Masarykova univerzita

Sborník příspěvků

4. letní škola aplikované informatiky

Editor

Jiří Hřebíček

Bedřichov, 24.-26. srpna 2007

ISBN 80-210-4146-3

Vážení čtenáři,

právě držíte v rukou sborník příspěvků 4. letní školy, která je již tradičním místem setkání řady lidí, jejichž společným pojítkem je aplikovaná informatika. Otcem myšlenky a hlavním organizátorem této akce pořádané na sklonku léta je profesor Jiří Hřebíček. Ačkoli celá letní škola probíhá v neformálním, a leze říci až rodinném prostředí, její odborná úroveň je přesto vysoká, o čemž svědčí i tento sborník.

Necháme-li promluvit řeč binárních čísel, která jsou nám informatikům tak blízká, zjistíme, že letní škola dovršila kulaté 100. výročí. Děje se tak v roce, kdy své šedesátiny slaví i její zakladatel. Netroufáme si na tomto místě uvést výčet všech významných činností, působišť a výsledků profesora Hřebíčka. Jen namátkově uvedeme klíčový podíl na uspořádání konferencí EnviroInfo 2005 v Brně, ISESS 2007 v Praze a TIES 2007 v Mikulově, což všechno byly vrcholné vědecké akce, které úspěšně reprezentovaly Českou republiku ve světě. Vedle aktivního působení v řadě mezinárodních společností, z nichž zde zmiňujeme IFIP (InternationalFederation for Information Processing) nebo EMSS (Environmental Modelling and Software Society), je jméno profesora Hřebíčka spojeno i s dlouholetou prací na mnoha vysokých školách, jako jsou Masarykova univerzita, Mendelova zemědělská a lesnická univerzita, Vysoké učení technické v Brně nebo Vysoká škola báňská – Technická univerzita Ostrava.

Je nám ctí, že můžeme říci, že během let se z našeho učitele profesora Hřebíčka stal i náš dobrý přítel Jiří, kterému děkuje za možnost být alespoň z části při tom všem úchvatném dění. Letní škole i jejímu duchovnímu otci přejeme mnoho dalších úspěšných a tvůrčích let.

Tomáš Pitner a Jaroslav Ráček

Obsah

Jan Pavlovič
Matematický model pro dekontaminační strategii nebezpečných odpadů 6

Jana Sedláčková
Využití CSP při odhadování nákladů workflow procesů 12

Pavel Kříž
Maple jako asistent při výuce matematiky 19

Zuzana Chvátalová
Podpora aplikací matematických disciplín systémem Maple při řešení současných problematik globálního charakteru 25

Matěj Štefaník
Současné metody vyhledávání informací a prezentace jejich výsledků 40

Jan Ministr
Metodický postup zpracování rámcového procesního modelu 48

Martin Pochyla
Metodický rámec implementace servisně orientované architektury 55

Vítězslav Novák
Návrh metodického postupu použití objektových databází při tvorbě webových aplikací na platformě Java Enterprise Edition 68

Petr Rozehnal
Metodický rámec tvorby multimediálních aplikací 81

Karel Kisza
Standardizace modelu pro environmentální data, informace a znalosti 92

Miroslav Kubásek
Service Oriented Approach for Accessible Publishing of Environmental Information in Web Environment 99

Michal Hejč
Hromadné zpracování neurčitých dat 111

Tomáš Ludík, Jaroslav Ráček
Modelovanie procesov v krízovom riadení 122

Matematický model pro dekontaminační strategii nebezpečných odpadů

Jan Pavlovič

Masarykova univerzita, Fakulta informatiky,
Botanická 68a, 602 00 Brno
pavlovic@fi.muni.cz

Abstrakt
Příspěvek popisuje problematiku dekontaminační strategie z pohledu matematické formalizace modelu systému. Dále jsou v článku uvedeny základní rysy dekontaminačního procesu, zejména z pohledu souvislostí mezi hodnocením ekologických rizik a modelováním řešení dekontaminační strategie.

