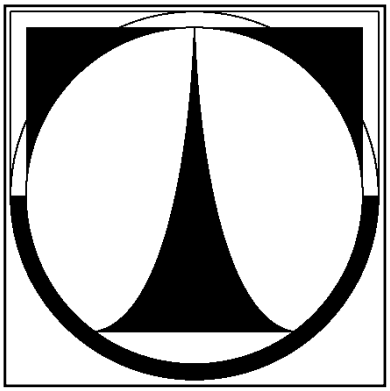
TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

DIPLOMOVÁ PRÁCE



Studijní program : N 2612
Ak. rok: 2010/2011

Vypracoval: **Bc. Václav Mareš**

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: N 2612 – Elektrotechnika a informatika
Studijní obor: 1802T007– Informační technologie

Návrh formátu pro agregaci dat na webu

Design of data pattern for the web aggregation

Diplomová práce

Autor: **Bc. Václav Mareš**
Vedoucí práce: **Mgr. Jiří Vraný Ph.D.**

V Liberci 3.1.2011

(ZDE BUDE VLOŽENO ORIGINÁLNÍ ZADÁNÍ)

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 - školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum:

Podpis:

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce Mgr. Jiřímu Vranému Ph.D. za jeho cenné rady a trpělivou pomoc.

Dále bych také rád poděkoval svým rodičům za jejich podporu během mého studia na Technické univerzitě v Liberci.

Abstrakt

Cílem této diplomové práce je navržení vhodného formátu pro webovou agregaci dat v rámci B2B elektronických tržišť s případnou možností implementace do stávajících systémů.

Nejrozšířenějším formátem u českých B2B portálů zaměřených na agregaci produktových dat je struktura popsaná pomocí značek jazyka XML bez deklarace jakéhokoliv jmenného prostoru. Jedná se o formát jednoúčelový a pro aplikace, které neznají přesný význam značek také nepoužitelný.

Práce se proto orientuje na technologie sémantického webu, které nabízejí jazyky určené ke strojovému zpracování elektronických dokumentů. Navržený formát je založen na syntaxi RDF/XML s vlastní deklarací jmenného prostoru v OWL.

Případné využití formátu je úzce závislé na podpůrných aplikacích. V druhé části diplomové práce se tedy věnuji vývoji generátoru a agregační čtečky pro navržený formát. Výsledkem je PHP knihovna, umožňující generování a úpravu produktových kanálů ve strojově čitelné podobě na základě manuální konstrukce, či převodu z jiných formátů.

Abstract

The aim of these graduation theses is a proposition of acceptable data pattern for web aggregation of data in the framework B2B of e-market place with contingent possibility of implementation into current systems.

The most widespread format of Czech portal B2B which are focused on the aggregation of product data is a structure described by marks of XML language without a declaration any name space. It’s single-purpose useless format for applications, which don’t knot exact sense of marks.

My work is oriented on a technology of semantic web, which give languages for machinery elaboration of electronic documents. Proposed format is established on syntax RDF/XML with own declaration of the name space in OWL.

Appropriate usage of format is closely dependent on support applications. In the second part of my graduation theses I study a development of generator and a feed reader for proposed format. The result is PHP library, which makes possible to generating and correction of product feed in machine-readable form on the basis of manual construction or by conversion from other formats.

OBSAH

Seznam tabulek

Seznam obrázků

Seznam použitých zkratek

B2B (Business-toBusiness) – elektronické obchody zaměřené na komerční aktivity mezi firmami

B2C (Business-to-Consumer) – elektronické obchody zaměřené na nabídku produktů koncovým spotřebitelům

C2C (Consumer-to-Consumer) – elektronické obchody zaměřené na prodej a nákup zboží mezi jednotlivými spotřebiteli

XML (Extensible Markup Language) – rozšířitelný značkovací jazyk umožňující strukturování dat

RDF (Resource Description Framework) – obecný formát pro popis, výměnu a znovupoužití metadat

RDF/XML – zápis RDF pomocí jazyka XML

OWL (Ontology Web Language) – značkovací jazyk určený pro tvorbu ontologií

RSS (Really Simple Syndication) – nejrozšířenější metadatový formát, určený k syndikaci obsahu na webu.

FOAF (Friend of a Friend) – ontologie pro popis osob, subjektů, jejich aktivit a vzájemných vztahů

DC (Dublin Core) – ontologie pro popis zdrojů, jako jsou knihy, zvuk, obraz, nebo textové soubory či webové stránky.

SOAP (Simple Object Access Protocol) – protokol určený k výměně strukturovaných dat přes síť pomocí HTTP

URI (Uniform Resource Identifier) – jedinečný identifikátor webového zdroje (skládá se z adresy objektu a schéma)

URL (Uniform Resource Locator) – podmnožina URI, taktéž slouží jako identifikátor zdroje, ale oproti URI je mnohem konkrétnější v použitém schématu

URN (Uniform Resource Names) – významově blízká množina jmen (jmenný prostor)

W3C (World Wide Web Consortium) – mezinárodní společenství, zabývající se vývojem standardů pro dlouhodobý růst webu

TBOX – slovník pojmů aplikační domény

ABOX – výroky, tvrzení o individuích popsaná pomocí pojmů z TBOXu

DOM API (Document Object Model Application Programming Interface)

– standart W3C pro reprezentaci XML nebo HTML dokumentů v podobě objektového stromu

1. Úvod

Provádění obchodní transakcí na internetu využívá čím dál více lidí i firem napříč průmyslovými odvětvími a zájmovými skupinami. Tento trend je v celku logický. Nabízí totiž značnou úsporu peněz a firmám umožňuje využití potenciálu celé sítě a nabízet své produkty všem uživatelům po celém světě. Děje se tak prostřednictvím elektronických obchodů (e-shopů), které lze dle charakteru obchodních vztahů podle Petra Spáčila[1] rozdělit na B2B, B2C a C2C.

- B2B (Business-toBusiness) – jedná se o elektronické obchody zaměřené na komerční aktivity mezi firmami. B2B obchody nabízejí zboží za účelem dalšího prodeje, proto je velký důraz kladen zejména na důkladnou specifikaci logistických požadavků a podchycení smluvních vztahů.
- B2C (Business-to-Consumer) – tento druh e-shopů je orientován na nabídku produktů koncovým spotřebitelům. Realizovány jsou především jednorázové nákupy s důrazem na oslovení zákazníka.
- C2C (Consumer-to-Consumer) – elektronický obchod C2C umožňuje prodej a nákup zboží mezi jednotlivými spotřebiteli. Typickým představitelem jsou aukční a dražební portály a předmětem prodeje je zpravidla použité zboží.

Speciálním případem B2B obchodů jsou elektronická tržiště (e-marketplace). Lze si je představit jako virtuální místo, kde se shromažďují nabídky zboží široké skupiny obchodníků. Cílovou skupinou, které je nabídka určena, nemusejí být jen obchodníci, ale i koncoví zákazníci. Z tohoto důvodu se nejedná o typický případ B2B obchodu.

Účel shromažďování (agregace) nabídek (dat) je zcela prostý. Uživatel (koncový zákazník, obchodník) má totiž na jednom místě přehled o nabídce zboží hned několika distributorů. Díky tomu tedy může porovnat cenu, parametry produktu a vybrat tu nejlepší nabídku.

Jak bylo uvedeno výše, elektronická tržiště jsou založena na jednostranné výměně dat, která poskytuje obchodník prostřednictvím svých produktových kanálů (feedů). Existuje celá řada technologií a formátů, které jsou k tomuto účelu určeny. Návrh formátu a výběr vhodné technologie k výměně dat v rámci B2B elektronických tržišť je stěžejním předmět dalších kapitol.

[1] SPÁČIL, P. Modely elektronického obchodu v celosvětovém podnikání [online]. Vědecko-populární článek z Vysoké školy báňské - Technická univerzita Ostrava, 2009.
URL: <http://moodle.vsb.cz/moodle/mod/resource/view.php?inpopup=true&id=79283>

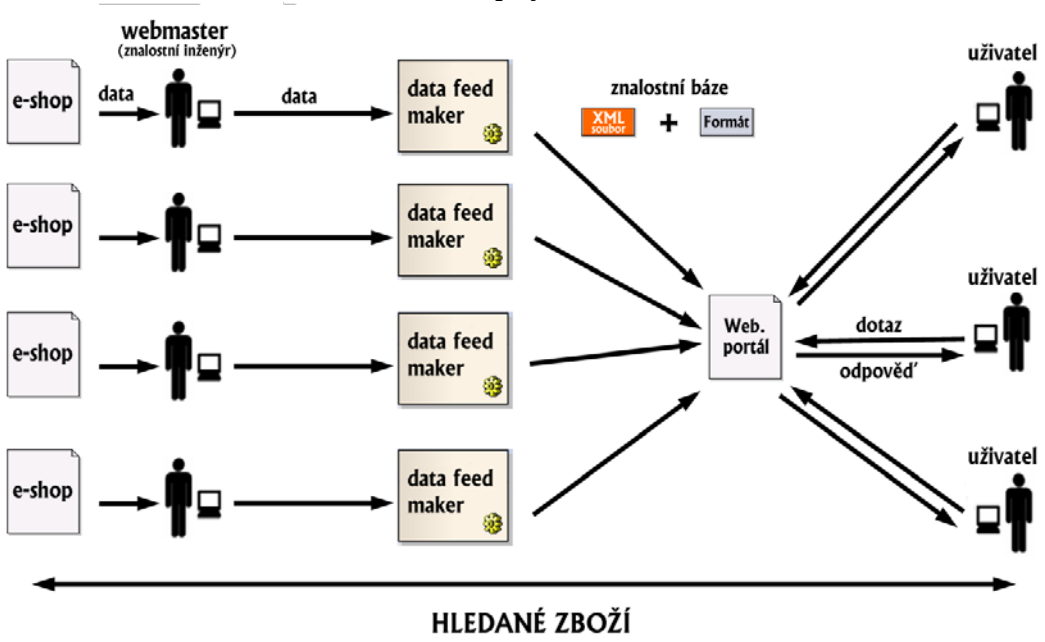
2. Princip funkce B2B elektronických tržišť

Agregace, nebo také online syndikace obsahu, je pojem používaný pro shromažďování opakovaně dodávaných aktuálních informací dostupných na webu. Distribuci obsahu provádí syndikátor (syndicator). Může to být webová aplikace, v našem případě e-shop, potažmo administrátor e-shopu, který uveřejní svůj dokument s obsahem určeným k syndikaci. Zvěřejněný dokument přebírá odběratel (agregátor, subscriber), který ho zpracuje a získané informace nabízí uživatelům. Odběratelem může být specializovaný portál, elektronická tržiště (zbozi.cz, google.com/base, heurek.cz atd.), nebo aplikace jako jsou RSS, RDF a ATOM čtečky. Výsledkem jsou kolekce dat, prezentované na jednom místě. Uživatel tak k získání hledaných informací nemusí navštěvovat velké množství webů, ale pouze jeden portál (osobní čtečku).

