w3</LM>
  <LM>a#doc73-w4</LM>
</aux.rf>
```

Odkazy na data v jiném formátu

V současnosti žádná omezení ani pravidla.

Odkazy na data v jiném formátu

V současnosti žádná omezení ani pravidla.

Příklad možné reprezentace odkazu na zvukový soubor v PML:

```
<references>
  <reffile id="au1" href="spk1_129.ogg"/>
</references>
...
<w id="w-12941">
  <token>_SIL_</token>
  <audio>
    <time_start>600000</time_start>
    <time_end>4700000</time_end>
    <file.rf>au1</file.rf>
  </audio>
</w>
```

Rozdělení na roviny

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.
- Roviny jsou propojeny pomocí PML-odkazů.

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.
- Roviny jsou propojeny pomocí PML-odkazů.
- Pokud to anotační struktura umožňuje, lze ke sloučení rovin použít roli `KNIT`.

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.
- Roviny jsou propojeny pomocí PML-odkazů.
- Pokud to anotační struktura umožňuje, lze ke sloučení rovin použít roli `KNIT`.

„Meziroviny“

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.
- Roviny jsou propojeny pomocí PML-odkazů.
- Pokud to anotační struktura umožňuje, lze ke sloučení rovin použít roli `KNIT`.

„Meziroviny“

- Různé přístupy mohou různě definovat roviny popisu.

  Typický příklad: tokenizace a rozdělení na věty (obvykle prováděny před morfologickou analýzou, ale např. pro arabštinu potřebují všechny tři operace probíhat najednou).

Rozdělení na roviny

- Každá rovina anotace má jedno PML-schéma.
- Roviny jsou propojeny pomocí PML-odkazů.
- Pokud to anotační struktura umožňuje, lze ke sloučení rovin použít roli `KNIT`.

„Meziroviny“

- Různé přístupy mohou různě definovat roviny popisu.

  Typický příklad: tokenizace a rozdělení na věty (obvykle prováděny před morfologickou analýzou, ale např. pro arabštinu potřebují všechny tři operace probíhat najednou).

- „Meziroviny“ mohou také vznikat jako průběžné výsledky anotačního procesu.

  PML-schémata pro takové roviny lze snadno definovat pomocí modularizace PML.

Modularizace

Snadné vytvoření PML-schématu na základě jiného.

Modularizace

Snadné vytvoření PML-schématu na základě jiného.

- **verze**

  PML-schématu lze přiřadit verzi ve tvaru `X.Y.Z...`

  Např.

```
<pml_schema
  xmlns="http://ufal.mff.cuni.cz/pdt/pml/schema/"
  version="1.1">
  <revision>1.0.2</revision>
  <description>PDT 2.0 tectogrammatic trees</description>
```

Modularizace

Snadné vytvoření PML-schématu na základě jiného.

- verze
- **import**

  Kopíruje deklaraci typu z jiného PML-schématu.

  Příklad

```
<import schema="lex_schema1.xml" type="lexitem"/>
```

  Vloží do schématu deklaraci typu `lexitem`.

```
<import schema="lex_schema2.xml" revision="1.1.2"/>
```

  Vloží do schématu všechny typy z verze 1.1.2 schématu `lex_schema2.xml`.

Modularizace

Snadné vytvoření PML-schématu na základě jiného.

- verze
- import
- **derive**

  Odvodí nový typ z dříve definovaného nebo importovaného kontejneru, sekvence, AVS nebo výčtového typu.

  Příklad

