České Vysoké Učení Technické v Praze,
Fakulta elektrotechnická

Semestrální práce 33SEM
Porovnání datových formátů v e-commerci

Pavel Jisl
mailto:jislp@mobot.felk.cvut.cz

Contents

Chapter 1

Úvod do problému

1.1 Úvod

1.1.1 Motivace

V posledních několika letech se velmi rozmáhá odvětví technologií zaměřených na Internet. Tyto technologie jsou žádané hlavně z důvodu snadné dostupnosti, jednoduchosti používání a hlavně rychlosti.

Vzhledem k těmto možnostem je možné vysledovat přechod od standardních metod obchodování k metodám obchodování na Internetu. Je to jasný a očekávaný přesun, který má kromě výše zmíněných přínosů ale i své problémy.

Obchodování na Internetu lze rozdělit do dvou kategorií

Bussiness to Bussines -B2B – obchodování v oblasti obchodních partnerů. Tato oblast je velmi důležitá pro výměnu informací mezi výrobními podniky a prodejci.

Bussiness to Customer -B2C – jinak taktéž *Web Commerce* neboli prodej po Internetu koncovým zákazníkům. Je to v podstatě obdoba zásilkových obchodů s tím, že klient si zboží vybírá a objednává přes Internet. A v lepším případě ho i přes Internet zaplatí.

A vzhledem k tomu je třeba zajistit, aby si jednotliví obchodní partneři mohli data vyměňovat a jejich software jim rozuměl. To zajišťuje standard *EDI/FACT*[1] (*Electronic Data Interchange for Administration, Commerce and Transport*), jehož vývoj začal už v 60tých letech a starají se o něj

[1] UN/EDIFACT: http://www.unece.org/trade/untdid/welcome.htm

mezinárodní organizace OSN[2], ISO[3] a v naší republice ČSNI[4]. Ovšem tento standard je složitý a náročný na implementaci a proto se začaly používat formáty založené na XML. O tomto jazyku se zmíním v následující kapitole.

1.1.2 Cíle práce

Cílem této práce je porovnat několik datových formátů, používaných v oblasti e-Commerce nebo e-Bussines. V úvodní části naleznete krátký popis používaných datových formátů, dále popis jazyka XML a dalších používaných standardů pro zápis a popis formátů a popis a porovnání vybraných datových formátů s ohledem na jejich transformaci mezi sebou.

1.1.3 Datové formáty v e-commerce

Zde bych se rád věnoval letmému seznámení s několika formáty, založenými na XML v oblasti elektronického obchodu.

Prvním z nich je ***cXML***[5] (commerce eXtensible Markup Language), vytvořený a udržovaný firmou Ariba Inc. Jedná se o otevřený, univerzální jazyk, jehož hlavním cílem je tvorba katalogů, tzv. PunchOut protokolu (viz. kapitola 3.2.1) a platebních příkazů.

Zajímavým projektem je XML Common Business Library, ve zkratce ***xCBL***[6]. Stejně jako i cXML i xCBL je otevřený a volně použitelný formát. O tento formát vytvořila a o jeho rozvoj se stará firma CommerceOne[7], na jejichž stránce nalezneme spoustu zajímavých informací o e-Bussinesu a e-Commerci. xCBL jako jazyk je postaven na standardu EDI/FACT, ale pro zápis je využíván jazyk XML. Zápis je podobný jako v EDI/FACT a proto i převod mezi těmito dvěma formáty je snadný.

Konsorcium W3[8], které v poslední době vydává standardizaci jednoho formátu za druhým (zde stále zmiňované XML, jeho zjednodušení a odnož pro www stránky XHTML a CSS), se v poslední době zaměřuje také na sémantický web a ontologie pro práci s ním. Tímto standardem je ***OWL***[9], který vychází z jazyka ***DAML+OIL***. OWL slouží pro definici ontologií (termín vypůjčený z filosofie, sloužící k popisu druhů entit a jejich vztahů)

[2] UN/ECE - CE/FACT: http://www.unece.org/cefact/
[3] ISO: http://www.iso.org/iso/en/ISOOnline.frontpage
[4] ČSNI: http://www.csni.cz
[5] cXML - http://www.cxml.org
[6] xCBL - http://www.xcbl.org
[7] CommerceOne - http://www.commerceone.com
[8] W3 Consorcium - http://www.w3.org
[9] Web Ontology: http://www.w3.org/2001/sw/WebOnt/

a s nimi souvisejícími bázemi znalostí. V OWL jsou ontologie soupis definicí tříd a vlastností a elementy jazyka, které mohou označovat souvislost mezi třídami, popis atributů elementů a vztahy mezi elementy mezi třídami. Tyto jazyky jsou založeny na *RDF* (Resource Description Framework), jsou ale jsou spíše myšleny pro použití v budoucnosti.

Další firmou, zabývající se vývojem formátů a zázemím pro e-Commerce, je RosettaNet[10]. Tato firma navrhla celý komplex formátů a komunikačních protokolů, které shrnula pod názem - Partner Interface Processes. Celý postup návrhu, vlastní tvorby a zprovoznění e-Commerce nebo e-Bussines serveru je na stránce RosettaNet velmi podrobně popsáno. Samozřejmostí je podpora XML a otevřenost projektu.

Zajímavým počinem je ***OCF*** - Open Catalog Format firmy MartSoft[11], který definuje jednoduché DTD pro práci s katalogy. Pomocí tohoto DTD lze reprezentovat, ukládat a přenášet obsah katalogu, který může obsahovat buď jeden produkt, kategorii nebo podkategorii produktů nebo celý produkt. Vzhledem k tomu, že se jedná o obecnou definici formátu, nedefinuje kategorizaci produktů. To zůstává na uživateli. Na svých stránkách ale firma MartSoft nabízí převod z tohoto formátu do různých formátů katalogů jiných firem, například do cXML, xCBL nebo do formátu používaného firmou RosettaNet. Tato služba je nabízena na stránkách firmy MartSoft (viz. [12] a [13]), ale je zřejmě placená.

[10] RosettaNet - http://www.rosettanet.org
[11] MartSoft - http://www.martsoft.com
[12] http://www.martsoft.com/technology.htm
[13] http://www.martsoft.com/products/ccc/

Chapter 2

Formát XML

2.1 Historie

Jazyk *XML* [3] (eXtensible Markup Language) je takzvaný „značkovací“ jazyk, který slouží pro obecnou definici formátu souboru. Dalšími příklady těchto jazyků je např. jazyk HTML, SGML nebo třeba i jazyk, který se používá pro tvorbu dokumentů v programových balících TEXa LATEX.

Jedním z prvních značkovacích jazyků byl již zmíněný jazyk *SGML* (Standard Generalized Markup Language), který je definován v normě ISO 8879 z roku 1986. Jeho možnosti jsou velmi silné, lze samozřejmě definovat vlastní značky (tagy) pomocí *DTD* (Document Type Definition, tedy definice typu dokumentu). Jeho komplexnost a velká rozšiřitelnost ovšem měla problém s jeho implementací a jednoduchostí použití a tak se příliš nerozšířil. Jeho nejznámější aplikací je jazyk *HTML* (HyperText Markup Language), jehož největší rozšíření umožnil rozvoj Internetu a hlavně jeho jednoduchost.

V druhé polovině 90. let bylo zřejmé, že jazyk SGML je velmi dobře navrhnut, ale v praxi se využívají jen jeho části a proto jeho nejdůležitější podmnožina byla vybrána pro nový jazyk. Tento jazyk dostal název XML a o jeho standard se stará konsorcium W3 [1]. Je samozřejmě rozšiřitelný o nové prvky pomocí DTD, ale jeho syntaxe je oproti SGML striktnější a mnoho parametrů nelze měnit.

Tyto vlastnosti zapříčinily široké rozšíření jazyku XML do všech sfér Internetu, kde je třeba uchovávat a zpracovávat množství informacích. Jazyk XML se hodí stejně k publikování dat na Internetu (jeho nyní nastupující odnož XHTML, nový standard pro publikaci na WWW, který nahrazuje předchozí standardy HTML) jako pro uchovávání dat v databázích. Jeho výhodou je totiž zachycení důležitých informací o struktuře a významu uložených dat.