Aby bylo možné zmiňované kolekce vytvářet, musí být data prezentována v přesně definovaném formátu (podrobněji viz kapitola 6), za použití některého z reprezentačních schémat (podrobněji viz následující kapitola 3) a jedinečného identifikátoru URI.

Princip webové agregace dat je patrný z obrázku 1.

Obr. 1: Princip syndikace obsahu



Zdroj: Vlastní zpracování

3. Reprezentační schémata

Důležitým pojmem v oblasti popisu dat jsou metadata. Jedná se o strukturovaná data poskytující rozšiřující informace o jiných datech. Dle článku Sémantický web a jeho technologie[2] od autorů P. Matulíka a T.Pitnera metadata zachycují obsah, kontext a strukturu dat, které popisuji. Jsou důležitá ke správné interpretaci pojmů a usnadňují automatizované zpracování webových zdrojů. Zápis metadat umožňují jazyky pro popis dat (reprezentační schémata). Těmi nejdůležitějšími jsou:

- XML,
- RDF a RDF Schema,
- OWL.

Základním stavebním kamenem popisu dat je XML (Extensible Markup Language). Reprezentační schémata (RDF, RDF Schema a OWL) jsou rozšířenými implementacemi jazyka XML.

3.1 XML

Jedná se o jednoduchý rozšiřitelný značkovací jazyk, který umožňuje strukturování dat. Pojem značkovací jazyk vychází ze skutečnosti, že uspořádání jednotlivých částí dokumentů je realizováno pomocí značek (tagů), které popisují význam jednotlivých částí textu. Dokument je tak informačně bohatší a umožňuje mnohem širší škálu uplatnění. Jednou z možností použití XML je popis produktových kanálů (feedů) k výměně produktových dat. Toho využívají internetové portály, jako je např. zbozi.cz, heureka.cz, google.com/base a další.

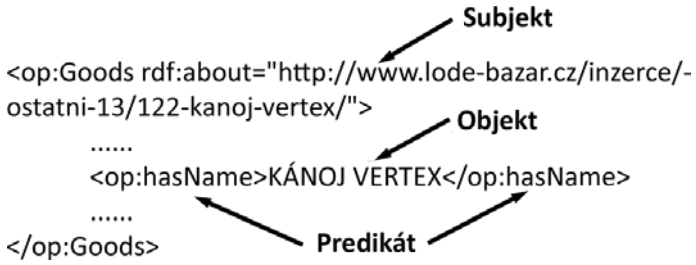
Značky, které jsou k tomuto popisu využívány definují strukturu dokumentu (formát). Jejich význam však už není přesně definován - jedná se o implicitní sémantiku. Aplikace, které s tímto formátem pracují, musejí přesně znát co která značka znamená, jinak je pro ni dokument nepoužitelný a nastává problém mnohoznačností a špatné interpretace obsahu dokumentu. Pokud s dokumentem pracují lidé, tak musí dojít k jednotnému konsensu o významu použitých značek.

[2] MATULÍK, P. ; PITNER, T. Sémantický web a jeho technologie [online]. Zpravodaj ÚVT MU. ISSN 1212-0901, 2004, roč. XIV, č. 3, s. 15-17.
URL: <http://www.ics.muni.cz/zpravodaj/clanky_tisk/296.pdf>.

3.2 RDF

Standard RDF (Resource Description Framework) je obecný formát pro popis, výměnu a znovupoužití metadat a využívá syntaktického zápisu v XML. Je založen na předpokladu, že každý zdroj lze popsat sadou jeho vlastností. Definován je pomocí tzv. trojic (tripple), které se skládají ze zdroje, vlastnosti a hodnoty vlastnosti. Trojici lze také interpretovat jako tvrzení, kde subjekt má nějakou vlastnost objektu. Syntaktický zápis těchto trojic lze vyjádřit několika možnými způsoby, avšak konsorcium V3C doporučuje použití syntaxe XML.[3] Zápis RDF pomocí jazyka XML (viz obrázek 2) je označován jako RDF/XML a lze ho chápat jako metajazyk (rámec) pro popis dalších jazyků.

Obr. 2: Příklad RDF/XML syntaxe



Zdroj: Vlastní zpracování

Z uvedeného příkladu (viz obrázek 2) vyplývá, že subjekt je reprezentován pomocí URI a objekt tzv. literálem (hodnotou), ne vždy tomu ale tak musí být. V případě objektu se totiž může také jednat o URI. URI by však měla být vždy unikátní a mohla tak jednoznačně identifikovat zdroj.

3.3 Jmenný prostor

Významově blízkou množinu jmen lze označit jako jmenný prostor - URN (Uniform Resource Names). Používá se k rozlišení XML elementů se stejným názvem, nebo datovým typem. V dokumentu tak lze definovat shodné elementy s rozdílnými významy. Deklarace jmenného prostoru v RDF je definována pomocí URI atributu z jmenného prostoru xmlns. Jednotlivé elementy dokumentu lze přiřadit k jmennému prostoru použitím prefixu, což je v podstatě zkrácený zápis jmenného prostoru. Výsledný indetifikátor je pak ve tvaru ***xmlns : prefix = "URI jmenného prostoru"***.

[3] BECKETT, D . ; W3C. RDF/XML Syntax Specification (Revised) [online]. W3C Recommendation, 10 February 2004. URL: <http://www.w3.org/TR/REC-rdf-syntax/>.

3.4 RDF Schema

RDF Schema je sémantickou nadstavbou RDF. Stejně jako jazyk RDF umožňuje popis zdrojů a jejich vzájemných vztahů. Do struktury však také zavádí typově podobné pojmy známé z objektově orientovaného programování: třídy, podtřídy a binární sloty. Nad těmito literály umožňuje definovat hiearchii tříd a podtříd nebo třeba také doménu a rozsah vlastností.

I tento formát však má svá omezení. Mezi třídami nelze definovat disjunktnosti a u vlastností zase jejich rozsah a kardinalitu. Zmíněné nedostatky vedly k vývoji nových rozšíření, která by umožňovala popis těch nejsložitějších aplikací sémantického webu. Výsledkem byl modelovací ontologický jazyk OWL.

3.5 Ontologie

Nezbytnou součástí sémantického webu jsou tzv. ontologie. Pomocí nich lze jasně definovat význam pojmů, jejich vzájemné vztahy a gramatiku pro jejich použití. Podle definice T.R.Grubera[4] je ontologie: „Specifikace sdílené konceptualizace“. Což znamená, že systém pojmů popisujících určitou oblast (konceptualizace) musí být popsán pomocí jazyka s přesně definovanou syntaxí (formální specifikace) a na základě konsensu širší skupiny lidí. Z pohledu znalostního inženýrství lze ontologii také vnímat jako datový model, určený ke sdílení mezi procesy a to nezávisle na své finální reprezentaci.

Specifikací zápisů ontologií je hned několik. Nejstarší z nich je již jednou zmiňovaný koncept RDF schema, dále jeho rozšíření v podobně DAML+OIL nebo třeba také již ustálený jazyk OWL. Existují však i jiné specifikace. Jejich popis by však zasahoval mimo rámec této práce, proto je uveden pouze zdroj s odkazem na doplňující informace. [5]

3.6 OWL

OWL (Ontology Web Language) je značkovací jazyk vyvinutý organizací W3C pro tvorbu ontologií využitelných v prostředí sémantického webu (viz následující kapitola 4.). Obsahuje množinu axiomů popisujících třídy, vlastnosti a vztahy mezi nimi. Dále

[4] GRUBER, T.R.. A Translation Approach to Portable Ontology Specifications [online]. Knowledge Acquisition, 1993.
URL:<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.117.3273&rep=rep1&type=pdf >.
[5] ABDENADHER, S. Ontology and Knowledge Representation Formalisms [online]. Vědecko-populární článek z univerzity v Mnichově, 2009
URL:<http://www.pms.ifi.lmu.de/mitarbeiter/ohlbach/Ontology/index.html>.

nabízí použití několik dialektů jazyka. Nejjednodušším je OWL light, který je určen pro začínající uživatele, vytvářejí základní klasifikační hierarchickou strukturu dokumentu s jednoduchými omezeními.[6] Jeho rozšířením je OWL-DL zaměřený na deskripční logiku. S OWL-DL je možné realizovat odvozování, např. doplnit ontologickou hierarchii a případně zkontrolovat, zda je ontologie konzistentní. Uživatelé s požadavkem na maximální stupně vyjadřovací síly jazyka mohou využít kompletní verze OWL Full. Jednotlivé dialekty mají rozdílné odvozovací schopnosti a platí, že čím je vyšší výrazová schopnost jazyka, tím více je odvoditelných faktů a zároveň vyšší složitost odvozování.

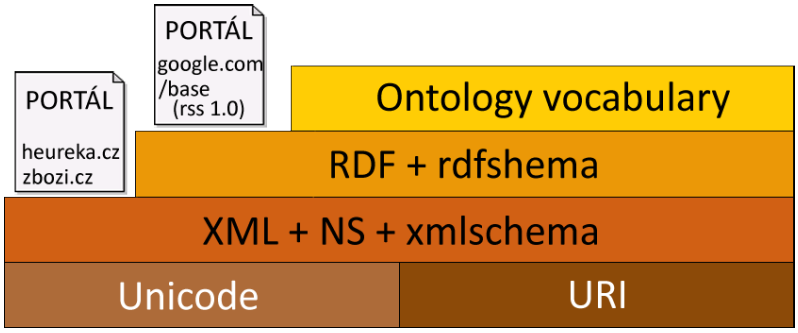
[6] STUCHLÍK, R. Ontologie a Sémantický Web [online]. Diplomová práce, Vysokém učení technické v Brně, Fakulta informačních technologií, 2007.