Abstract
This paper describes decontamination strategy problem domain. Mainly from the mathematical formalization of model of system point of view. There are describe the main aspect of the decontamination process. The relations between ecological risk assessment and decontamination strategy modeling.

Klíčová slova
3MRA, ekologická rizika, dekontaminační strategie, matematické modelování, nebezpečné odpady.

Keywords
3MRA, ecological risks, decontamination strategy, hazardous waste, mathematical modeling.

1 Úvod

Výsledkem mnoholetého špatného zacházení s přírodou jsou lokace zamořené toxickými látkami. Například jen v USA to byla tisíce neevidovaných a nezabezpečených vysoce toxických znečištěných území[1]. Stejně tomu bylo i v České republice, kde jsou po odchodu sovětských vojsk mnohá území zamořena dodnes nezjištěnými vysoce toxickými látkami, jejichž dopad na environment není plně znám.

Likvidace takto znečištěných území je komplikovaný, extrémně drahý a energeticky náročný proces. V současné době je v USA evidováno na 500 znečišťujících látek a na druhé straně na 130 technologií, které umožňují jednotlivé kontaminanty z daného území odstranit [1].

[1] zdroj: Comprehensive Environmental Response Compensation, and Liability Act 2003

Zjišťování vhodné kombinace a nastavení technologií pro vyčistění daného území představuje netriviální úlohu. Vše závisí na mnoha parametrech, což v praxi představuje obrovské množství kombinací.

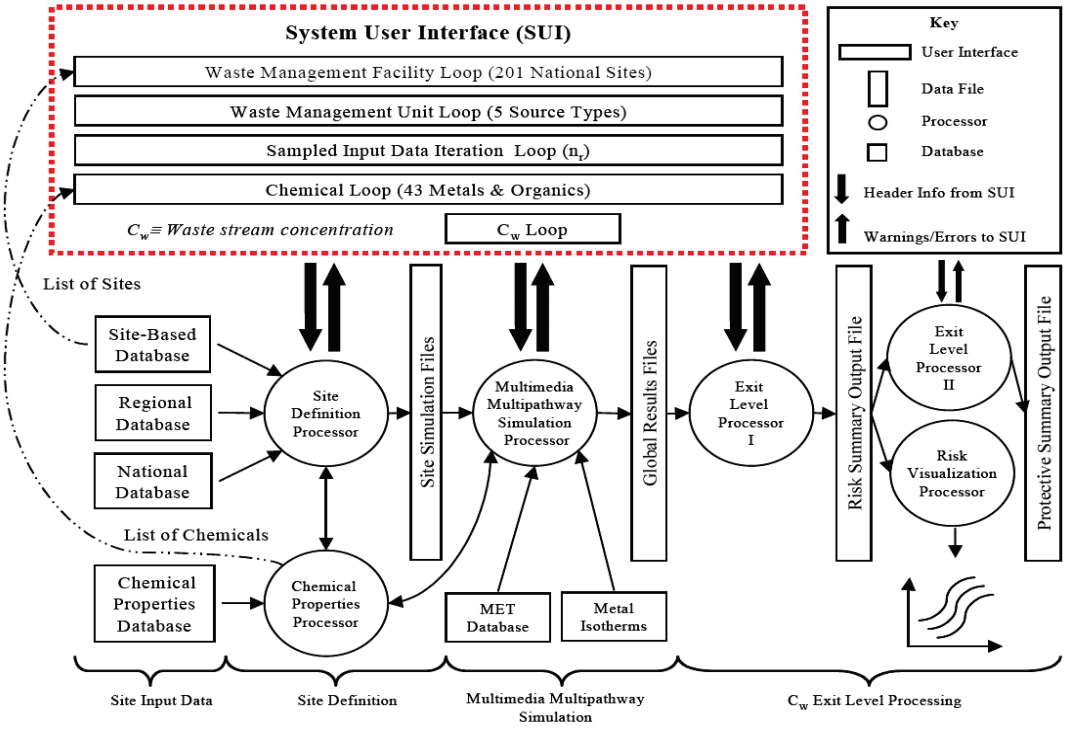
V průběhu úspěšné spolupráce mezi PPRC (Pacific Northwest Pollution Prevence Resource Center) U. S. EPA (Environmental Protection Agency) USA a Fakultou informatiky Masarykovy university na environmentálním kalkulátoru [2], jsem byl požádán o navržení teoretického postupu a technického řešení, jak vyřešit otázku multikriteriálního výběru optimálního sledu dekontaminačních technologií. A tím, s využitím moderních poznatků teoretické informatiky, napomoci v rozhodovacím procesu výběru technologií.

