**Seminární práce XML**

Seminární práce na XML se skládá ze dvou úkolů:

* Případová studie: Produktový XML Feed (8b)
* Extrakce informací z CI zprávy (4b)

První úkol je zaměřen na komplexní procvičení XML technologií na případu strukturovaných dat na vstupu. Druhý úkol se soustředí na využití regulárních výrazů, přičemž XML technologie se používají jako rámec pro vykonání regulárních výrazů a souhrnnou prezentaci výsledků extrakce.

**Případová studie: Produktový XML Feed**

Management Vaší firmy se rozhodl využít XML pro popis nabízených produktů/služeb. Cílem této iniciativy je standardizovat a zautomatizovat komunikaci s obchodními partnery a zákazníky.

Vaším úkolem jako podnikového informatika je:

1. navrhnout XML formát pro tuto komunikaci,
2. navrhnout efektivní způsob tvorby XML souborů odpovídajících tomuto formátu,
3. připravit zobrazení XML ve formátu HTML pro management firmy,
4. zdokumentovat využití XML technologií v bodech 1-3,
5. popsat možné obchodní příležitosti pro využití navrženého XML formátu.

**Úkoly 1,2,3: Podrobné zadání**

* + Při návrhu XML formátu vyjděte ze specifikace na stránkách [Seznamu](http://napoveda.seznam.cz/cz/specifikace-xml.html) nebo [Jyxa](http://admin.vybereme.cz/format-xml-feedu) (jedná se v podstatě o tentýž formát), která je v jazyce XML Schéma uložena na ISIS v souboru **zbozi\_feed.xsd.**
  + Do tohoto schématu přidejte formou elementů alespoň 5 nejvýznamnějších parametrů produktů, které vaše firma nabízí (tzn. například technické specifikace palivových kotlů atd.). Hodnoty alespoň jednoho z nově přidaných elementů musí být kontrolovány pomocí regulárního výrazu.
  + Vytvořte XML soubor odpovídající tomuto schématu, který bude obsahovat alespoň 5 produktů věcně souvisejících s tématem vaší informační studie[[1]](#footnote-1) (viz. úkol č. 1). Vyjděte ze souboru popisujícího fair-trade produkty **zbozi\_feed.xml**.
  + Ujistěte se, že je vytvořený XML soubor validní proti schématu.
  + Zajistěte uživatelsky přívětivé zobrazení ve webovém prohlížeči pomocí XSL stylu. Vyjděte ze stylu **zbozi\_feed.xsl** pro fairtrade produkty, který upravte tak, aby výsledný seznam odpovídal kultuře Vaší firmy (změna textu nadpisů, fontu, uspořádání tabulek).
  + Pomocí upraveného XSL stylu a nově vytvořeného XML s produkty firmy vygenerujte XHTML dokument. Ujistěte se, že vygenerovaný soubor je validní pomocí W3C validátoru.
  + Stručně představte použité technologie (XML, XSL, XSD) a zdokumentujte způsob, jakým jste je použili.

**Úkoly 4,5: Dokumentace a business scénář**

Popište modelové využití vytvořeného souboru ve Vaší firmě, zaměřte se především na:

* **Dokumentaci.** Dokumentace by měla především popisovat nově přidané elementy.
* **Vznik XML Feedu**. Doporučujete pro Vaši firmu ruční tvorbu XML Feedu nebo automatické generování? Jaké konkrétní nástroje a technologie by byly použity?
* **Využití XML.** Pokud nejsou produktové vyhledávače (B2C) pro vaši firmu relevantní, zaměřte se na využití v komunikaci B2B.
* **Využití XML Schéma.** Popište možnosti využití formalizované definice výměnného formátu pomocí XML Schéma.
* **Využití XSL transformací.** XSL transformaci lze kromě transformace do XHTML pro vizualizaci XML využít i pro komunikaci mezi informačními systémy, které využívají různé XML formáty. Popište takovýto scénář relevantní Vaší firmě.

Podklady pro úkol (viz ISIS):

* **zbozi\_feed.xsd** XML Schéma
* **zbozi\_feed.xsl**  XSL transformace
* **zbozi\_feed.xml**  Ukázkový XML soubor
* **zbozi\_feed.html** Výstup transformace na ukázkovém souboru
* **zbozi\_feed.css** kaskádovýstyl definující některé vizuální parametry výstupního html

Odevzdávané soubory (**odevzdejte zabalené v souboru: ukol\_XML\_XXX.zip**):

* **zbozi\_feed\_XXX.xml**  XML se seznamem produktů (1b)
* **zbozi\_feed\_XXX.xsd** Upravené schéma (1,5b)
* **zbozi\_feed\_XXX.xsl** Upravená transformace (1,5b)
* **zbozi\_feed\_XXX.html** Výsledek transformace (1,5b)
* **vyuziti\_XXX .(doc|html|rtf)** Dokumentace apřípadová studie (2,5b)