4. Semantický web

Na tomto místě je nutné se ještě zmínit o pojmu sémantický web. Ten totiž úzce souvisí s využitím metadat jako nástroje pro popis webových zdrojů. Termín sémantický web pochází od Timothy Berners-Leeho, který v roce 2001 prezentoval vize dalšího rozvoje internetu a s tím spojených úskalí.[7] Definoval ho jako možné rozšíření současného webu, kde informace jsou prezentovány dle standardizovaných pravidel a ve strojově čitelné formě.

Sémantický web lze také vnímat jako koncept, který využívá dalších technologií. Technologie používané v sémantickém webu a jejich vzájemné vztahy jsou patrny z obrázku 3. Jednotlivé vrstvy „koláče“ odpovídají úrovni sémantické síly dané technologie a tedy nejvýše postavenou formou metadat jsou ontologie.

Obr. 3: Koláč Timothy Berners-Leeho s technologiemi sémantického webu



Zdroj: Vlastní zpracování dle koláče Timothy Berners-Leeho

4.1 Agenti

Nástrojem automatizovaného zpracování informací v sémantickém webu mohou být tzv. agenti. Jedná se o inteligentní softwarová zařízení schopná vzájemné výměny dat. Využívají možností, které nabízí sémantický web a díky porozumění obsahu webových zdrojů dokáží vykonávat složité vyhledávací úlohy a nacházet souvislosti mezi zjištěnými informacemi.

[7] BERNERS-LEE, T. James Hendler and Ora Lassila (May 17, 2001). The Semantic Web. Scientific American Magazine, 2008.

5. Standardy

Hlavním účelem návrhu ontologií je zvýšit vzájemné porozumění mezi lidmi, zlepšení komunikace mezi počítačovými systémy a usnadnění návrhu aplikací orientovaných na znalosti. Aby tato myšlenka mohla být naplněna, je potřeba uznat nějaké standardy, které přesně a pravdivě určí význam pojmů. Jinými slovy tedy přijmout nějaké ontologie jako obecně platný rámec (formát) ke zvýšení interoperability na webu. Uznání standartu musí provést některá z respektovaných autorit. Může to být koncorcium W3C, společnost Microsoft, IBM či jiný uznávaný subjekt. Důležité však je, že výsledný formát bude akceptován širokou skupinou vývojářů a firem vyvíjejících webové aplikace.

V posledních letech se nejvíce prosadili tyto tři standardy:

- RSS (Really Simple Syndication, popř. RDF Site Summar)[8] – patří mezi nejrozšířenější metadatový formát, určený k syndikaci obsahu na webu. Vychází z obecného jazyka XML a existuje v několika modifikacích. Obsahuje přesnou definici jmenného prostoru, tak aby byl univerzální pro co možná nejširší množinu webových zdrojů. Umožňuje tak popsat abstrakty článků, novinky či zprávy z webových portálů a blogů. Informace mohou být prostřednictvím exportních souborů (RSS kanálů) koncentrovány na jednom místě, například na webovém portálu, nebo v osobní čtečce. Použití formátu RSS k popisu produktových dat není vhodné. Příliš obecná definice RSS jmenného prostoru by totiž zaváděla do produktového kanálu řadu nepřesností.

- FOAF (Friend of a Friend)[9] – populární ontologií sémantického webu je FOAF, která používá ke svému zápisu RDF a OWL. Popisuje osoby, subjekty, jejich aktivity a vzájemné vztahy.

[8] HARWARD, L. RSS 2.0 Specification [online]. RSS Advisory Board, 2009. URL:< http://www.rssboard.org/rss-specification>.
[9] BRICKLEY, D. ; MILLER, L. FOAF Vocabulary Specification 0.98 [online]. Namespace Document, 2010. URL:<http://xmlns.com/foaf/spec/>.

- DC (Dublin Core)[10] – je ontologie pro popis zdrojů jako jsou knihy, zvuk, obraz, nebo textové soubory či webové stránky. DC je zavedený standart, který se používá jak v oblasti knihovnictví, tak informatiky. Název je odvozen od města Dublin v Ohio (USA), ve kterém se konala konference, kde byl představen.

- SOAP (Simple Object Access Protocol)[11] – je protokol určený k výměně strukturovaných dat přes síť pomocí HTTP. Založen je na XML, který definuje rozšiřitelný pracovní rámec zpráv (značek, tagů). Tělo dokumentu tak může být tvořeno nezávisle na jazyku pro popis dat a jmenném prostoru.

[10] BACKER, T. ; DCMI Usage Board [online]. Dublin Core Metadata initiative, 2010. URL:< http://dublincore.org/>.
[11] GUDGIN, M.; HADLEY, M.; MANDELSOHN, N.; Latest SOAP versions, W3C Recommendation, 2007. URL: < http://www.w3.org/TR/soap/>

6. Formáty elektronických tržišť

Navržených formátů pro výměnu produktových dat v rámci elektronických tržišť je v dnešní době již nespočet a dá se říci, že každý portál má svůj vlastní formát. Obecně však využívají jazyka XML, jeho modifikací a nadstaveb. Formáty technologicky vyspělejší využívají definice jmenného prostoru. Často však opět platí, co agregační portál, to vlastní jmenný prostor. Přehled použitých XML značek (elementů, tagů) a formátů je patrný z tabulky 1. V úvahu jsou brány „pouze“ tři největší tuzemské portály; zbozi.cz, gogel.com/base/, a heureka.cz.

Tab. 1: Přehled používaných značek u 3 největších tuzemských portálů

	ELEMENTY		
PORTÁLY	zbozi.cz	heureka.cz	google.com/base/
1	PRODUCT	PRODUCT	title
2	PRODUCTNAME		
3	PRODUCTNAMEEXT		
4	DESCRIPTION	DESCRIPTION	description
5	URL	URL	link
6	IMGURL	IMGURL	image_link
7	PRICE	PRICE	
8	VAT	VAT	
9	PRICE_VAT	PRICE_VAT	price
10	DUES	DUES	
11	DELIVERY_DATE	DELIVERY_DATE	
12	SHOP_DEPOTS		
13	ITEM_TYPE	ITEM_TYPE	condition
14	TOLLFREE		
15	FIRMY_CZ		
16	MANUFACTURER	MANUFACTURER	manufacturer
17	CATEGORYTEXT	CATEGORYTEXT	product_type
18	EAN	EAN	gtin
19	PRODUCTNO		id (mpn)
20	VARIANT	PARAM_NAME	
21		VAL	
22		ISBN	gtin
23			quantity
24			
25			
FORMÁTY	XML	XML	RSS 2.0, + národní specifikace

Zdroj: Vlastní zpracování dle portálů zbozi.cz, heureka.cz, google.com/base/

Z obrázku je patrno ne příliš šťastné řešení, kdy k popisu téměř stejných dat jsou použity rozdílné technologie a elementy. Nastává tak problém vzájemné nepřenositelnosti mezi jednotlivými agregačními portály a tedy nutnost generovat pro každý portál speciální feed. Pokud by došlo k standardizaci používaných technologií a značek, tento problém by odpadal. Návrh možného řešení přináší následující kapitola číslo 7.

7. Tvorba doménového schématu (ABOX)

Jak bylo uvedeno výše, práce se zabývá návrhem optimálního formátu, určeného k agregaci dat v rámci B2B elektronických tržišť. Teoretické podklady k výběru vhodné technologie byly popsány výše. Nyní tedy lze přistoupit k popis vlastního návrhu.

Nejprve musejí být stanovena kritéria, která by měl takový optimální formát splňovat. Na základě stanovených podmínek bude pak vybrána vhodná technologie a v konečné fázi samotný návrh formátu.

7.1 Kritéria optimálního formátu a výběr technologie

Jednoduchou úvahou dojdeme k závěru, že mezi ty nejdůležitější kritéria optimálního formátu patří:

- technologická vyspělost (výběr vhodného jazyka pro popis dat);
- snadná rozšiřitelnost;
- zachování informační bohatosti, kompatibility se stávajícími formáty (v celé práci jsou uvažovány - podporovány formáty 3 největších tuzemských tržišť: zbozi.cz, google.com/base, heureka.cz).

Výběr vhodné technologie pro popis dat je snad tím nejdůležitějším ze všech kritérií a rozhodující měrou určuje budoucnost navrhovaného formátu. Agregační portály by totiž jen stěží přecházeli na formát, který by byl za pár let nahrazen modernější variantou. Formát tedy musí být svým způsobem nadčasový s potenciálem jeho možného přijmutí jako standardu. Uvážíme-li, že ne všechny jazyky pro popis dat mají stejnou sémantickou sílu (viz kapitola 4) a vývoj internetu se bude nejspíš uchylovat k sémantickému webu, jeví se jako nejvhodnější použití jazyka RDF s deklarací ontologie v jazyce OWL FULL. Od této technologie lze také odvodit název výsledného formátu a to ontoprodukt s označením RDF_OP.

Ze základních podmínek tvorby ontologie plyne, že musí umožňovat definici nových pojmů, založených na existujícím slovníku bez nutnosti revize stávajících definic.[12] Další kritérium optimálního formátu - snadná rozšiřitelnost - je tedy také splněno.

[12] HAŠLER, P. Experimentální systémy pro sémantický web. Diplomová práce, České vysoké učení technické v Praze, Fakulta elektrotechnická, 2006.

Splnění poslední podmínky - zachování informační bohatosti, kompatibility se stávajícími formáty - je nejsložitější. Nelze totiž předpokládat, že obecný rámec (ontologie) by obsahoval(a) definice lokálních (rozšiřujících) elementů jako je např. element FIRMY_CZ pro portál zbozi.cz (viz tabulka 1). Naštěstí však každý z uvažovaných portálů definuje dva druhy elementů a poslední kritérium tak bude splněno alespoň z části (podrobněji viz kapitola 7.5.2).

Na tomto místě se nabízí otázka, vytvářet vlastní ontologii nebo použít některou z již dříve vytvořených (např. GoodRelations). Uvážíme-li, že neexistuje ontologie, která využívá stejného členění kategorií jako portál google.com/base a jednou z podmínek optimálního formátu je zpětná kompatibilita, tak nezbývá než tvorba vlastní ontologie.

7.2 Protégé

Návrh ontologií lze teoreticky vytvářet v běžném editoru textu. Tento způsob však není moc efektivní a nenabízí komplexní představu o návrhu. K mnohem účelnějšímu způsobu vytváření a editace ontologií dojde použitím specializovaného softwaru. Nejznámějšími jsou komerční OntoEdit[13] či Ontosaurus a volně stažitelný open source program Protégé.