2 Hodnocení ekologických rizik

Koncepce hodnocení zdravotních a ekologických rizik vychází z materiálů vypracovaných US EPA v letech 1983-1987 a počátkem devadesátých letech byla přijata jako základ dokumentů EU. V roce 1997 EPA Office of Solid Waste (OSW) a Office of Research and Development (ORD) začali společně pracovat na vývoji nového modelovacího systému: Multimedia, Multipathway, and Multireceptor Risk Assessment (3MRA) [4]. Mezi základní princi 3MRA patří zejména:

- Řízením se paradigmatu rizika a nejnovějších EPA pokynů a doporučení;
- Je založen na výsledcích soudobého výzkumu a „state-of-the-art“ modelovacích technikách;
- Obsahuje multimédia a multipathway hodnocení ekologických rizik;
- Obsahuje hodnocení dopadu ekologických rizik na lidské zdraví;
- Umožňuje zařazení nových vědeckých poznatků a softwarových komponent;
- Reflektuje řízení kvality, kontrolních metod a reprodukovatelnost;
- Je porovnatelný s jinými analytickými řešení a numerickými modely.

Hlavní idea 3MRA modelovacího systému reflektuje jeden z nejdůležitějších cílů OSW: určování kdy je koncentrace chemických látek zásadních rizikem pro lidské zdraví a environment. 3MRA modelovací systém je zamýšlen jako EPA systém pro podporu hodnocení ekologických rizik nové generace. EPA původně navrhla systém pro potřeby OSW programů, nicméně jej lze využít i pro jiné oblasti. Na příklad byl navržen pro poskytovaní informací managerům o ekologickém riziku a jeho dopadu na lidské zdraví. Z pohledu koncentrace nebezpečných průmyslových odpadů. Na obrázku 1. je znárodněna základní funkcionalita 3MRA systému.

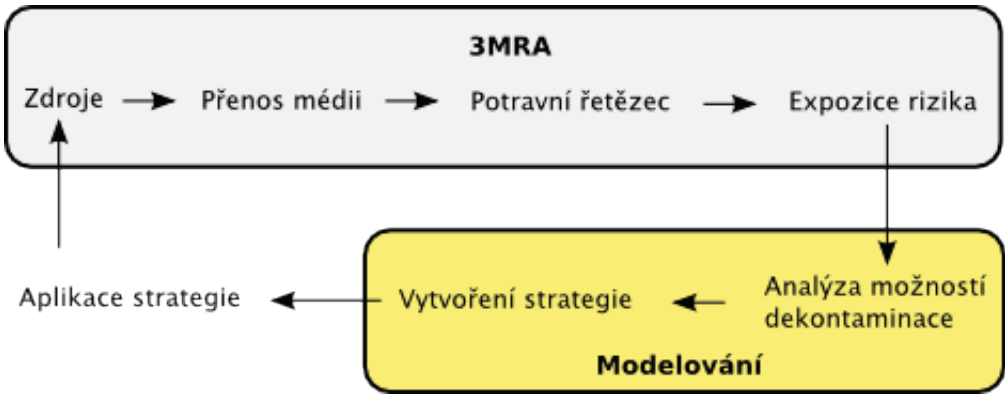


Obr 1: Design 3MRA modelovacího systému.