2.2 Vlastnosti

2.2.1 Standardní formát pro výměnu dat

Jazyk XML je otevřený formát, jehož specifikace je k dispozici zdarma pro každého na stránce konsorcia W3. Tím je umožněna jeho snadná implementace oproti proprietárním datovým formátům, např. datovým formátům firmy Microsoft a to hlavně z aplikací Word a Excel.

Další výhodou je, že celý jazyk XML je zapsán jako obyčejný text a proto na jeho tvorbu a úpravu stačí běžný textový editor. Je samozřejmé, že plnohodnotné využití všech vlastností umožní pouze aplikace, která ho vytvořila, ale pro jednoduché úpravy je to výhoda.

2.2.2 Jazyková podpora

Jazyk XML byl od počátku tvořen jako multilinguální, o čemž svědčí použitá jazyková sada ISO 10646. Tato jazyková sada je 32bitová, lze tedy do ní uložit všechny světové jazyky a nářečí. Lze tedy v dokumentu psát nejen česky a německy, ale třeba i japonsky a hebrejsky.

Je samozřejmé, že 32bitová znaková sada je plýtváním místa, proto lze samozřejmě použít i standardní národní sady jako ISO 8859-2 nebo 16bitovou sadu UTF-8.

2.2.3 Konverze do dalších formátů

Další výhodou jazyka XML je velmi jednoduchá konverze do jiných formátů. K této konverzi se používá aplikace šablon. Nejjednodušší a asi nejznámější jsou šablony CSS, tedy kaskádové styly známé z HTML a XHTML. Ty lze ovšem použít jen pro formátování.

Pro úplnou transformaci do jiného formátu se používají speciálně navržené styly *XSL* (eXtensible Stylesheet Language). S tímto jazykem lze např. vypouštět některé informace z dokumentu nebo generovat obsahy.

2.2.4 Kontrola struktury dokumentu

Vzhledem k možnosti definice vlastních značek je důležité, zda v našem dokumentu používáme značky tak, jak jsme si je nadefinovali. To umožňuje soubor s DTD, kde je zapsané, jaké značky používáme, jaké mohou mít parametry a obsah. K tomu slouží tzv. *validity parser*, který ovaliduje XML soubor oproti zadanému DTD.

XML dokument	**popis**
`<?xml␣version="1.0"␣encoding="utf-8"?>`	XML deklarace a specifikace použitého jazyka
`<!DOCTYPE␣katalog␣SYSTEM␣"katalog.dtd">`	připojení použitého DTD k dokumentu
`<katalog>`	kořenový element
`␣␣<firma␣id="12345678"␣typ="S.R.O.">`	otevírací část párového elementu a atributy
`␣␣␣␣<název>Moje␣firma</název>`	text *Moje firma* je obsah elementu *název*
`␣␣</firma>`	uzavírací část párového elementu
`␣␣<platnost␣do="1.2.2003"␣/>`	nepárový nebo prázdný element
⋮	⋮
`</katalog>`	uzavření kořenového elementu

Table 2.1: Příklad XML dokumentu

2.3 Základy jazyka XML

2.3.1 Jazyk XML

Jazyk *XML* je, jak již bylo zmiňováno, značkovacím jazykem (*markup language*) a tyto značky se nazývají *elementy*. Pomocí elementů se pak označují důležité části dokumentu. Elementy lze do sebe vnořovat a tím zachytit strukturu dokumentu.

V tabulce 2.1 je ukázka XML dokumentu s popisem.

2.3.2 Definice typu dokumentu

Takto vytvořený dokument je však potřeba pro správnou činnost doplnit jeho definicí typu dokumentu (*DTD*), pak lze totiž automatizovat jeho validaci. Přiřazení DTD k dokumentu se provede vložením řádku (viz. 2.1)

```
<!DOCTYPE katalog SYSTEM "katalog.dtd">
```

kde

katalog je název kořenového elementu

SYSTEM je klíčové slovo, které říká, že následující atribut je cesta k souboru s DTD a že se jedná o definici privátní. Lze použít klíčové slovo PUBLIC, které ukazuje, že se jedná o standardizované DTD a prohlížeč (nebo jiný nástroj) ji pak nemusí stahovat z Internetu. Po této definici

`<!ELEMENT␣katalog␣(firma+,␣platnost)>`	definice kořenového elementu
`<!ELEMENT␣firma␣(#PCDATA,␣název)*>`	element firma, lze použít element název a obecná data
`<!ATTLIST␣firma`	definice atributů, patřících k elementu firma
`id␣CDATA␣#REQUIED`	id firmy, vyžadované
`typ␣(S.R.O.|A.S.|V.O.S.)␣'S.R.O'>`	výčtový typ, standardně použito S.R.O.
`<!ELEMENT␣název␣(#PCDATA)>`	
`<!ELEMENT␣platnost␣(#PCDATA)>`	
`<!ATTLIST␣platnost`	
`do␣CDATA␣#REQUIRED>`	

Table 2.2: Ukázka definice DTD

následují dva atributy, první s veřejným identifikátorem a druhý s URL k definovanému DTD.

Definici lze samozřejmě také vložit do XML souboru a tím např. upravit vložená externí DTD. Ale vzhledem ke speciálnímu použití se zde touto možností nebudu zabývat.

V tabulce 2.2 je zobrazen ukázkový příklad souboru DTD, který popisuje elemnty, které lze použít v příkladu 2.1.

V této definici jsou použity symboly ?, * a +, určující, kolikrát lze podřízený element v elementu použít. Zde je popis těchto elementů:

? - element nemusí být použit nebo může být použit jednou

***** - element nemusí být použit nebo může být použit n-krát

+ - element musí být použit alespoň jednou

a | **b** - možnost použití elementu **a** nebo elementu **b**

2.4 XML-Schema

Vzhledem k tomu, že jednotlivé datové formáty jsou v různých odrůdách XML, musím se zde zmínit i o formátu *XML-Schema*[2], neboť v něm je zapsána definice formátu xCBL. XML-schémata byla vytvořena jako univerzální jazyk, který by měl nahradit všechny existující jazyky pro popis struktury XML. Jeho definice je však podle vývojářů tak složitá, že se zatím moc nerozšiřuje. Možnosti definice datových typů jsou totiž tak veliké, že by si vyžádaly samostatnou definici.

Lze například určit typ také u elementů a k dispozici jsou všechny možné datové typu, od datumu, přes logické hodnoty až po různé typy řetězců, ale je samozřejmě možné si definovat či odvodit své vlastní typy. Lze si tak například nadefinovat element, u kterého bude určená minimální a maximální délka řetězce nebo počet desetinných míst u čísel.

V tabulce 2.3 je tedy vidět, jak v tomto jazyku nadefinovat „DTD“ pro příklad uvedený v tabulce 2.1. Jedná se samozřejmě o velmi jednoduchý náčrt možností jazyka XML-Schema, ale pro složitější definice je z mého pohledu jeho použití lepší než použití DTD. To hlavně z důvodu, že definice v XML-Schema vypadá podobně jako zápis v dokumentu, který ho používá. Také další možnosti jsou robustnější, mohu zde zmínit například možnost „natvrdo“ nadefinovat pořadí, v jakém mají být elementy použity, možnost definice výčtových (enumerate) typů, union typů a mnoho dalších vlastností, které usnadní autorům práci s výslednými XML dokumenty.

2.4.1 XPath

XPath je standard konsorcia W3, který slouží pro adresování částí XML dokumentu. Jeho použití vypadá podobně jako odkazování v souborovém systému operačních systémů. Dále umožňuje manipulace s řetězci, čísly a logickými hodnotami. XPath modeluje XML dokument jako strom uzlů. Za uzel se pak berou elementy, atributy a texty v elementech a atributech.

Zde je zjednodušený popis (pro náš případ dostačující) nejdůležitějších vlastností (pro podrobnější popis doporučuji [5] a tutorial [6]).