V práci je využito posledně jmenovaného programu Protégé ve verzi 3.3.2. Tento systém umožňuje návrh, editaci a export ontologií do formátů RDF(S), OWL a XMLSchema. Vyvinut byl v institutu Standford Center for Biomedical Informatics pod vedením M. Musena. Vývoj programu stále pokračuje a jelikož se jedná o bezplatný produkt, který se funkčně nijak výrazně neliší od komerčních programů, je jeho výběr zcela pochopitelný.

7.3 Postup návrhu ontologie

Každá ontologie se týká specifické domény, tj. popisované oblasti. Modeluje koncepty, které jsou abstraktní nebo konkrétní, obecné nebo specifické. Je pouze na tvůrci ontologie, jak danou doménu bude vnímat a jak s ní bude pracovat. Tvorbu ontologie lze rozdělit do několika kroků:

- modelování taxonomie tříd (viz kapitola 7.4);
- deklarace vlastností (viz kapitola 7.5);
- nastavení globálních a lokálních omezení (viz kapitola 7.6);

[13] SURE, Y., ERDMAN, M., STUDER, R.: OntoEdit: Collaborative Engineering of Ontologies, Semantic Web enabled Knowledge Management. J. Davies, D. Fensel, F. van Harmelen (eds.), 2002.

- vymezení popsaných a definovaných tříd (viz kapitola 7.7).

Navržená ontologie tedy bude obsahovat třídy, podtřídy a jejich vzájemné vztahy(vlastnosti). Doporučuje se také[14] dodržovat určitá pravidla zápisu. Názvy tříd by měly začínat velkými písmeny, vlastnosti malými a složené názvy psát bez mezer s tím, že další nové slovo bude začínat velkým počátečním písmenem. Jazykem ontologie by měla být angličtina a to z důvodů mezinárodní srozumitelnosti. Nemusí to však být pravidlem.

7.4 Modelování tříd

Pojem třída je v jazyce OWL FULL chápán stejně jako v objektově orientovaném programování (OOP). Sdružuje skupinu jedinců se stejnými vlastnostmi. Jazyk OWL nabízí použití předdefinovaných, nebo námi vytvořených tříd, mezi kterými lze definovat taxonomii.[15] Předdefinovanou třídou je **owl:Thing.** Jedná se o jakéhosi předka všech námi vytvořených tříd. Označována je také jako systémově nejobecnější třída.

Vlastní tvorbě tříd musí předcházet analýza popisované oblasti. Je třeba si uvědomit na jaké třídy lze oblast rozdělit, a které třídy jsou obecnější a které specifičtější. Jelikož žádnou doménu nelze úplně komplexně formálně popsat (nelze zaznamenat všechny vlastnosti a omezení), tak je důležité si určit správný rozsah, nad kterým bude prováděna konceptualizace dané oblasti.

V případě popisu produktových dat je doména omezena na třídy a vlastnosti nezbytné k výměně dat v rámci B2B elektronických tržišť. Popsány jsou vzájemné vztahy mezi prodejcem , zákazníkem, prodávaným zbožím a způsobem jeho doručení. Jedná se tedy o model části reálného obchodního styku orientovaný na nabídku zboží koncovým zákazníkům.

Jak je patrné z obrázku 4, oblast je možné rozdělit do osmi hlavních tříd. Lze vycházet z jednoduché úvahy. Výrobce (třída Producer) vyrábí zboží (třída Goods), které lze zařadit do určité kategorie (třída Category). Zboží má svého spotřebitele (třída Consumer) a v nějaké kvalitě (Quality) je nabízeno prodejcem (třída Seller). Každé zboží a prodejce lze indenifikovat pomocí jedinečné URL adresy (třída URL) a je doručováno nějakou společností (Deliverer).

[14] HUSÁKOVÁ, M. Znalostní technologie I [online]. Osobní stránky a stránky pro podporu výuky na Univerzitě Hradec Králové, Fakulta informatiky a managementu, 2008. Kapitola 4.: představení jazyka OWL. URL:<http://lide.uhk.cz/fim/ucitel/fshusam2/lekarnicky/zt1/zt1_kap04_02.html>.
[15] HUSÁKOVÁ, M, c.d, Kapitola 4.: představení jazyka OWL.

Dále je také nutné zdůraznit, že mezi třídami lze definovat disjunktnosti, jež odrážejí jejich odlišnosti v ontologii. Disjunktností říkáme, že neexistuje jedinec patřící více třídám. Vždy musí spadat pouze do jedné z nich - takové třídy nemají žádné společné prvky.

Vzájemné disjunktnosti musí být nastaveny například mezi potomky třídy Quality tj. třída Bazaar, New a Refurbished. Nabízený produkt totiž nemůže být zároveň nový, použitý a renovovaný.

Obr. 4: Hlavní třídy ontologie

Zdroj: Program Protégé verze 3.3.1

7.5 Deklarace vlastností

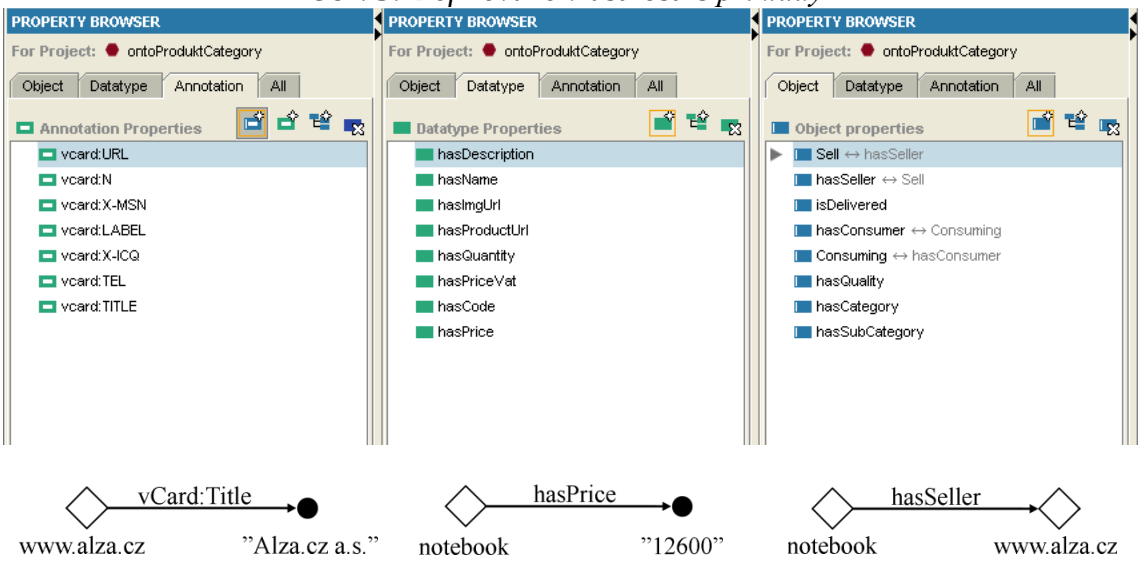
V ontologii vystupuje také řada vlastností. Vlastnost jako taková umožňuje vystihnout význam třídy, popsat ji nebo definovat. Jazyk OWL poskytuje několik skupin vlastností:

- anotační,
- datotypová,
- objektová,
- ontologická.

Anotační vlastnost lze použít k přidaní další informace třídám, jedincům, nebo celé ontologii ve formě metadat.[16] Datotypová vlastnost spojuje jedince s hodnotou, která je určitého datového typu. Objektová vlastnost je určena pro vytváření spojení mezi objekty tříd (jedinci). Aby bylo možné popsat vztahy mezi jednotlivými třídami v produktové ontologii, byly využity vlastnosti anotační, datotypové a objektové (viz. obrázek 5).

[16] HUSÁKOVÁ, M, c.d, Kapitola 4.: představení jazyka OWL.

Obr. 5: Definované vlastnosti s příklady



Zdroj: Program Protégé verze 3.3.1
s vlastním zpracováním příkladů

7.5.1 Anotační vlastnosti (informace o společnosti)

Jak bylo uvedeno v kapitole 7.4 v ontologii je využito anotačních vlastností. V návrhu slouží k popisu informací o společnosti a využívají předdefinovaného standartu vCard. Konkrétně pak modifikaci vCard Objects in RDF určenou k popisu osob a společností v kódování RDF[17]. Jedná se o velice rozšířený a zaběhlý formát, obsahující mnoho tříd. Produktová ontologie využívá 7 z nich (viz. obrázek 5)

7.5.2 Datotypové vlastnosti (informace o produktech)

Datotypové vlastnosti jsou v ontologii použity k popisu produktů. Jejich smyslem je propojení jedince z nějaké třídy s hodnotou určitého datového typu. Podobnou funkci zastávají i elementy z používaných XML formátů u elektronických tržišť (viz tabulka 2). Uvážíme-li také, že ontologie musí zaručovat zpětnou kompatibilitu formátů (viz kapitola 7), nabízí se možnost „ztotožnění“ používaných elementů s datotypovými vlastnostmi třídy.

Jak bylo uvedenu v úvodu 7. kapitoly, elementy lze rozdělit na dva druhy. První skupinou jsou nutné elementy, bez kterých by nebylo možné produkt zařadit do databáze. Druhou skupinou jsou rozšiřující (lokální) elementy, které přidávají doplňující informace o produktu a prodejci (viz tabulka 2). Jelikož nelze předpokládat, že obecný rámec (ontologie) by obsahoval(a) definice všech lokálních elementů (viz kapitola 7),

[17] HARRY, H., RENATO, I., BRIAN, S., NORMAN, W., Representing vCard Objects in RDF [online]. W3C Member Submission, 2010. URL:< http://www.w3.org/Submission/vcard-rdf/>

ztotožnění vlastností bude provedeno pouze s elementy z první skupiny. Výsledkem je soubor datotypových vlastností zachovávající vzájemnou kompatibilitu formátů.