(za **XXX** doplňte jméno Vašeho týmu)

**Software**

**Volný**

* Kernowforsaxon Kernowforsaxon.sourceforge.net (doporučeno)

Validace, XSL transformace

* JEdit (dostupný v síti VŠE)

Tvorba XML dokumentů, validace, XSL transformace

**Komerční**

* **Oxygen XML Editor** (doporučeno, dostupný v síti VŠE)

Propracovaná tvorba XML dokumentů, validace, XSL transformace, …

Javová aplikace (i Linux)

**Licenci pro domácí účely získáte na** [**https://badame.vse.cz/oxygen/index.php**](https://badame.vse.cz/oxygen/index.php)

* **Altova XML Spy**

Propracovaná tvorba XML dokumentů, Validace, XSL transformace, …

**Testování regulárních výrazů pro XML Schéma**

Použijte XML Schema Regular Expression Builder vestavěný do Oxygenu (přístupný z menu Tools)

**Extrakce informací z CI zprávy**

Management firmy Vás pověřil úkolem vytvořit program pro automatickou extrakci určitých informací z textových dokumentů vkládaných do informačního systému firmy. Tyto informace se budou využívat jako metadata pro rychlé nalezení požadovaného dokumentu. Management rovněž slyšel o sémantické anotaci jako alternativním způsobu řešení tohoto problému extrakce metadat a zadal vám vypracování krátkého srovnání obou metod. Vaším úkolem, coby podnikového informatika, je:

1. Sestrojit regulární výrazy pro extrakci metadat, které ve zprávě CI odevzdané v prvním úkole naleznou zadané druhy údajů.
2. Regulární výrazy vložit do XSL transformace, která na vstupu přijme CI zprávu (ve formátu.txt - čistý text) a na výstupu vygeneruje XML dokument přehledně shrnující nalezené informace dle typu. XSL transformaci vytvořte úpravou vzoru.
3. Napsat krátkou zprávu o tom, kdy považujete za vhodné použít sémantickou anotaci, a kdy extrakci pomocí regulárních výrazů.

**Úkol 1: Tvorba regulárních výrazů pro extrakci metadat (2 body)**

Ze zprávy vyextrahujte následující druhy údajů:

* NACE kód (ne pouze číslo, také textové vysvětlení kódu!)
  + (\d{6}):\w\*(.+)
* Adresy
  + ((?:\p{Lu}\p{Ll}+ ?)+) (\d+(?:/\d+)?), (\d{3} \d{2}) ((?:\p{Lu}\p{Ll}+ ?)+), ((?:\p{Lu}\p{Ll}+ ?)+)
* *+ další 3 druhy údajů dle vlastní volby*
* \s\*(\d+(?:/\d+)?),\s\*(\d{{3}} \d{{2}})\s+((?:\p{{Lu}}\p{{Ll}}+ ?)+),\s\*((?:\p{{Lu}}\p{{Ll}}+ ?)+)
* *HS*
  + *HS \d{4}(\.?\d{2}\.?\d{4}|\.?\d{2})?*
* *ISO*
  + *(ČSN EN )?ISO(/IEC)? \d{1,5}(:[12]\d{3})?*
* *Tel.*
  + *\+\d{3} ?\d{3} ?\d{3} ?\d{3} ?*

*Zásady pro tvorbu regulárních výrazů***-**  **Zprávu není dovoleno upravovat**. Kontrola správnosti výrazů bude probíhat proti zprávě odevzdané v rámci prvního úkolu.  
**-** **Regulární výraz by měl postihovat všechny výskyty daného údaje** v textu, které jsou alespoň částečně strukturovaně zadány a u kterých lze očekávat, že regulární výrazy pro ně sestrojené by byly použitelné i na jiné dokumenty.

|  |
| --- |
| Například pokud vaše zpráva obsahuje text *"firma investovala velké prostředky kvůli novelizaci zákona o životním prostředí"* pak tento výskyt požadované informace nemusí regulární výraz extrahovat*.* Naopak text *"Zákon č. 17/1992 Sb. o životním prostředí "*je již dostatečně strukturovaný a regulární výraz extrahující právní normy by ho měl postihovat. |

Pokud je určitý druh informace zapsán v dokumentu výrazně odlišnými způsoby, které by vyžadovaly samostatné regulární výrazy, sestrojte nejdříve tyto výrazy samostatně a následně je spojte do jednoho výrazu pomocí operátoru „|” a vhodného seskupování pomocí závorek ().

**-** **Vytvořte více regulárních výrazů pro zachycení jednoho typu údaje, pokud se jedná o jemnější rozlišení.** Například pokud budete tvořit regulární výraz pro extrakci *právních norem,* můžete vytvořit samostatně regulární výrazy rozpoznávající vyhlášky a zákony.

**-** **Nahraďte příslušný bod zadání vlastním,** pokud zpráva neobsahuje požadovanou informaci, nebo tato není dostatečně strukturovaná (viz bod B).