`/AAA` – vybírá kořenový element („absolutní cesta“)

`/AAA/BBB` – vybírá všechny elementy BBB, které jsou přímými potomky kořenového elementu AAA

`//BBB` – vybírá všechny elementy BBB, které jsou kdekoli ve stromu

`/AAA/BBB/*` – vybírá všechny elementy, které jsou přímými potomky /AAA/BBB

`/*/*/CCC` – vybírá elementy CCC, které mají právě 2 předky

`//*` – vybírá všechny elementy ve stromu

`/AAA/BBB[1]` – vybírá prvního přímého potomka BBB elementu AAA

`/AAA/BBB[last()]` – vybírá posledního přímého potomka BBB elementu AAA

```
1. <xsd:schema␣xmlns:xsd="http://www.w3.org/2001/XMLSchema">
2.
3. <xsd:element␣name="katalog"␣type="katalogType"␣/>
4.
5. <xsd:complexType␣name="katalogType">
6. ␣␣<xsd:sequence>
7. ␣␣␣␣<xsd:element␣name="firma"␣type="firmaType"␣minOccurs="1"␣/>
8. ␣␣␣␣<xsd:element␣name="platnost"␣type="xsd:string"␣/>
9. ␣␣</xsd:sequence>
10. </xsd:complexType>
11.
12. <xsd:complexType␣name="firmaType">
13. ␣␣<xsd:sequence>
14. ␣␣␣␣<xsd:element␣name="název"␣type="xsd:string"
15. ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣minOccurs="1"␣maxOccurs="1"␣/>
16. ␣␣</xsd:sequence>
17. ␣␣<xsd:attribute␣name="id"␣type="xsd:string"
18. ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣minOccurs="1"␣maxOccurs="1"␣/>
19. ␣␣<xsd:attribute␣name="typ"␣type="typFirmy"␣fixed="S.R.O"␣/>
20. </xsd:complexType>
21.
22. <xsd:simpleType␣name="typFirmy">
23. ␣␣<xsd:restricted␣base="xsd:string">
24. ␣␣␣␣<xsd:enumeration␣value="S.R.O."␣/>
25. ␣␣␣␣<xsd:enumeration␣value="A.S."␣/>
26. ␣␣␣␣<xsd:enumeration␣value="V.O.S."␣/>
27. ␣␣</xsd:restricted>
28. </xsd:simpleType>
29.
30. </xsd:schema>
```

Table 2.3: Příklad definice XML-Schématu k příkladu v tabulce 2.1.

`//@id` – vybere všechny atributy `id`

`//CCC[@id]` – vybere všechny elementy CCC, které mají atributy `id`

`//CCC[@*]` – vybere všechny elementy CCC, které mají jakýkoli atribut

`//CCC[@id='1234']` – vybere všechny elementy CCC, které mají atribut `id` rovný 1234

2.4.2 Resource Description Framework

Formát ***RDF***[4] je určen pro zpracování metadat – zprostředkovává spolupráci mezi aplikacemi, které si vyměňují informace strojově srozumitelné na Webu. V při návrhu RDF byl kladen důraz na možnost automatizovaného zpracování informací obsažených na Webu. Lze ho použít pro spoustu aplikací – od vyhledávacích agentů, přes hodnocení obshahu (content rating) až po popis ucelených WWW stránek.

Hlavní vlastnosti formátu RDF jsou:

- možnost zachycení složitých vztahů mezi zdroji
- syntaxe založená na XML
- podpora libovolného slovníku metadat (například Dublin Core[1], ...)
- snadné automatické zpracování
- může být součástí dokumentu, který popisuje, nebo může být samostatným dokumentem
- RDF umožňuje jednoduché skladování v relačních databází

[1] Dublin Core: http://dublincore.org, http://www.ics.muni.cz/dublin/_core/index.html

Chapter 3

Popis vybraných formátů

3.1 Úvod

Vzhledem k velké obšírnosti formátů jsem se rozhodl porovnat mezi sebou pouze dva, cXML a xCBL a jako zajímavou alternativu také projekt Open Catalog Project. Pro porovnání jsem zvolil porovnání katalogů, protože z mého pohledu se jedná o jednu z nejzajímavějších částí.

Katalog slouží k nabídce zboží dalším e-Bussines klientům, ale lze je využít také pro koncové zákazníky. V katalogu by mělo být zahrnuto nejen jaké zboží se nabízí, jaká je jeho cena a popis, ale také kdo toto zboží vyrábí, kdo ho dodává a jaké jsou možnosti dodání. V těchto ohledech mi přijdou mnou vybrané datové formáty asi nejobsažnější a vzhledem k dalšímu vývoji také nejdůležitější.

Pokusím se tedy porovnat DTD a navrhnout možný proces transformace pro převod katalogu z jednoho formátu do druhého.

3.2 cXML

Jak jsem se zmínil již v úvodní části práce, ***cXML*** je datový formát založený na XML a veškeré důležité vlastnosti se definují v DTD. Formát cXML má velmi obsáhlý katalog DTD, jeho velikost je okolo 65 kB, což je přibližně 2000 řádek. Ale firma Ariba nabízí DTD také rozdělené do menších celků, které lze používat jednotlivě.

3.2.1 PunchOut

Zde bych rád popsal jednu ze zajímavostí datového formátu cXML. Jedná se o protokol *PunchOut*. Je to jednoduše inmplementovatelný protokol pro

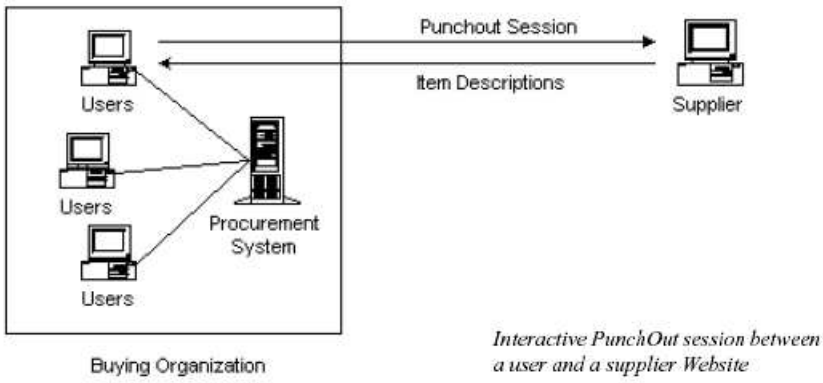


Figure 3.1: cXML: Interaktivní PunchOut sezení mezi uživatelem a WWW stránkou dodavatele.

tvorbu interaktivní komunikace přes Internet, která se provádí v reálném čase zasíláním synchronních zpráv v jazyce cXML. Tím je umožněna výměna aktuálních dat mezi aplikacemi a jednotný vzhled aplikací na vzdálených počítačích. V následujících podkapitolách blíže popíšu tři existující druhy PunchOut.

3.2.1.1 Procurement PunchOut

Procurement[1] PunchOut je altrarnativa ke statickým katalogům. Při použití této možnosti lze stránky popsat jako interaktivní katalogy. Uživatel místo toho, aby viděl informace a ceny o produktech, stiskne tlačítko, které ho přenese na stránky katalogu dodavatele. Tam si uživatel může prohlížet informace o výrobcích, může si je vybírat a určit typ dodání. Po návratu z této aplikace se veškeré informace o objednaném produktu objeví v uživatelově objednávce. To je zjednodušeně zobrazeno v obrázku 3.1.

3.2.1.2 PunchOut Chaining

PunchOut Chaining[2] je rozšířený Procurement PunchOut a to tak, že je možné tyto PunchOuty zřetězovat. To umožňuje vlastnost *cXML Path Routing*. Lze tedy objednávky a další zprávy PunchOut vracet obchodům a dodavateli do fronty (viz. obrázek 3.2 a informovat všechny zainteresované strany o konečné objednávce.

[1] procurement – zprostředkování
[2] chaining – zřetězení

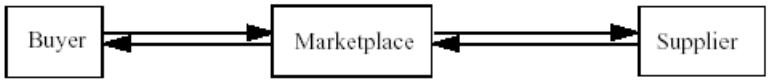


Figure 3.2: cXML: PunchOut Chaining.

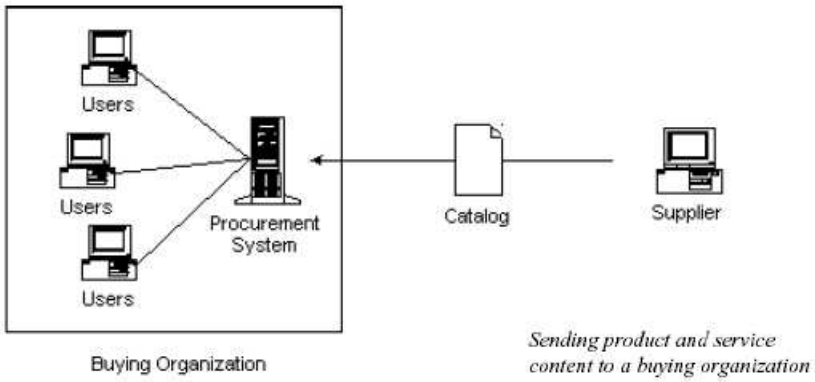


Figure 3.3: cXML: Zasílání katalogu kupující organizaci.