Tab. 2: Přehled značek 3 největších tuzemských portálů a odpovídající ekvivalenty navrženého formátu ontoprodukt (elementy napsané kurzívou jsou u daného formátu nutnými)

	ELEMENTY			
PORTÁLY	zbozi.cz	heureka.cz	google.com/base/	ontoprodukt
1	***PRODUCT***	***PRODUCT***	***title***	***hasName***
2	PRODUCTNAME			
3	PRODUCTNAMEEXT			
4	***DESCRIPTION***	***DESCRIPTION***	***description***	***hasDescription***
5	***URL***	***URL***	***link***	***hasProductUrl***
6	IMGURL	***IMGURL***	image_link	***hasImgUrl***
7	PRICE	PRICE		***hasPrice***
8	VAT	VAT		
9	***PRICE_VAT***	***PRICE_VAT***	**price**	***hasPriceVat***
10	***DUES***	DUES		***hasDues***
11	DELIVERY_DATE	***DELIVERY_DATE***		***hasDeliverDate***
12	SHOP_DEPOTS			
13	ITEM_TYPE	***ITEM_TYPE***	**condition**	***hasQuality***
14	TOLLFREE			
15	FIRMY_CZ			
16	MANUFACTURER	MANUFACTURER	manufacturer	***hasProducer***
17	CATEGORYTEXT	***CATEGORYTEXT***	***product_type***	***hasCategory, hasSubCategory***
18	EAN	EAN	gtin	hasCode
19	PRODUCTNO		***id*** (mpn)	***hasId*** **(auto.generated)**
20	VARIANT	PARAM_NAME		
21		VAL		
22		ISBN	gtin	
23			quantity	
24				isDelivered
25				hasConsumer
FORMÁTY	XML	XML	RSS 2.0, + národní specifikace	RDF specifikace + vCard specifikace

Zdroj: Vlastní zpracování dle portálů zbozi.cz, heureka.cz, google.com/base/

7.5.3 Objektové vlastnosti (doplňující informace)

Poslední skupinou jsou objektové vlastnosti. Ty slouží k doplnění a ucelení vztahů mezi instancemi tříd (jedinci). Do struktury zavádějí nové informace, které jsou nezbytně nutné ke komplexnímu popisu ontologie (viz obrázek 5).

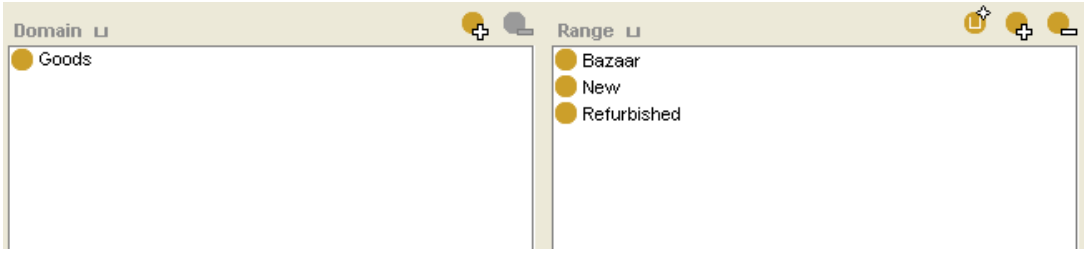
7.6 Omezení

Omezení jsou podmínky využívající se pro bližší upřesnění vlastností tříd. Případné aplikace, která bude s ontologií pracovat říkají, co si má pod třídou představit a jaký je její význam, sémantika. Omezení mohou být buď globální nebo lokální. [18]

7.6.1 Globální omezení vlastnosti

U nadefinovaných vlastností je dobré specifikovat tzv. globální omezení tj. definiční obor a obor hodnot vlastnosti.[19] Vytvoří se tak množina tříd, respektive jedinců, mezi kterými může vlastnost existovat. Definiční obor určuje, u jaké třídy se může daná vlastnost vyskytnout a obor hodnot definuje jaké hodnoty vlastnost nabývá a ze které třídy pochází. Výsledkem je jednoznačné vymezení role vlastností a tedy i zabránění nekonzistence ontologie. Globální omezení vlastnosti hasQuality je patrno z obrázku 6.

Obr. 6: Globální omezení vlastnosti hasQuality



Zdroj: Program Protégé verze 3.3.1

7.6.2 Lokální omezení vlastnosti

Tam, kde chceme vystihnout sémantiku (význam) třídy použijeme lokální omezení vlastnosti. Mezi lokální omezení patří:

- existenční omezení vlastnosti (symbol ∃),
- kvantitativní omezení vlastnosti:
  - minimum (symbol ≤)
  - maximum (symbol ≥)
  - rovnost (symbol =),
- hasValue omezení (symbol ∋).[20]

Jak je vidět z obrázku 8, tak třída Goods je popsána kvantitativním a existenčním omezením. Mezi jednotlivými omezeními platí logická spojka AND.

[18] MAREŠ,V.; NOVÁK,O. AirCraft ontologie, Semesterální práce z TSW, Technická Univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, 2010.
[19] HUSÁKOVÁ, M, c.d, Kapitola 5.: Tvorba OWL ontologie krok za krokem.
[20] HUSÁKOVÁ, M, c.d, Kapitola 6.: Reprezentace sémantiky tříd (1).

7.7 Popsané a definované třídy

V prostředí Protégé se v závislosti na tom, kde jsou lokální omezení uvedena, dělí třídy do několika skupin.

Popsaná třída je třída, která obsahuje omezení uvedená jen v sekci NECESSARY. Tato omezení slouží pro popis třídy a říkají jaká omezení jsou nezbytně nutná pro to, aby bylo možné jedince do třídy zařadit.

Definovaná třída je taková třída, která obsahuje omezení uvedená v sekci NECESSARY & SUFICIENT. Tyto podmínky jsou nezbytnými a zároveň postačujícími k tomu, aby jedinec mohl být považován za člena definované třídy.

Z výše uvedeného plyne že, zboží je zbožím, pokud jsou splněna všechna předdefinovaná omezení umístěná v sekci NECESSARY (viz obrázek 7).

Obr. 7: Popsaná třída Goods – sekce NECESSARY

Zdroj: Program Protégé verze 3.3.1

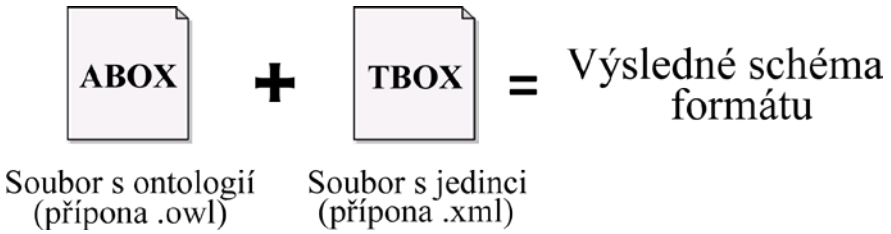
7.8 Začlenění ontologie do formátu

Začlenění ontologie do formátu je umožněno prostřednictvím jmenných prostorů XML. Jak bylo uvedeno (viz kapitola 3.3) každý jmenný prostor musí mít vlastní jednoznačný identifikátor v podobě URI. URI většinou odkazuje na místo, kde je uložena deklarace jmenného prostoru (klasifikační schéma).[21] V případě mnou navrhovaného formátu se jedná o soubor s deklarací ontologie umístěným v ontologickém registru MMI Ontology Registry and Repository s URI adresou: „http://mmisw.org/ont/ontoProdukt/op1.owl“.

[21] MATULÍK, P., PITNER, T. Sémantický web a jeho technologie. Zpravodaj ÚVT MU. ISSN 1212-0901, 2004, roč. XIV, č. 3, s. 15-17. URL: <http://www.ics.muni.cz/zpravodaj/articles/296.html>.

Výsledné schéma formátu se skládá ze dvou soborů (viz obrázek 8). První z nich obsahuje deklaraci ontologie v jazyce OWL FULL a druhý soubor zahrnuje instance tříd s konkrétními jedinci. Navržený formát v takovéto podobě (ABOX + TBOX) obsahuje přesnou definici pojmů použitých k popisu produktových dat a je tedy umožněno jeho strojové čtení a zpracování. Formát tak může být použit jak k výměně dat v rámci B2B elektronických tržišť, tak i k poskytování informací o nabízených produktech sémanticky orientovaným vyhledávačům. Jeho uplatnění je tedy oproti dosavadním formátům mnohem širší a odbourává problémy spojené s hodně obecným vymezením pojmů.

Obr. 8: Výsledné schéma formátu



Zdroj: Vlastní zpracování

8. Tvorba báze znalostí (TBOX)

V předchozí kapitole byla popsána nejoptimálnější podoba formátu. Nyní tedy ještě zbývá navrhnout vhodný způsob jeho implementace. Potenciální uživatele lze rozdělit do dvou skupiny; na ty, kteří vlastní e-shop s generátorem a ty bez generátoru některého z podporovaných kanálů. Aplikace by tedy měla brát ohled na obě dvě skupiny uživatelů a umožňovat jak automatickou (e-shopy s generátorem) tak manuální konstrukci dokumentu (bez generátoru).

Manuální konstrukce dokumentu bude velmi zdlouhavá a pro e-shopy, které nemají vlastní generátor jediná možná. Odpadá však nutnost tvorby dokumentu v textovém editoru a tedy i potřeba důsledné kontroly správné syntaxe jazyka.

Naproti tomu automatické generování (převod) dokumentu bude mnohem rychlejší a pohodlnější. To je také hlavní důvod, proč byl kladen velký důraz na vzájemnou kompatibilitu mezi nově navrženým formátem a formáty třech největších tuzemských elektronických tržišť.

8.1 Princip generování feedu

Knihovna je navržena tak, aby ji bylo možné implementovat do co možná nejširší skupiny aplikací. Programátorovi je umožněno využít jednotlivé třídy či celou knihovnu jako celek. Knihovna je napsána v jazyce PHP verze 5.0 za použití objektového programování. Její základ tvoří 4 třídy, includované soubory a specifikace ontologie. Celý princip generování feedu je možné popsat několika níže uvedenými kroky (viz tabulka 3). Upozorňuji, že se jedná pouze o velice hrubý popis činnosti a jednotlivé kroky s jim příslušejícími třídami budou podrobněji popsány v kapitolách 8.1.1; 8.1.2; 8.1.3.

Tab. 3: Jednotlivé kroky generování feedu a jim příslušející třídy

Krok	**Odpovídající třídy a soubory**
Výběr vhodného způsobu generování a formátu feedu	***example/index.php***
Získání dat	***generator.class.php*** ***parser.class.php*** ***viewer.class.php***
Vygenerování výsledného souboru	***newFeed.class.php***

Zdroj: Vlastní zpracování

8.1.1 Výběr způsobu generování a formát feedu

Uživatel musí nejprve vybrat způsob generování feedu. Tak jak bylo napsáno výše, správným předpokladem je možnost výběru z manuální a automatické volby. V případě, že uživatel již feed někdy v minulosti generoval, může ho vybrat zadáním URL adresy nebo výběrem souboru. Aplikace podporuje generování formátů, které jsou dle mého názoru v České republice nejrozšířenější. Jsou to formáty portálu zbozi.cz, heureka.cz, google.com/base a také samozřejmě mnou navržený formát ontoprodukt. Tato část programu je implementována v souboru ***example/index.php*** a slouží jako ilustrativní příklad možného použití knihovny. Ta je tvořena jedním formulářem, několika vstupními poli a tlačítkem pro odeslání formuláře.