```
<derive type="lexitem" name="phrase-lexitem">
  <structure>
    <member name="phrase">
      <cdata format="any"/>
    </member>
    <delete>word</delete>
  </structure>
</derive>
```

Modularizace

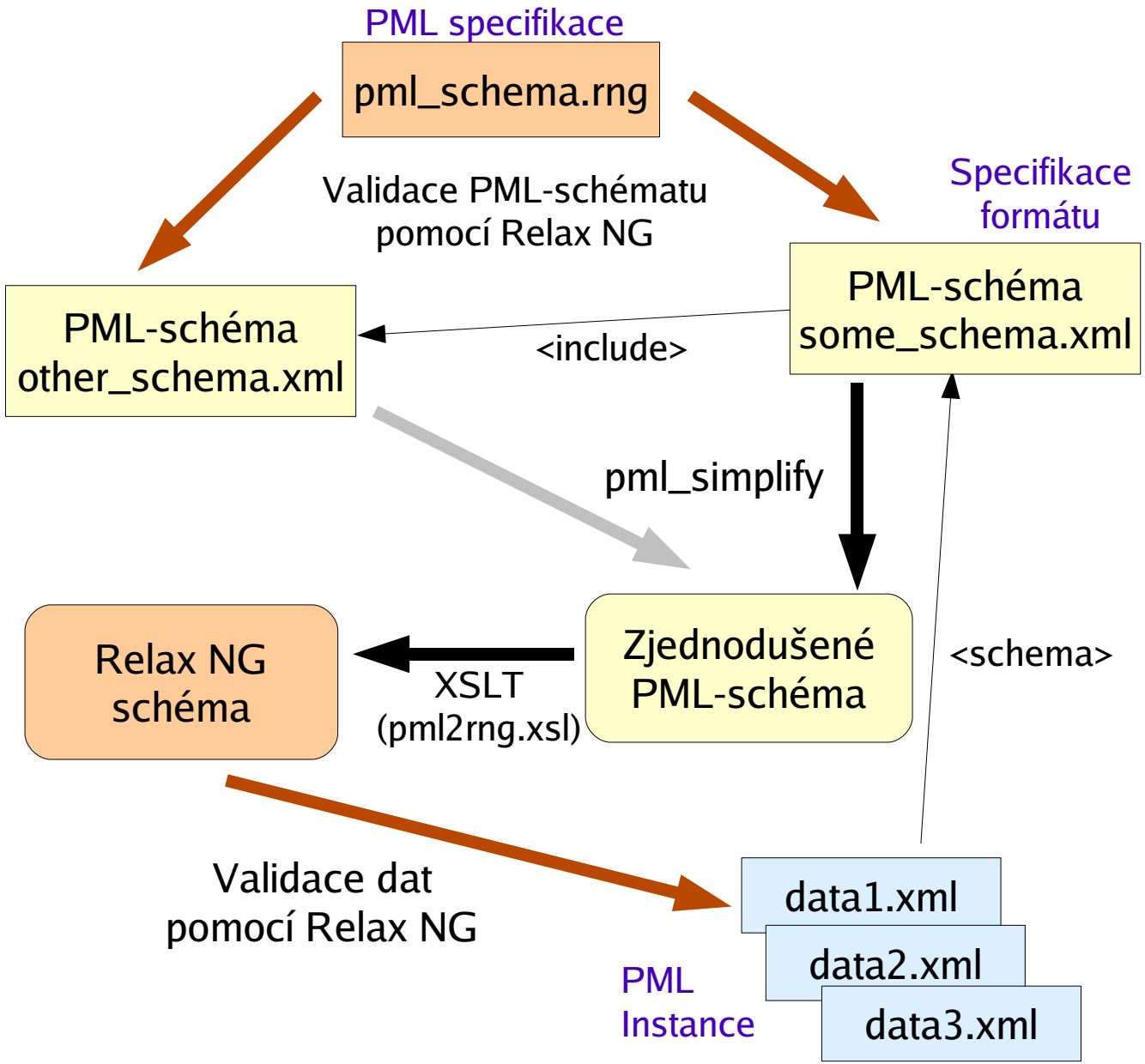
Snadné vytvoření PML-schématu na základě jiného.

- verze
- import
- derive
- **zjednodušené PML-schéma**

  `pml_simplify` - *předzpracovává PML-schéma, provádí všechny instrukce* `<import>` *a* `<derive>`.

  Nutno použít např. před XSLT 1.0 transformacemi.

Diagram modularizace PML-schématu



Technické obtíže s anotací na několika rovinách v PML

V PDT 2.0 se anotace dokumentu skládá ze *čtyř* instancí (každá pro jednu rovinu), které obsahují odkazy mezi sebou.

To však vede k potížím pro uživatele:

- PML instance se nedají volně kopírovat (nezle např. přejmenovat soubor bez opravení reference uvnitř dalších souborů, které na něj odkazují)
- Data jsou distribuována komprimovaná pomocí nástroje **gzip**. Stejný problém: rozbalením dojde ke změně názvu souborů (odstranění přípony `.gz`). Je tedy třeba opravit odkazy.

Další vývoj

- **podpora pro namespace**

  povolení XML dat z jiných namespace než PML uvnitř PML-instancí (MathML, XLink, RDF,...)

Další vývoj

- podpora pro namespace

  povolení XML dat z jiných namespace než PML uvnitř PML-instancí (MathML, XLink, RDF,...)

- **meta-data**

  jednotná reprezentace (např. pomocí RDF)

Další vývoj

- podpora pro namespace

  povolení XML dat z jiných namespace než PML uvnitř PML-instancí (MathML, XLink, RDF,...)

- meta-data

  jednotná reprezentace (např. pomocí RDF)

- **nové role**

  Např. role pro ukládání slovníků ve formátu PML:

  `ITEM` *pro slovníková hesla,*
  `INDEXABLE` *pro seznamy hesel s indexovacím klíčem,*
  `INDEXKEY` *pro indexovací klíč, apod.*

Další vývoj

- podpora pro namespace

  povolení XML dat z jiných namespace než PML uvnitř PML-instancí (MathML, XLink, RDF,...)

- meta-data

  jednotná reprezentace (např. pomocí RDF)

- nové role

  Např. role pro ukládání slovníků ve formátu PML:

  `ITEM` *pro slovníková hesla,*
  `INDEXABLE` *pro seznamy hesel s indexovacím klíčem,*
  `INDEXKEY` *pro indexovací klíč, apod.*

- **automatické generování API**

  překlad PML-schématu do knihovny, která půjde ihned používat, s optimální paměťovou reprezentací, validací, parserem, serializací dat, indexací apod.

Další vývoj

- podpora pro namespace

  povolení XML dat z jiných namespace než PML uvnitř PML-instancí (MathML, XLink, RDF,...)

- meta-data

  jednotná reprezentace (např. pomocí RDF)

- nové role

  Např. role pro ukládání slovníků ve formátu PML:

  `ITEM` *pro slovníková hesla,*
  `INDEXABLE` *pro seznamy hesel s indexovacím klíčem,*
  `INDEXKEY` *pro indexovací klíč, apod.*

- automatické generování API

  překlad PML-schématu do knihovny, která půjde ihned používat, s optimální paměťovou reprezentací, validací, parserem, serializací dat, indexací apod.

- **další typy odkazů**

  text, audio, video, grafika...

PML specifikace

http://ufal.mff.cuni.cz/jazz/PML

Děkuji.

Otázky?