3.2.1.3 Provider PunchOut

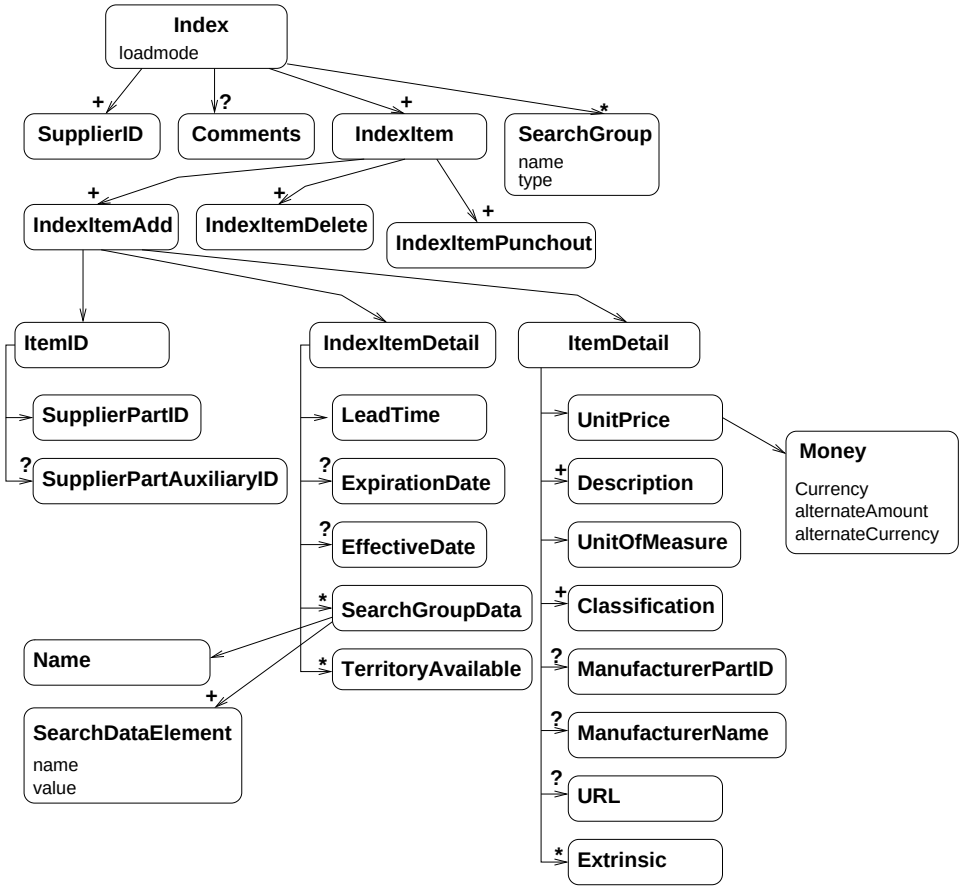
Provider[3] PunchOut umožňuje přenos komunikaci a přenos služeb mezi aplikací, běžící např. u dodavatele a jinou vzdálenou aplikací. Vzdálená aplikace pak může poskytovat služby jako je ověřování kreditních karet nebo autorizace uživatelů.

3.2.2 cXML katalog

Katalogy v podání cXML jsou data, která nabízejí dodavatelé a organizace, používající aplikace založené na cXML (procurement application), mohou vidět nabízené zboží a služby, které lze zakoupit. Tato „procurement“ aplikace katalog přečte a uloží ho do interní databáze (viz. obrázek 3.3). Po schválení katalogu kupující organizací, je tento katalog přístupný uživatelům, kteří mohou vybírat položky a vytvářet tak objednávky.

Definice katalogu cXML (viz. [14]) je rozdělena do dvou hlavních elementů - *Supplier* a *Index*. Element *Supplier* obsahuje všechny důležité informace o dodavateli, tedy jeho adresu, kontakty a možnosti objednání. V elementu *Index* nalezneme veškeré informace o nabízeném zboží, tedy cenu, množství,

[3]provider – dodavatel



V grafu jsou zobrazeny symboly ?, * a +, určující, kolikrát se daný element může použít. Jejich popis je na str. 8.

Figure 3.4: cXML: Zjednodušený graf struktury elementu Index

výrobce a další doplňkové informace.

3.2.3 Popis důležitých elementů

V této sekci bych se chtěl věnovat důležitým příkazům, které se objevují v obrázcích 3.4 a 3.5. Nejvyšším elementem, který všechny ostatní ohraničuje je **Index**. Druhým nadřazeným elementem je element **Supplier**, který definuje informace o prodejci. Tuto informaci pak zveřejňuje nastavením hodnoty elementu **SupplierID**, na který se pak element **Index** odkazuje.

Další hladinou je **IndexItem**. V té mohou být elementy, které representují činnost, jakou má uživatel při přijetí katalogu provést. Zde jsou tři možnosti činností:

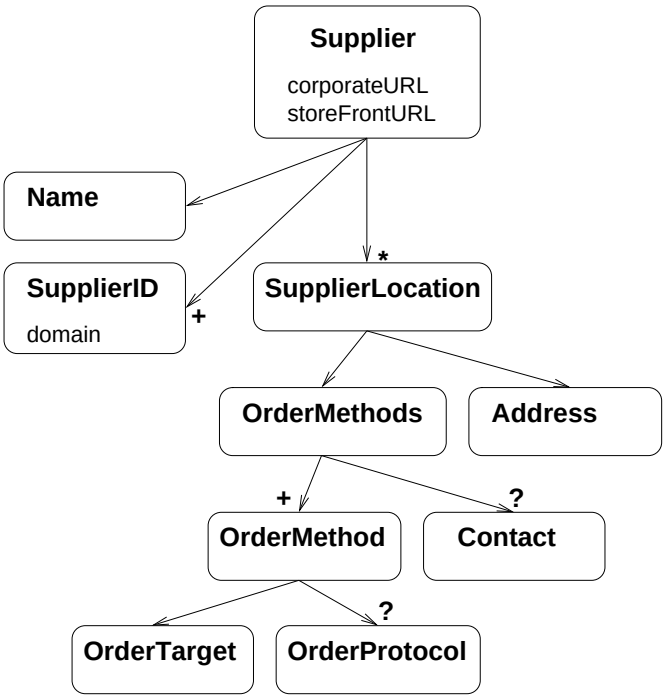


Figure 3.5: cXML: Zjednodušený graf struktury elementu Supplier

IndexItemAdd – vkládá nové položky nebo upravuje vlastnosti existujících položek v katalogu. Obsahuje elementy **ItemID**, **ItemDetail** a **IntexItemDetail**.

IndexItemDelete – maže položky z katalogu. Mazaná položka je identifikována elementem **ItemID**.

IndexItemPunchout – vkládá položku pro inicializaci komunikace PunchOut s WWW stránkou dodavatele. Má podobné vlastnosti jako **IntexItemAdd**, ale nevyžaduje informace o ceně, ty jsou získány přímo ze stránky dodavatele. Obsahuje elementy **PunchoutDetail** a **ItemID**.

3.2.3.1 ItemID

Element **ItemID** jednoznačně definuje zboží, které přísluší určitému dodavateli. Proto obsahuje dva elementy - **SupplierPartID** a **SupplierPartAuxiliaryID**. Pokud první z nich dostatečně neidentifikuje zboží (např. lze zakoupit úplně stejnou mechaniku CD-RW v krabici nebo pouze v igelitovém obalu - tj. stejné označení, pouze rozdíl v balení a ceně), je nutné nadefinovat ještě druhý element.