8.1.2 Získání dat

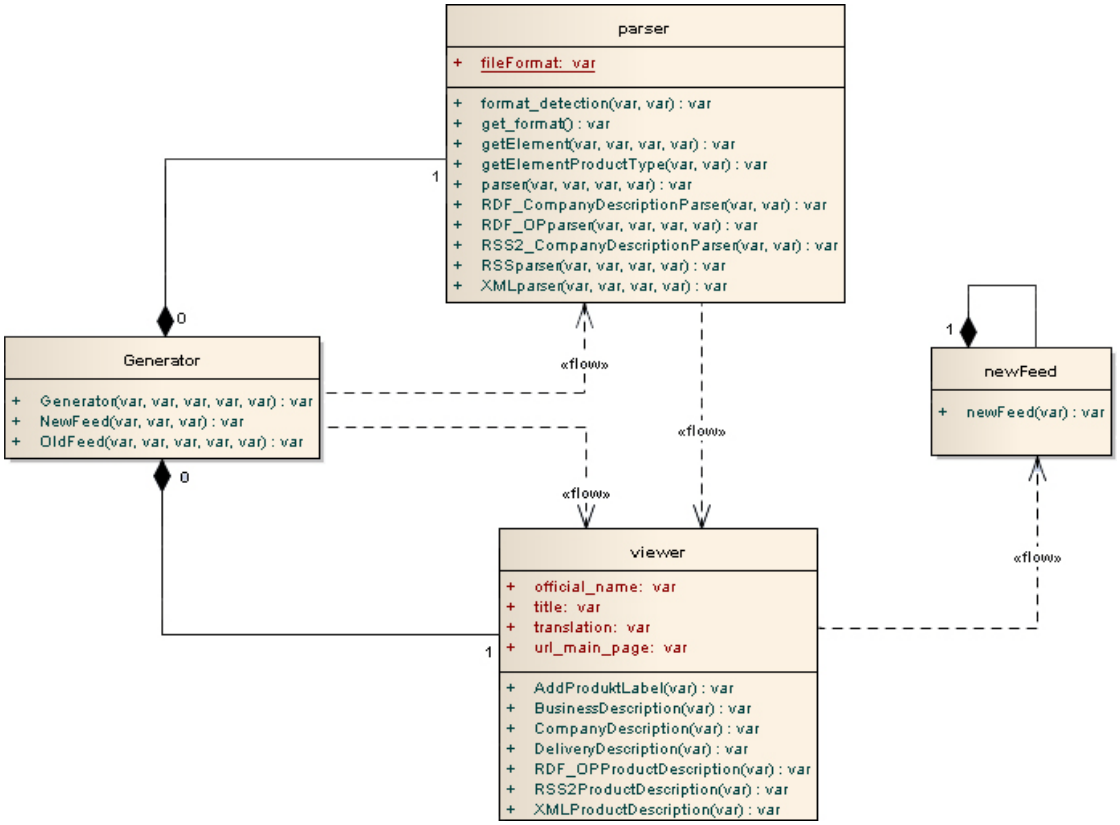
Dalším krokem při generování kanálu je získání vstupních dat, tedy buď zpracování již existujícího kanálu a zobrazení, nebo vyplnění předdefinovaného formuláře s produkty. V případě zpracovávání existujícího kanálu je využíváno třídy ***parser.class.php*** (podrobněji viz kapitola 8.3)**.** Zobrazení dat a vstupních polí zabezpečuje třída ***viewer.class.php*** (podrobněji viz kapitola 8.4)**.**

8.1.3 Vygenerování výsledného souboru

Poslední a nejdůležitější fází generování výstupního souboru je samotná konstrukce dokumentu. Předaná data ze vstupních formulářů jsou postupně začleňována do struktury dokumentu a výsledkem je kanál (soubor) s požadovanou strukturou a obsahem. Ten lze samozřejmě stáhnou a umístit na stránky webu popř. nahrát do produktových portálů. Výše popsanou činnost umožňuje použití třídy ***newFeed.class.php*** (podrobněji viz kapitola 8.5)**.**

Názornější pochopení fungování jednotlivých tříd a jejich vzájemného propojení je patrné z tabulky 3 a UML-class diagramu (viz obrázek 9).

Obr. 9: UML - class diagram knihovny



Zdroj: Program Enterprise Architect

8.2 Třída Generator

Základem celé knihovny je třída ***Generator.class.php***, která obaluje všechny funkce a podtřídy potřebné ke správné funkci skriptu. Zabezpečuje vytváření objektů a předávání dat ve správném pořadí.

Aby bylo možné vytvořit nový objekt třídy Generator, musí dojít k importu některých funkčních komponent ze separátních souborů. V nich jsou umístěny kusy kódu, které je nutné častěji vkládat do aplikací. Použit je příkaz require_once(), který provede import souboru s nastavením knihovny (configuration.php).

Následné vytvoření objektu lze vyvolat pomocí operátoru new, za kterým následuje název třídy a její parametry. V tomto případě jsou to tyto 3:

1) $generatedFormat – výsledný formát,
2) $translation – texty popisků formuláře,
3) $new_feed – vytvoření nového feedu, nebo úprava starého.

Samotná struktura třídy je tvořena konstruktorem a definicemi jednotlivých členských funkcí. Konstruktor při svém volání vytváří nový objekt třídy viewer a dále

dle předaného parametru $new_feed volá členské funkce (metody objektu) **NewFeed()** či **OldFeed()**. Obě metody jsou samozřejmě volány s příslušnými parametry a to nově vytvořeným objektem viewer, výsledným formátem, překladem textů a v případě funkce OldFeed také URL adresou feedu.

NewFeed() – volá členské funkce třídy objektu viewer, které zobrazují vstupní pole pro zadání informací o společnosti, produktech a doručení.

OldFeed() – zabezpečuje získání produktových dat z externího XML souboru, který může být načten z lokálního disku či absolutní URL adresy. Zároveň také předaný soubor dle umístění načte (viz. obrázek 10) a voláním třídy ***parser.class.php*** zpracuje. Získané informace následně zobrazí vytvořením instance třídy ***viewer.class.php*** a svoji činnost ukončí.

Obr. 10: Příklad načtení souboru z lokálního disku

```
if (file_exists($_FILES["file"]["tmp_name"]))
{    $fp = fopen($_FILES["file"]["tmp_name"], "r");
     $xml = fread($fp, $_FILES["file"]["size"]);
     $xml = stripslashes($xml);
     fclose($fp);
     unlink($_FILES["file"]["tmp_name"]);      }
else{echo "Došlo k chybě při přenosu souboru!";}
```

Zdroj: Vlastní zpracování

8.3 Třída parser

Třída ***parser.class.php*** jak už název napovídá, slouží ke zpracování XML dokumentů. Využívá DOM API metod implementovaných v rozšíření PHP 5. DOM (Document Object Model) je standart W3C pro reprezentaci XML nebo HTML dokumentů v podobě objektového stromu.[22] Definuje logickou strukturu, způsob přístupu a manipulaci s dokumenty.[23] Programátor tak má k dispozici velmi efektivní nástroj pro vytváření XML, HTML dokumentů , změnu struktury souboru či mazání nebo přidávání jak elementů, tak obsahu. Dokument je reprezentován pomocí uzlů. Nejvýznamnějšími jsou elementy, atributy, komentáře a texty. Rozšíření je součástí jádra PHP a tudíž nepotřebuje žádnou dodatečnou instalaci.

[22] GILMOR, W.Jason. Velká kniha PHP a MySQL 5 – kompendium znalostí pro začátečníky i profesionály, 2.vydání. Brno : Zoner Press, 2007. ISBN 80-86815-53-6

[23] OCTAVIA A. A. Parsing XML with the DOM Extension for PHP 5 [online]. PHP Builder, 2010. URL: <http://www.phpbuilder.com/columns/DOM-XML-extension/Octavia_Anghel102710.php3>

Navržená třída je volána se 4 parametry:

- dokumentem určeným ke zpracování,
- objektem viewer,
- texty popisků vstupních formulářů,
- výsledným formátem.

Konstruktor třídy nejprve vygeneruje nový DOMDocument, do kterého načte obsah předávaného dokumentu a vytvoří asociativní stromovou DOM reprezentaci v podobě objektů. Aby bylo možné začít s parsrováním dokumentu, je nutné zjistit jeho strukturu (formát). K tomu slouží metoda **format_detection()** s návratovou hodnotou zjištěného formátu. Možnými typy jsou XML pro portál zbozi.cz a heureka.cz, RSS pro google.com/base nebo mnou navržený formát RDF_OP (ontoProdukt). Jiné formáty nejsou podporovány. Metoda zpracováním root elementu zjistí o jaký formát se jedná a na základě zjištěného typu volá odpovídající metody (viz tabulka 4) pro zpracování obsahu dokumentu.

Tab. 4: Volání metod dle zjištěného formátu

Formát	Metody
XML	**XMLparser()**
RSS	**RSS2_CompanyDescriptionParser()** , **RSSparser()**
RDF_OP	**RDF_CompanyDescriptionParser(), RDF_OPparser()**

Zdroj: Vlastní zpracování

Formáty RSS a RDF_OP obsahují také informace o společnosti; proto je nutné nejprve volat **RDF_CompanyDescriptionParser()** popř. **RSS2_Company - DescriptionParser()** pro jejich zpracování a až poté metodu pro zpracování vlastního obsahu dokumentu.

RSS2_CompanyDescriptionParser() – je členskou funkcí třídy parser se dvěma parametry. Prvním je stromová DOM reprezentace zpracovávaného dokumentu a druhým parametrem je objekt viewer pro zobrazení získaných informací. Na předávaný dokument je aplikována DOM metoda getElementsByTagName(), která přistupuje ke stromové struktuře dokumentu a postupně získává obsahy jednotlivých elementů. Následuje úprava nově získané hodnoty pomocí funkcí pro práci s řetězci preg_split(), trim() a předání do členské proměnné objektu viewer pro zobrazení.