**Úkol 2: Vložení regulárních výrazů do XSL transformace (1 bod)**

Regulární výrazy z úkolu 1 vložte do XSL transformace **metadata.xsl**. Pro každý typ extrahovaného údaje vytvořte nový element infotype zkopírováním prvního vzorového elementu infotype. Do atributu regex instrukce xsl:analyze-string vložte regulární výraz. Protože chceme vypsat text v okolí hitu, vložte před i za regulární výraz řetězec (.{{0,30}}), tak jak je naznačeno na příkladu obsaženém v souboru **metadata.xsl**.

Dále upravte element value, ve kterém se vypisuje vlastní výsledek dotazu, tak, aby se odkazoval s využitím instrukce regex-group na správnou skupinu (skupiny) v regulárním výrazu[[2]](#footnote-2). Pokud se hledaný údaj skládá z několika částí, vypište tyto části samostatně do elementu details a do elementu value jejich zřetězení, tak jak je naznačeno v druhém elementu infotype. Vytvářené elementy infotype a details vhodně pojmenujte pomocí atributu name.

Podklady pro úkol (viz ISIS):

* **metadata-sample-input.txt** Ukázkový vstupní text vytvořený ze vzorové CI zprávy. *Nahraďte vstupem z vaší CI zprávy*
* **metadata-sample-settings.xml** Ukázková konfigurace vstupu XSL transformace metadata.xsl. *Upravte*
* **metadata.xsl**  Hlavní transformace rozpoznávající regulární výrazy. Pusťte na soubor \***settings.xml.** *Upravte.*
* **metadata-result\_style.xsl** styl pro uživatelsky přívětivé zobrazení vygenerovaného XML dokumentu v prohlížeči. *Nemusíte měnit.*
* **metadata-result.xsd** XML Schéma, kterému musí výsledek transformace metadata.xsl odpovídat. *Neměňte*.

**Úkol 3: Porovnání vhodnosti regulárních výrazů a sémantické anotace (1 bod)**

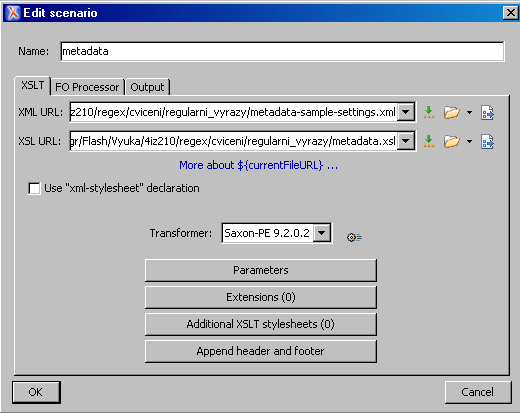
Napište krátkou zprávu (rozsah půl až jedna strana A4) porovnávající předpřipravené regulární výrazy s ručně tvořenými anotacemi z hlediska tvorby metadat.

**Odevzdávané soubory (odevzdávejte zabalené v souboru ukol\_XML\_regvyr\_XXX.zip):**

* **metadata\_XXX.txt** Vstupní text vytvořený ze vzorové CI zprávy
* **metadata\_XXX.xsl**  Upravená XSL transformace s regulárními výrazy
* **metadata\_XXX.xml** Výstup XSL transformace
* **metadatareport\_XXX .(doc|html|rtf)** Report o využití sémantické anotace/reg. výrazů
* **regex\_XXX.txt** Vytvořené regulární výrazy

**Software**  
Pro testování odevzdávaných regulárních výrazů použijte například program Regex Builder <http://www.redfernplace.com/software-projects/regex-builder/>, prohledávaný dokument vložte z textového editoru přes schránku (CTR+A, CTR+C v textovém editoru, CTR+V v Regex Builder). Ujistěte se, že nemáte zaškrtnuto Explicit Capture.

Pro úpravu a provedení XSL transformace využijte program Oxygen. Jako XSL procesor zvolte procesor podporující XSLT 2.0, např. **Saxon 9** (PE). Ukázka nastavení viz níže.

  
  
**Literatura**

Kromě podkladů k přednáškám a cvičení je vhodná následující literatura:

* <http://www.regular-expressions.info/>
* Kay, Michael, 1951- XSLT 2.0 and XPath 2.0 : programmer's reference / Indianapolis : Wiley Publishing, 2008 *(v knihovně, str. 230-237 – instrukce analyze-string a kapitola 14 o regulárních výrazech)*
* Regex in XSLT Cheatsheet: <http://www.mulberrytech.com/quickref/regex.pdf>

1. Pokud firma žádné produkty nemá, nebo o nich na www není dostatek informací, použijte jako podklad zboží podobného charakteru nebo si údaje vymyslete. [↑](#footnote-ref-1)
2. Pro vrácení skupiny s pořadovým číslem x použijte XPath 2.0 funkci regex-group(x). [↑](#footnote-ref-2)