3.2.3.2 ItemDetail

Element **ItemDetail** obsahuje podrobné informace o zboží. Musí obsahovat následující elementy:

UnitPrice – cena za jednotku zboží, tento element obsahuje element **Money** a jeho podelementy **currency**, **alternateCurrency** obsahují typ měny podle ISO normy 4217 [8]

UnitOfMeasure – definice jednotky zboží podle standardu UN/CEFACT (doporučení 20) [10]

Description – slouží k podrobnému popisu zboží, může obsahovat element **ShortName**, tedy zkrácený popis

Clasification – slouží ke kategorizaci zboží do skupin podle druhu zboží

Následující elementy jsou nepovinné, ale vhodné pro ještě podrobnější popis zboží:

ManufacturerPartID – identifikace zboží výrobcem

ManufacturerName – jméno výrobce

URL – www adresa výrobce, popř. přímo zboží

Extrinsic – doplňující informace ke zboží

3.2.3.3 IndexItemDetail

Element **IndexItemDetail** definuje doplňující informace o zboží.

LeadTime – určuje dobu ve dnech, za kterou je možné zboží dodat

ExpirationDate – určuje dobu, za kterou už zboží není platné (hodnoty podle normy ISO 8601 [9])

EffectiveDate – určuje dobu, za kterou zboží bude platné (hodnoty podle normy ISO 8601 [9])

SearchGroupData – slouží ke specifikaci informací, sloužících k vyhledávání

TerritoryAvailable – specifikace zemí (ISO kódy zemí a/nebo kódy regionů [7]), kde je zboží dostupné

3.2.4 Závěr

Jak je vidět, podpora tvorby katalogů v jazyku cXML je domyšlená do konce a snaží se o zjednodušení návrhu na nejnutnější věci. Hlavní výhodu vidím v možnosti ponechat katalog u prodejce a objednávat pomocí protokolu PunchOut z jeho katalogu. Lze samozřejmě používat katalog i „staticky“, tedy tak, že se hodnoty z katalogu ukládají na server a při změně se obnoví.

3.3 xCBL

Jazyk ***xCBL*** je opět založen na XML, ale jeho definice nejsou zapsány v DTD jako u cXML, ale v jazyku XML-Scheme. Tento jazyk jsem zjednodušeně popsal na straně 8. Tento jazyk umožňuje přehlednější zápis formátu a toho je využito v popisu jazyka xCBL. Veškeré použité elementy jsou popsány v jednotlivých souborech, pro každý typ jeden. Celkem je těchto souborů přes 700 (v mnou použité verzi xCBL 4.0 beta) a zabírají téměř 4 MB disku. Ukázka definice jednoho z datového typů (ProductIDType.xsd) je v tabulce 3.1.

Oproti datovému formátu xCBL není dostupná příliš kvalitní dokumentace (cXML obsahuje 250ti stránkovou uživatelskou příručku ve fromátu PDF, kde je vše potřebné popsané). Tato dokumentace je v HTML a jedná se pouze o popis elementů. Navíc nefunguje v jiných prohlížečích než Internet Explorer (kvůli nestandardní implementaci funkcí v JavaScriptu).