RDF_CompanyDescriptionParser() – taktéž se jedná o členskou funkci třídy parser pracující obdobě jako předchozí funkce. Liší se však zpracovávanou strukturou dokumentu. Formát RDF obsahuje oproti RSS navíc informace o adrese společnosti, které je nutné samozřejmě zpracovat (DOM metoda getElementsByTagName()). Jedná se však o složitější operaci. Jelikož získaná hodnota adresy je v podobě textového řetězce, musí dojít k použití funkce preg_split() s oddělovačem v podobě „;“ pro její rozdělení. Dojde tak k vytvoření asociativního pole hodnot s jednotlivými částmi adresy společnosti, které pak už není problém společně s ostatními daty předat objektu viewer pro zobrazení.

Hlavní část dokumentu bez rozdílu formátů obsahuje informace o produktech (viz. tabulka 2) jako je cena, název produktu, nebo třeba kategorie, ve které je zboží zařazeno. Zpracování těchto dat se nijak výrazně neliší od postupů popsaných v členských funkcích pro zpracování informací o společnosti**.** Využito je opět DOM metody getElementsByTagName() k získání hodnot elementů a funkce pres_split() s oddělovačem v podobě „<“ pro zpracování kategorie, ve které je produkt zařazen. Zařazení produktu je taktéž vyjádřeno textovým řetězcem a skládá se z kategorie, subkategorie a oddělovače.

V případě formátu RDF_OP použití preg_split() odpadá. Kategorie a subkategorie nejsou totiž vyjádřeny v jednom řetězci, ale každá má svůj popisový element. Název kategorie a subkategori musí být na závěr ještě upraven pomocí funkce **getElementProductType(),** která provádí nahrazení některých znaků v řetězci (viz obrázek 11). Veškerou výše popsanou činnost mají na starosti třídy **XMLparser()** a **RSSparser()**.

Obr. 11: Ukázka členské funkce getElementProductType() pro nahrazení znaků v řetězci

```
function getElementProductType($viewer, $product_type)
     {
        $numbers = array("0", "1", "2", "3", "4", "5", "6", "7", "8",
        "9"); $product_type[0] = preg_replace("/And/", "&",
        trim(Str_Replace ($numbers, "", Str_Replace ("#", "",
        $product_type[0]))));
        $product_type[0] = preg_replace("/_/", " ", $product_type[0]);
        $viewer -> product_type[0] = $product_type[0];
        $product_type[1] = trim(Str_Replace ($numbers, "", Str_Replace
        ("#", "", $product_type[1])));
```

```
  $product_type[1] = preg_replace("/_/", " ",
  preg_replace("/And/", "&", $product_type[1]));
  $viewer -> product_type[1] = $product_type[1];
}
```

Zdroj: Vlastní zpracování

Produkty jsou v dokumentu řazeny za sebou, proto musí být vždy vykonány tyto kroky:

- zjištění root elementu produktu,
- načtení obsahu,
- zpracování a předání hodnoty do členské proměnné třídy viewer.

Celý postup se v cyklech takto opakuje až do zpracování posledního produktu.

8.4 Třída viewer

Grafické rozhraní mezi uživatelem a knihovnou zabezpečuje třída viewer. Zobrazuje vstupní formulář a řídící tlačítka pro přidání produktu a generování kanálu. Formulář je rozdělen na 4 části.

- CompanyDescription,
- ProdukDescription,
- BusinessDescription,
- DeliveryDescription.

Názvy částí implikují svoji funkci a každá z nich odpovídá jedné metodě. Jejich volání zobrazuje příslušné části formuláře s řadou vstupních polí (inputů). V případě, že byla předána data z třídy parser.class.php, jsou načtena a zobrazena společně s formulářem jako obsah vstupních polí. V opačném případě dojde k inputů bez obsahu.

Správnost zadávaných hodnot do vstupních polí je již v průběhu vyplňování zabezpečena validátorem, který kontroluje, zda se zadaný řetězec shoduje s regulárním výrazem nastaveným pro vyplňovaný input. Při zadání jiného obsahu, než dovoluje regulární výraz, dojde k vypsání varovné hlášky a formulář nelze odeslat. Validátor je vytvořen pomocí pokročilejších programátorských technik, za použití javascriptovského frameworku jQuery.

Mezi jeho přednosti patří výše nastíněný přístup k jednotlivým prvkům HTML stránky a tedy i dynamické chování formuláře. Tento framework využívá tzv. selektorů (Viz obrázek 12), které jsou schopny identifikovat jakýkoliv prvek na stránce.

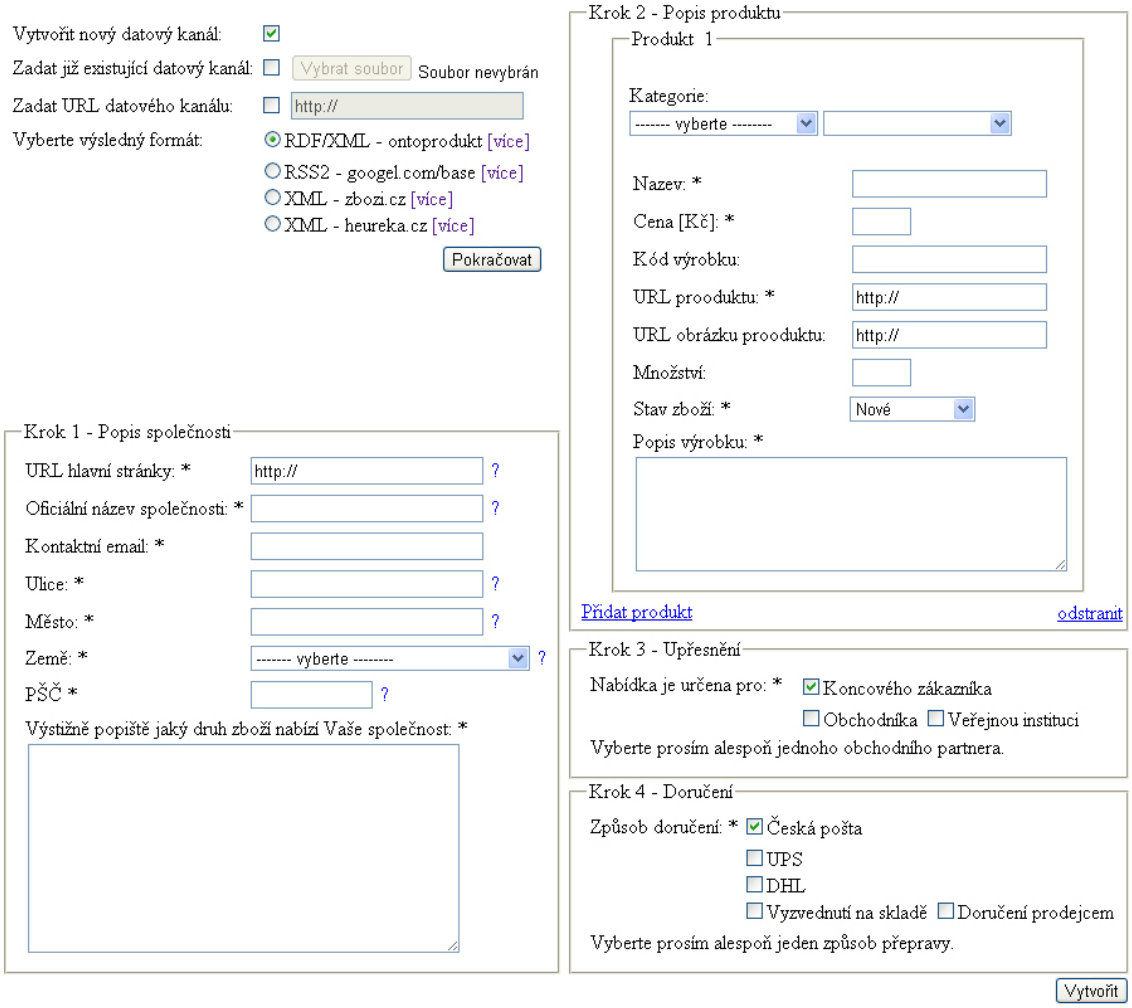
Obr. 12: Příklad vstupního pole pro e-mail společnosti se selektorem cemail

```
<input type='text' name='companyEmail' id='cemail' size='25'
class='required email' value='$this->companyEmail'>
```

Zdroj: Vlastní zpracování

Dalším dynamicky se měnícím prvkem ve stránce je část určená k popisu produktu. V průběhu vyplňování formuláře (viz obrázek 13) lze totiž pomocí tlačítek „přidat produkt“ nebo „odstranit produkt“ dynamicky měnit jednotlivé počty vyplňovaných produktů. Tato činnost je obstarávána voláním funkcí **addProdukt()** a **RemoveProdukt()** umístěných v javascriptovském souboru **setProdukt.js.**

Obr. 13: Vstupní formuláře



Zdroj: Vlastní zpracování

Po vyplnění všech požadovaných údajů lze data odeslat ke zpracování. Použito je metody POST, která předávaná data posílá v těle dotazu. Pokud tak uživatel učiní, je následně přesměrován na stránku **newFeed.class.php** obsahující vygenerovaný feed. Podrobný popis této stránky viz následující kapitola 8.5.

8.5 Třída newFeed

Jedná se o samogenerující se třídu, která vytváří novou instanci třídy vlastním voláním. V podstatě tak při každém spuštění stránky dochází k vytvoření nového objektu. Opět je využito již jednou zmiňovaných DOM API metod. Tentokrát však ne pro zpracování dokumentu, ale k jeho konstrukci. Nejdůležitější DOM API metody, jsou uvedeny v tabulce 5.

Tab. 5: Nejpoužívanější DOM API metody

DOM API Metoda	Činnost
createElement()	Vytvoření elementu
appendChild()	Vytvoření potomka rodičovského elementu
createAttribute()	Vytvoření atributu elementu
createTextNode()	Vytvoření obsahu elementu

Zdroj: Vlastní zpracování

Třída při svém volání nejprve vytvoří hlavičku dokumentu a poté dle výsledného formátu zavolá metodu pro sestavení dokumentu. Všechny volané metody pracují obdobně a jejich činnost lze rozdělit do těchto kroků:

1) načtení proměnných (soubory **set_POST_products_variable.php** a **set_POST_company_variable.php**)
2) vytvoření root elementu (**createElement()**)
3) cyklické vytváření položek produktů (metoda **newXMLfeed(), newRSS2feed(), newRDF_Opfeed()**)
4) zobrazení výsledného dokumentu

Načítání proměnných netřeba nijak složitě vysvětlovat. Jedná se o klasické zpracování předávaných dat pomocí metody POST. Vytvoření root elementu je též velice jednoduché. Postačuje volání DOM metody **createElement().**

Cyklické generování položek produktů je poněkud složitější. U formátu RDF je generovaný element složen z jmenného prostoru, názvu, atributu a vlastního obsahu

elementu. Opět musí být použito předdefinovaných DOM metod (viz. tabulka 5) a vytvořen element s příslušnými parametry.