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
3.      xmlns="rrn:org.xcbl:schemas/xcbl/v4_0/catalog/v1_0/catalog.xsd"
4.      xmlns:core="rrn:org.xcbl:schemas/xcbl/v4_0/core/core.xsd"
5.  targetNamespace="rrn:org.xcbl:schemas/xcbl/v4_0/core/catalog.xsd"
6.      elementFormDefault="qualified">
7.
8.  <xsd:import namespace="rrn:org.xcbl:schemas/xcbl/v4_0/core/core.xsd"
9.      schemaLocation="../../core/core.xsd"/>
10.
11.   <xsd:complexType name="ProductIDType">
12.     <xsd:simpleContent>
13.       <xsd:extension base="xsd:string">
14.         <xsd:attribute name="Type" type="BuyerSupplierCodeType"
15.             use="optional" default="Supplier"/>
16.       </xsd:extension>
17.     </xsd:simpleContent>
18.   </xsd:complexType>
19. </xsd:schema>
```

Table 3.1: Ukázka definice datového typu ProductID (ProductIDType.xsd)

Pro přehlednost zápisu jsem vytvořil hierarchický strom (obrázek 3.6), který zobrazuje hierarchii elementu **ProductCatalog**. Z obrázku je při

porovnání se stromem katalogu v jazyku cXML (obrázek 3.4), že tyto dva katalogové formáty až na několik odlišností v názvech elementů jsou téměř shodné. V další části se budu věnovat popisu nejdůležitějších elementů tak, aby tyto shody byly patrné.

3.3.1 Popis důležitých elementů

3.3.1.1 ProductCatalog

ProductCatalog je dokument, který se používá k poskytování informací o zboží (cena, popis, ...) mezi obchodními partnery. Lze ho ale využít také k nastavení nestandardních atributů zboží a někdy ho lze používat jako nástroj výměny nestandardních atributů a kategorií zboží.

3.3.1.2 CatalogHeader

CatalogHeader obsahuje administrativní informace o katalogu, jako je informace o dodavateli, poskytovateli katalogu a dalších, které se objevují v katalogu.

CatalogID – unikátní identifikátor katalogu

CatalogDate – datum vytvoření katalogu

CatalogProvider – specifikuje poskytovatele katalogu. Může obsahovat volitelný element **Party** nebo volitelný atribut **ProviderID**. Jeden z těcto elementů (atributů) musí být přítomen.

ValidFrom – specifikuje dobu, od kdy je katalog platný

ValidUntil – specifikuje dobu, do kdy je katalog platný

DefaultLanguage – specifikuje defaultní jazyk, ve kterém je katalog poskytován. Hodnota se nastavuje atributem **xml:lang**, jehož hodnota musí odpovídat specifikaci *RFC 1766* [11]

DefaultCurrency – specifikuje defaultní měnu katalogu (podle normy ISO 4217 [8])

ShortDescription, LongDescription – krátký resp. dlouhý popis katalogu, lze použít více jazykových mutací, které se rozlišují atributem **xml:lang**, jehož hodnota musí odpovídat specifikaci *RFC 1766* [11]

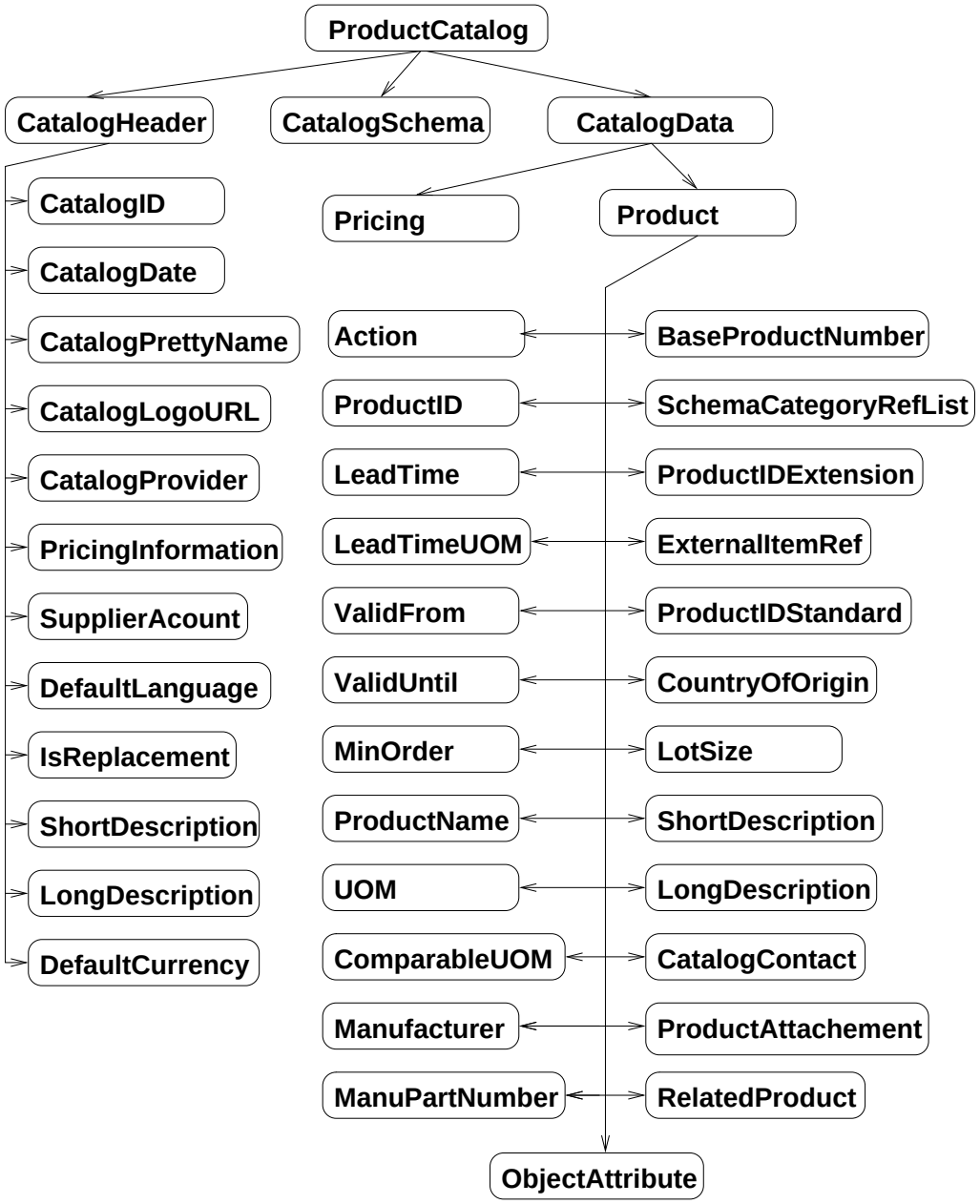


Figure 3.6: xCBL: Zjednodušený graf struktury elementu ProductCatalog

3.3.1.3 CatalogData

Element **CatalogData** slouží jako nadřazený element pro elementy, sloužící k přesnému popisu zboží. Obsahuje v sobě podelementy **Product** a **Pricing**.

3.3.1.4 Pricing

Element **Pricing** obsahuje všechny ceny, přiřazené jednotlivým produktům z jednotlivých cenových katalogů. Jeho podelement **ProductIDRef** resp. **PriceCatalogIDRef** je identifikátor zboží, kterému je cena přiřazena resp. identifikátor cenového katalogu, v němž je zboží zařazeno.

Posledním podlelementem je **ProductPrice**, který již přesně popisuje cenu výrobku. Element je možné opakovat, pokud je výrobku přiřazeno více cen. Jeho podelementy:

Amount – cena výrobku (dekadické číslo)

PriceType – typ ceny

Currency – ISO kód měny (viz. [8])

UOM – jednotky zboží dle [10]

MinimumQuantity – určuje minimální počet zboží, které lze za určenou cenu odebrat

ShortDescription – krátký popis ceny

ValidFrom – určuje, od kdy bude cena platná

ValidUntil – určuje, do kdy bude cena platná

PriceBasisQuant – určuje počet zboží, které je základem ceny (např. balíček po pěti kusech - zde bude uloženo číslo 5)

3.3.1.5 Product

V elementu **Product** nalezneme veškeré elementy, které se přímo váží ke zboží. Jejich počet je však značný (viz. obrázek 3.6), takže v popisu vyberu jen ty z mého pohledu nejdůležitější.

Action – určuje jaká akce se provede s katalogem. Tato hodnota se určuje nastavením atributu **Value**. Pokud je element prázdný, položky se přidávají (**Add**), možné hodnoty jsou:

Add – přidání do katalogu

Update – obnovení

Delete – vymazání

Replace – přepsání

ProductID – unikátní identifikátor zboží

ProdustIDExtension – dodatečný identifikátor zboží (viz. kapitola 3.2.3.1 o cXML)

ProductName – jméno zboží. To lze uvést také v různých řečích použitím elementu vícekrát, ale s různým nastavením atributu **xml:lang** podle RFC 1766 (viz. [11]). Pokud není tento atribut nastaven, bere se jazyk angličtina (en).

UOM – jednotky zboží podle [10]

Manufacturer – jméno výrobce nebo odkaz do elementu **ListOfPartners**

LeadTime – doba, za kterou je možné zboží dodat. Je vyjádřena v počtu dní, pokud není určeno jinak elementem **LeadTimeUOM**, kterým lze nastavit časové jednotky.

ValidFrom – udává datum, od kterého je zboží dostupné

ValidUntil – udává datum, od kterého bude zboží nedostupné

CountryOfOrigin – ISO kód (viz. [7]) země původu zboží

MinOrder – minimální počet zboží, který lze odebrat.

LotSize – určuje počet zboží, které lze objednat, jako by bylo jeden kus. Tedy, pokud bude LotSize=3 a MinOrder=5, pak zákazník bude muset odebrat 6 kusů.

ShortDescription, LongDescription – krátký resp. dlouhý popis zboží.

3.3.2 Závěr

Jak je z předchozího popisu zřejmé, formát xCBL je velmi silným popisným jazykem. Jeho katalog lze v podstatě použít na cokoliv, ať už zboží nebo služby. To je však vykoupeno vyšší složitostí použití.

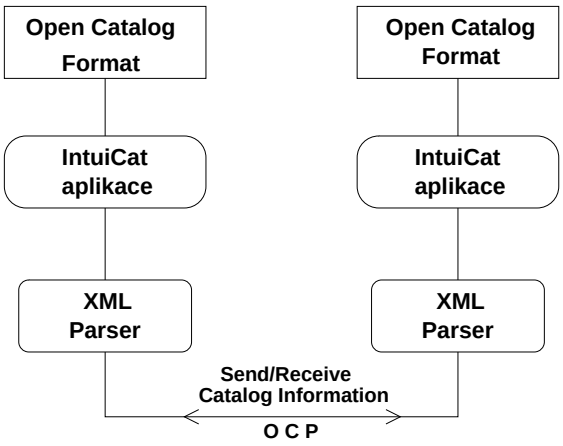


Figure 3.7: OCP: Graf komunikace mezi katalogy.

3.4 Open Catalog Format

Open Catalog Format[4] byl vytvořen firmou MartSoft a slouží k popisu produktových katalogů a k jejich ukládání a přenosu. Pro komunikaci se používá **Open Catalog Protocol**, také navržený firmou MartSoft a implementovaný do jejich e-Commerce produktu *IntuiCat*[5]. Tento produkt v sobě obsahuje prostředí **IntuiCat** pro práci s katalogy, databázi, schopnou ukládat do XML a WWW server. Toto řešení je navrženo tak, aby bylo možné vytvářet hierarchické katalogy s děděním atributů, parametrizované a fultextové vyhledávání, spolupráci s lokálními a vzdálenými katalogy a vícedimenzionální pohled na data.

3.4.1 Open Catalog Protocol

Open Catalog Protocol je zatím ve stádiu vývoje, ale plánuje se jeho standardizace konsorciem W3. Je to protokol založený na XML, který umožňuje práci s komplexními daty katalogů. Skládá se z jazykově-nezávislé reprezentace dat katalogu, založených na XML 1.0, a sady příkazů pro práci s daty katalogů. Tento protokol lze přirovnat k protokolu PunchOut, který využívá cXML. Zjednodušený graf komunikace je v obrázku 3.7. Podrobnější popis všech příkazů lze nalézt v [15].

[4] http://www.oasis-open.org/cover/ocp.html
[5] IntuiCat: http://www.martsoft.com/products/index.html#INTUICAT

3.4.2 Formát OCF

Datový formát OCF, jehož popis je v [16], je popsán buď DTD dokumentem nebo XML Schématem. Jejich délka je velmi malá, v porovnání s cXML nebo xCBL, ale podle ukázkových příkladů[6], umístěných na stránce firmy MartSoft, je jejich použití jednodušší a řekl bych, že podobně silné.

Výhodou je také specifikace **Open Catalog Query Language** (*OCQL*), které definuje dotazovací jazyk (opět založený na XML), jehož pomocí lze klást dotazy na katalogy. Jeho DTD je možné nalézt opět v [16].

Na obrázku na straně 28 je zobrazena hierarchická struktura hlavního elementu **catalog**. Z obrázku je dále patrné, že i přes velkou jednoduchost návrhu lze tvořit velmi složité, hierarchické katalogy. To je umožněno opakováním nejdůležitějších elementů, které se v OCF vyskytují. To je schematicky zobrazeno tečkovanými šipkami.

3.4.2.1 Element catalog

Tento kořenový element obsahuje podelementy **param**, **category**, **product**. Obsahuje atributy

id – identifikátor katalogu

name – jméno katalogu

version – verze OCF, obvykle obsahuje číslo 2.0

3.4.2.2 Element param

Element **param** umožňuje nastavení důležitých informací o kategoriích nebo zboží. Obsahuje atribut **name**, kde je uloženo jméno a podelementy **value** a **value-domain**. Jméno elementu také může mít speciální význam. Pokud začíná na '`s-`', myslí se tím parametry pro server (server-side), pokud na '`c-`', tak klientské parametry (client-side) a parametry začínající na '`n-`', jedná se o parametry síťové.

3.4.2.3 Element product

Element **product** může obsahovat parametry – **param**, atributy – **attr** a odkazy – **link**. Musí obsahovat atribut **name**, který určuje unikátní ID produktu. Dále může obsahovat atribut **path**, který určuje URL zboží (pro doménu `btb.martsoft.com` je definováno URL jako `ocp://btb.upyp.com/-/A/B/001`). Dále element **product** obsahuje seznam atributů, definovaných

[6] Ukázka OCF: http://www.martsoft.com/ocf/sample1-ocf2.ocf

pro kategorii, a nastavuje jejich hodnoty. Pokud jsou jejich hodnoty prázdné, produkt tyto atributy nepoužívá. Elementy **param** a **link** nastavují parametry resp. odkazy na produkty.

3.4.2.4 Element category

Element **category** obsahuje strom kategorií. Tyto kategorie dále obsahují atributy, parametry, produkty a samozřejmě také další podkategorie. Každá z kategorií musí mít atribut **name**, který nastavuje jméno kategorie (např. pro počítačové programy je název `software`). Lze dále nastavit unikátní identifikátor kategorie (atribut `id`) a URL kategorie (atribut `url`).

V elementu **category** se dále nastavuje seznam atributů (jména a hodnoty - element **attr**), ale ne jejich hodnoty, ty nastavuje element **product**. Podelementem **param** lze nastavit speciální parametry kategorie a podelementem **link** URL jako odkaz na informace o kategorii.

3.4.2.5 Element attr

Elementem **attr** lze nastavit vlastnosti produktů. Obsahuje atribut `name`, kterým lze nastavit jméno, a atribut `style`. Jeho podelementy **value** a **value-domain** obsahují jednotlivé vlastnosti produktů.

3.4.2.6 Element value

Elementem **value** jsou definovány typy pro atributy a parametry. Ty lze rozdělit do následujících skupin:

jednoduché hodnoty – nejjednodušší typy definované v OCF. Jsou předdefinována čísla integer, boolovské hodnoty (boolean), čísla v plovoucí řádové čárce (float), jednotlivé znaky (char) a řetězce (string). Př:

```
<attr name="size" valuetype="integer" value="5" />
```

výčtové hodnoty – obsahují seznam jednoduchých hodnot

intervalové hodnoty – obsahují škálu možných hodnot proměnných. Př.:

```
<attr name="weight" valuetype="float_i" value="[2.0,4.0)" />
```

uživatelem definované – uživatel si může nadefinovat další typy proměnných. Lze tak například používat hodnoty, vyjadřující čas. Př.:

```
<attr name="date" valuetype="date" value="9.2.2003 12:01" />
```

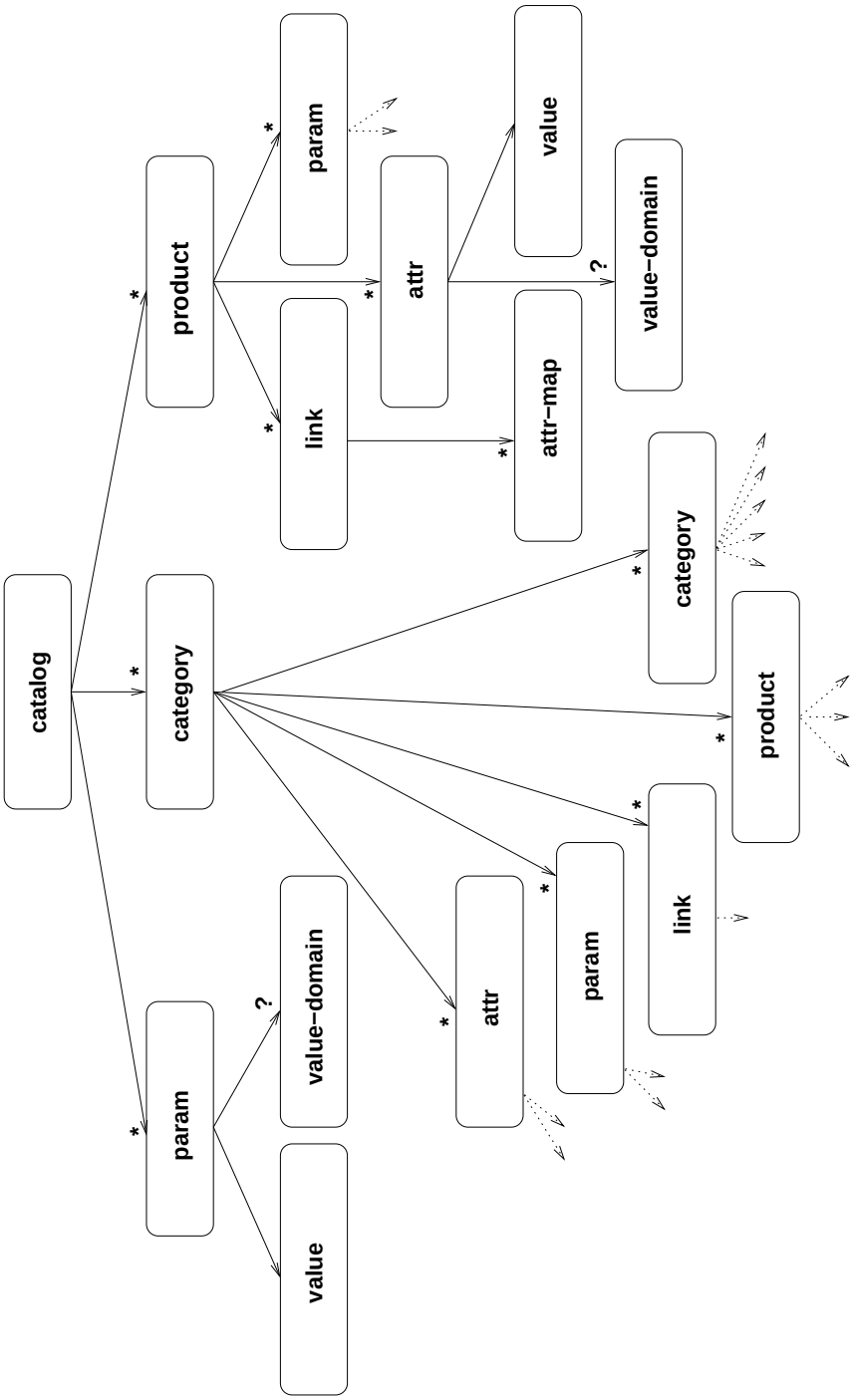


Figure 3.8: OCF: Graf struktury elementu Catalog.

3.4.3 Závěr

Definice jazyka OCF je velmi volná, tak jak ji firma MartSoft nabízí, ale lze vytvořit takový katalog, jaký je přesně potřeba. Pro práci s ním bylo vytvořeno několik dalších důležitých definic (jako je *Open Catalog Protocol* a *Open Catalog Query Language*), které další práci usnadňují. Výhodou je, že se ve všech případech jedná o jazyk XML, tudíž pro přenos všech příkazů protokolu OCP stačí jakýkoli znakově orientovaný protokol (například HTTP nebo STP - Simple Trafic Protokol). Podrobnější popis všech možností OCP a OCF lze nalézt na stránce [16].

Chapter 4

Porovnání vybraných formátů

4.1 Tabulky porovnání

4.1.1 Práce s položkami katalogu

Jednou z důležitých věcí je práce s položkami katalogu. V tabulce 4.1 je porovnání příkazů, sloužících k přidávání a odebírání položek z resp. do katalogu. V cXML je přidávání resp. odebírání řešeno elementem **ItemIndexItemAdd** resp. **ItemIndexItemDelete**. V hierarchii pod těmito elementy následuje vlastní seznam zboží. V jazyce xCBL je toto řešeno, že každé zboží (**product**) má element **Action**, který přidání (**Add**) nebo odebrání **Delete** nastavuje. Pokud je prázdný, bere se jako defaultní akce přidání zboží.

4.1.2 Porovnání důležitých elementů

V tabulce na straně 32 následuje porovnání důležitých elementů jednotlivých formátů. V řádcích tabulky je vždy tučně vytištěno jméno elementu v jednotlivých formátech, pod ním je jeho cesta z kořenového elementu (podle standardu XPath).

V tabulce jsou dále u elementů, které mají určený typ podle určité normy, je tato norma zapsána. Většinou jsou normy shodné, ale například u ceny za jednotku zboží formát OCF používá svůj vlastní typ, nadefinovaný jako atribut elementu **attr**.

Akce	**cXML element**	**xCBL element**
Přidání do katalogu	`//ItemIndexItemAdd`	`<Action>Add</Action>`
Vymazání z katalogu	`//ItemIndexItemDelete`	`<Action>Delete</Action>`

Table 4.1: Porovnání práce s katalogy

cXML element	xCBL element	OCF element	Poznámka

Description
`//ItemDetail/Description`[1]
	ProductName
	`//Product/ProductName`[2]
		`//product/attr[@name="Description"]`
		Příklad nastavení:
		`<attr name="Description">`
		`<value xsi:type="xs:string">nazev</value>`
		`</attr>`
			jméno výrobku

ShortName `//ItemDetail/Description/ShortName`	**ShortDescription** `//Product/ShordDescription`	`//product/attr[@name="ShortDescription"]`	krátký popis výrobku
ManufacturerName `//ItemDetail/ManufacturerName`	**Manufacturer** `//Product/Manufacturer`	`//product/attr[@name="ManufacturerName"]`	krátký popis výrobku

UnitPrice (ISO 4217)
`//ItemDetail/UnitPrice`
	ProductPrice (ISO 4217)
	`//Pricing/ProductPrice`
		`//product/attr[@name="Price"]`
		Příklad nastavení:
		`<attr name="Price" style="unit:dollar">`
		`<value xsi:type="xs:double">12.0</value>`
		`</attr>`
			cena za jednotku zboží

UnitOfMeasure
(UN/CEFACT, dop. 20)
`//ItemDetail/UnitOfMeasure`
	UOM
	(UN/CEFACT, dop. 20)
	`//Product/UOM`
		`//product/attr[@name="UOM"]`
		(UN/CEFACT, dop. 20)
		Příklad nastavení:
		`<attr name="UOM" required="true">`
		`<value xsi:type="xs:string">EA</value>`
		`</attr>`
			použité měrné jednotky zboží

LeadTime `//IndexItemDetail/LeadTime`	**LeadTime** `//Product/LeadTime`	`//product/attr[@name="LeadTime"]`	datum, do kdy je možné zboží dodat
ExpirationDate (ISO 8601) `//ItemDetail/ExpirationDate`	**ValidUntil** `//Product/ValidUntil`	`//product/attr[@name="ExpirationDate"]`	datum, do kdy je zboží dostupné
EffectiveDate (ISO 8601) `//IndexItemDetail/EffectiveDate`	**ValidFrom** `//Product/ValidFrom`	`//product/attr[@name="EffectiveDate"]`	datum, od kdy je zboží dostupné

Table 4.2: Porovnání důležitých elementů formátů

[1] Cesta zkrácena (podle XPath, str. 9), celý strom je v tabulce 3.4 na straně 16
[2] Cesta zkrácena (podle XPath, str. 9, celý strom je v tabulce 3.6 na straně 22

4.1.3 Kategorizace zboží

Zboží je výhodné zařadit do kategorií, do kterých logicky patří. Toto roztřídění je výhodné pro vyhledávání požadovaného zboží, ale také pro orientaci zákazníků.

4.1.3.1 Kategorizace v cXML

Datový formát cXML používá pro kategorizaci element **Classification**, jehož atribut **domain** určuje typ mapování kategorií. Je vyžadováno použití kategorií tak, jak je definuje standard ***UNSPSC*** v [12].