Návratovou hodnotou metody pro vytváření položek produktů je výsledný dokument, který je pomocí PHP příkazu „echo()“ vypsán do formulářového pole description. Tímto posledním krokem celá funkce končí.

9. Konfigurace a implementace knihovny

Navržená knihovna nabízí různé úrovně využitelnosti a je určena jak pro správce elektronických obchodů, tak pro vývojáře. V případě vývojářů se dá předpokládat, že dojde k implementaci knihovny do struktur elektronických obchodů a tedy i jakéhosi zautomatizování procesu vytváření produktových dat. Takovéto využití však vyžaduje úpravu některých nastavení knihovny. V souboru example/configuration.php je nutné nastavit tyto hodnoty:

- kód jazyka dle normy ISO 639-1,
- znakovou sadu (kódování výstupního souboru),
- adresu s umístěním specifikace (pouze v případě, že došlu ke změně ontologie),
- cestu k souboru s překlady.

Poslední fází implementace knihovny je umístění textu či obrázku (oranžová ikonka XML) na titulní stránku e-shopu, odkazující na umístění souboru s produktovými daty. Vyhledávačům tak je řečeno, že stránky obsahují rozšiřující informace a jsou k nalezení na odkazované adrese.

Navržený formát využívá deklarace ontologie pojmů umístěné v ontologickém registru (viz kapitola 7.1). Tato deklarace však může být nahrazena i jinou ontologií a v takovémto případě musí dojít k přepsání adresy s novým umístěním specifikace. Jedná se o změnu řádků 13 a 14 v souboru specifications/ns1.owl a změnu definice konstanty na řádku 8 v souboru example/configuration.php.

Jednou z možností využití navržené ontologie a části knihovny je tvorba aplikací založených na myšlence přenosu XML dat přes http. Z databází elektronických obchodů by tak šlo provádět automatizované extrahování produktových dat, jejich strukturalizaci (využití navržené knihovny) a následné zasílání webové službě (kombinace protokolu SOAP a mnou navrženého formátu).

10. Závěr

Tato práce si kladla za cíl navrhnout optimální datový formát (dále jen formát) pro agregaci dat v prostředí elektronických tržišť a následně pro navržený formát vytvořit podpůrné programové prostředky.

Rozbor problémové oblasti ukázal, že nejrozšířenějším formátem u českých B2B portálů, zaměřených na agregaci produktových dat, je struktura popsaná pomocí značek jazyka XML bez deklarace jakéhokoliv jmenného prostoru. Jde tedy o formát jednoúčelový a pro aplikace, které neznají přesný význam značek nepoužitelný. Takováto zjištění jen potvrzují, že má smysl se tímto tématem zabývat.

Jako možné řešení se nabízí použití technologií sémantického webu, konkrétně jazyků pro popis dat. Z široké škály možných variant byla vybrána kombinace sémanticky nejsilnějších jazyků (RDF/XML a OWL-FULL) a za použití podpůrných programových prostředků (Protégé) byla navržena ontologie. Díky jmenným prostorům XML mohlo dojít k začlenění ontologie do struktury formátu a vytvořit tak přesnou definici pojmů použitých k popisu produktových dat. Tím byly vyřešeny problémy spojené s hodně obecným vymezením pojmů (viz kapitola 3.1) a formát tak lze použít jak k výměně dat v rámci B2B elektronických tržišť, tak i k poskytování informací o nabízených produktech sémanticky orientovaným vyhledávačům. Uplatnění navrženého řešení je tedy oproti dosavadním formátům mnohem širší a nabízí se jeho možné použití v praxi.

Dalším přínosem práce je vytvoření PHP knihovny, umožňující generování a vzájemný převod až 4 druhů formátů. Potencionální uživatelé, kteří nevlastní e-shop s generátorem feedu, tak již nemusejí provádět konstrukci svých produktových kanálů v textových editorech a zároveň jim odpadá i nutnost důsledné kontroly správné syntaxe zápisu XML. Knihovna může být také implementována do struktur elektronických obchodů a provádět tak automatické generování XML, RSS a RDF validních dokumentů.

Práci je také možné interpretovat jako studii, která poukazuje na nedostatky stávajících řešení a navrhuje možná zlepšení.

Na tomto místě se také hodí uvést varianty dalšího vývoje formátu. Mezi ty nejdůležitější patří například rozšíření ontologie o definice lokálních elementů (viz kapitola 7.1) či implementace validátoru produktového kanálu.

Pohlížet na formát jako na možný standart by bylo v této chvíli asi předčasné. Záleží totiž především na vůli provozovatelů elektronických tržišť, zda začnou podporovat technologie sémantického webu a tedy i mnou navržený formát. Možnou alternativou na straně správců elektronických obchodů je také paralelní použití více formátů, což by vedlo k zviditelnění obchodů a tedy i nabízených produktů aspoň u sémanticky orientovaných vyhledávačů (SWSE, SWOOGLE nebo Semantic Web Client Library). Masovější nasazení navrženého formátu v takovémto případě však nelze předpokládat.

Možnému rozšíření PHP knihovny by také mohlo napomoci její publikování pod GNU licencí na portálech github[24], PHP classes[25] či umístění online verze na portál ontoProdukt[26].

[24] https://github.com/howorka/ontoPordukt
[25] http://www.phpclasses.org/browse/package/6706/download/zip.html
[26] http://www.ontoprodukt.ic.cz/example/

SEZNAM POUŽITÉ LITERATURY

[1] SPÁČIL, P. Modely elektronického obchodu v celosvětovém podnikání [online]. Vědecko-populární článek z Vysoké školy báňské - Technická univerzita Ostrava, 2009. URL: <http://moodle.vsb.cz/moodle/mod/resource/view.php?inpopup=true&id=79283>

[2] MATULÍK, P. ; PITNER, T. Sémantický web a jeho technologie [online]. Zpravodaj ÚVT MU. ISSN 1212-0901, 2004, roč. XIV, č. 3, s. 15-17. URL: <http://www.ics.muni.cz/zpravodaj/clanky_tisk/296.pdf>.

[3] BECKETT, D . ; W3C. RDF/XML Syntax Specification (Revised) [online]. W3C Recommendation, 10 February 2004. URL: <http://www.w3.org/TR/REC-rdf-syntax/>.

[4] GRUBER, T.R.. A Translation Approach to Portable Ontology Specifications [online]. Knowledge Acquisition, 1993. URL:<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.117.3273&rep=rep1&type=pdf >.

[5] ABDENADHER, S. Ontology and Knowledge Representation Formalisms [online]. Vědecko-populární článek z univerzity v Mnichově, 2009 URL:<http://www.pms.ifi.lmu.de/mitarbeiter/ohlbach/Ontology/index.html>.

[6] STUCHLÍK, R. Ontologie a Sémantický Web [online]. Diplomová práce, Vysokém učení technické v Brně, Fakulta informačních technologií, 2007.

[7] BERNERS-LEE, T. James Hendler and Ora Lassila (May 17, 2001). The Semantic Web. Scientific American Magazine, 2008.

[8] HARWARD, L. RSS 2.0 Specification [online]. RSS Advisory Board, 2009. URL: < http://www.rssboard.org/rss-specification>.

[9] BRICKLEY, D. ; MILLER, L. FOAF Vocabulary Specification 0.98 [online]. Namespace Document, 2010. URL:<http://xmlns.com/foaf/spec/>.

[10] BACKER, T. ; DCMI Usage Board [online]. Dublin Core Metadata initiative, 2010. URL:< http://dublincore.org/>.

[11] GUDGIN, M.; HADLEY, M.; MANDELSOHN, N.; Latest SOAP versions, W3C Recommendation, 2007. URL: < http://www.w3.org/TR/soap/>

[12] HAŠLER, P. Experimentální systémy pro sémantický web. Diplomová práce, České vysoké učení technické v Praze, Fakulta elektrotechnická, 2006.

[13] SURE, Y., ERDMAN, M., STUDER, R.: OntoEdit: Collaborative Engineering of Ontologies, Semantic Web enabled Knowledge Management. J. Davies, D. Fensel, F. van Harmelen (eds.), 2002.

[14] HUSÁKOVÁ, M. Znalostní technologie I [online]. Osobní stránky a stránky pro podporu výuky na Univerzitě Hradec Králové, Fakulta informatiky a managementu, 2008. Kapitola 4.: představení jazyka OWL. URL:<http://lide.uhk.cz/fim/ucitel/fshusam2/lekarnicky/zt1/zt1_kap04_02.html>.

[15] HUSÁKOVÁ, M, c.d, Kapitola 4.: představení jazyka OWL.

[16] HUSÁKOVÁ, M, c.d, Kapitola 4.: představení jazyka OWL.

[17] HARRY, H., RENATO, I., BRIAN, S., NORMAN, W., Representing vCard Objects in RDF [online]. W3C Member Submission, 2010. URL:< http://www.w3.org/Submission/vcard-rdf/>.

[18] MAREŠ,V., NOVÁK,O. AirCraft ontology, Semesterální práce z TSW, Technická Univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, 2010.

[19] HUSÁKOVÁ, M, c.d, Kapitola 5.: Tvorba OWL ontologie krok za krokem.

[20] HUSÁKOVÁ, M, c.d, Kapitola 6.: Reprezentace sémantiky tříd (1).

[21] MATULÍK, P., PITNER, T. Sémantický web a jeho technologie. Zpravodaj ÚVT MU. ISSN 1212-0901, 2004, roč. XIV, č. 3, s. 15-17. URL: <http://www.ics.muni.cz/zpravodaj/articles/296.html>.

[22] GILMOR, W.Jason. Velká kniha PHP a MySQL 5 – kompendium znalostí pro začátečníky i profesionály, 2.vydání. Brno : Zoner Press, 2007. ISBN 80-86815-53-6

[23] OCTAVIA A. A. Parsing XML with the DOM Extension for PHP 5 [online]. PHP Builder, 2010. URL: <http://www.phpbuilder.com/columns/DOM-XML-extension/Octavia_Anghel102710.php3>