```
<Classification domain="SPSC">5136030000</Classification>
```

4.1.3.2 Kategorizace v xCBL

V xCBL jsou kategorie definovány pomocí elementu **CatalogSchema**. Jedná se opět o velmi robustní definici, jejíž popis by zabral mnoho místa. Zboží v elementu **Product** má pak v podelementu **SchemaCategoryRefList** nastaven seznam kategorií, do kterého patří.

4.1.3.3 Kategorizace v OCF

Kategorizace zboží se ve formátu OCF používá element **category**, tyto elementy tvoří hierarchickou stromovou strukturu, která definuje nadřazené a podřazené kategorie. Jejich název je definován v atributu **name** a nadřazené kategorie jsou zapsány v elementu **path**. Jednotlivé zboží je pak zařazeno v příslušné kategorii, název zboží je v atributu **name** a cesta stromem ke zboží (tedy kategorie) jsou v **path**.

Kategorie jsou také definovány pro jednotlivé zboží dle standardu UNSPSD (viz. [12]). Ve výpisu 4.3 je ukázka části souboru katalogu s právě popsanými elementy. Element **link** umožňuje jednoduše definovat křížové odkazy mezi kategoriemi.

4.2 Možnost převodu

Možný postup převodu se zdá přes XSL transformaci, ale je třeba navrhnout dva stylesheety (pro převod z jednoho formátu do druhého a nazpět). Z popisu formátů je zřejmé, že vzhledem k používaným standardizovaným hodnotám např. měn a měrných jednotek, lze tyto hodnoty transformovat velmi jednoduše (v podstatě se jedná pouze o změnu názvu elementu a

```
<category name="Mac Software"
 path="/Computer+Related/Software/Mac+Software/">

<link value="/Computer+Related/Computers/Macs/"
 valuetype="CrossSell"/>

<product name="3ae7c5"
 path="/Computer+Related/Software/Operating+Systems/3ae7c5">

<attr name="UNSPSC" style="disp-name:UNSPSC CODE;">
  <value xsi:type="xs:string">43160000</value>
</attr>
```

Table 4.3: Kategorie v OCF

jeho atributů). Složitější to bude pravděpodobně např. s kategoriemi zboží, které musí být shodné a pravděpodobně to XSL transformací nepůjde snadno zařídit.

Další nevýhodou při použití tohoto převodu je nutnost obsáhnout všechna zákoutí formátů. To znamená, že tyto stylesheety budou velmi obsáhlé a jejich ladění náročné.

Chapter 5

Závěr

V této práci jsem se pokusil porovnat několik používaných datových formátů v e-Commerci. Vzhledem k tomu, že většina formátů je zapsaná v XML, bylo vhodné popsat také tento standard a další používané standardy, které jsou pro práci s XML potřebné.

Z popisu jednotlivých formátů je zřejmé, že jsou navrženy robustně, ale některé (xCBL) až příliš. Proto si myslím, že návrh katalogu v tomto formátu musí být velmi časově náročný. Oproti tomu se zdá, že formát OCF by se brzy mohl stát favoritem v této oblasti, a to pro jeho jednoduchost, a také pro nabízenou možnost převodu do dalších používaných formátů.

Bibliography

[1] http://www.w3.org

[2] http://www.w3.org/TR/xmlschema-0/

[3] http://www.w3.org/TR/REC-xml

[4] http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/ 2.1 2.4 2.1 2.4.2

[5] http://www.w3.org/TR/xpath

[6] http://www.zvon.org/xxl/XPathTutorial/General/examples.html 2.4.1 2.4.1

[7] http://www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html http://www.unece.org/cefact/rec/rec03en.htm

[8] http://www.xe.com/iso4217.htm

[9] http://www.mcs.vuw.ac.nz/technical/software/SGML/doc/iso8601/ISO8601.html

[10] http://www.unece.org/cefact/rec/rec20en.htm

[11] http://www.faqs.org/rfcs/rfc1766.html http://www.w3.org/WAI/ER/IG/ert/iso639.htm 3.2.3.3, 3.3.1.5 3.2.3.2, 3.3.1.2, 3.3.1.4 3.2.3.3 3.2.3.2, 3.3.1.4, 3.3.1.5 3.3.1.2, 3.3.1.5

[12] http://www.unspsc.org 4.1.3.1, 4.1.3.3

[13] http://www.xcbl.org/xcbl40/index.html

[14] http://www.cxml.org 3.2.2

[15] http://www.martsoft.com/ocp/semantics.htm

[16] http://www.martsoft.com/ocp/ 3.4.1 3.4.2, 20, 3.4.3

Index

List of Tables

List of Figures