3 Dekontaminační proces

Hodnocení ekologických rizik a jejich dopad na lidské zdraví je základní funkcionalitou pro další opatření na odstranění znečistění. Nicméně představuje pouze část v celém systému. Na obrázku 2. je znázorněn celý cyklus dekontaminačního procesu. EPA systém 3MRA se zabývá modelováním od zdroje znečištění až po vyhodnocení rizika.

Na základně hodnocení ekologického rizika a jeho dopadu na lidské zdraví je nutné provést protiopatření a stresory z dané lokace odstranit. Nejprve namodelujeme danou problémovou doménu. Tím zjistíme možné dekontaminační postupy, kterou jsou zejména závislé od media, kde se dané kontaminanty vyskytující, jejich koncentrace a počtu druhů. Tento model je matematicky popsán. Za základě těchto údajů a priorit dekontaminace, které do systému vloží expert dojde k namodelování dekontaminační strategie, která nabízí možné varianty, jak danou



Obr 2: Cyklus dekontaminace

lokaci dekontaminovat. Po aplikaci strategie dochází k opětovnému vyhodnocení rizik pomocí systému 3MRA.

4 Analýza možností dekontaminace

Nejprve si formalizujeme model systému celé dekontaminační problematiky. Poté zaneseme do modelu omezení, které se v systému vyskytují. Dále kritéria na výslednou strategii, která parametrizují proces hledání dekontaminační strategie z pohledu expertních priorit.

4.1 Model systému

Do systému nám vstupují následující veličiny.

U_l, l = 1,.., u – Velikost znečistěného území

Z_h, h = 1,…, z – Typ znečištěného území dle 3MRA

M_i, i = 1,…, m – Typ znečistěného média

P_j, j = 1,…, p – Skupina kontaminantů

K_f, f = 1,…, f – Koncentrace kontaminantů

G_t, t = 1,…, g – Clean up goal

T_k, k = 1,…, k – Dekontaminačních technologií

C_y, y = 1,…, c – Konstanty dekontaminačních technologií

V_b, b = 1,…, v – Výstupní parametry dekontaminačních technologií

D_a, a = 1,…, d – Vstupní parametry dekontaminačních technologií

H_r, r = 1,…, h – Hodnoty parametrů dekontaminačních technologií

S_q, q = 1,…, s – Vstupní kritéria strategie

Na (M_i, P_j) lze použít T_k, kde k je přípustné.

Modelem technologie rozumíme soustavu parametrů (vstupní parametry, výstupní parametry, hodnoty parametrů, konstanty). $T_k = (V_b, D_a, H_r, C_y)$.

5 Vytvoření dekontaminační strategie

Problém dekontaminační strategie je obecně výpočetně náročný problém. Naivní řešení můžeme zapsat jako: $V = (T!) \cdot H^{(P \cdot T)}$

V – velikost prostoru řešení

T – počet technologií

P – parametry technologií

H – hodnoty parametrů

Pro modelovou situaci, kdy máme k dispozici 15 technologií. Dále Uvažme, že každá z těchto technologií má přibližně tři volné parametry. Jelikož by se spojitými hodnotami parametrů vznikl nekonečný prostor řešení, zjednodušme úvahu ještě tím, že řekneme, že průměrný parametr může mít sedm možných hodnot. Tím se řešení dostane do hodnost 10^{50}. Což je bez efektivního algoritmu neřešitelný problém pří použití hrubé síly.

Cílem dekontaminační strategie je vytvoření posloupnosti přípustných technologií s danými hodnotami parametrům, které při daných hodnotách znečistěného území minimalizují množství kontaminantů v území s ohledem na vstupní kritéria.

Formálně je dekontaminační strategie hledání minima funkce kriteria na posloupnosti přípustných technologií.

Nechť $f(x)$ je funkce kritéria, M je množina možných řešení. $f(x)$: $M \rightarrow R$ na množině M je $y \in M$ takový, že $f(y) \leq f(x)$ pro všechna $x \in M$.

Algoritmus předpokládá vytvoření sekvence technologií tak, aby pro všechny kontaminanty byla jejich koncentrace minimální ve vztahu k „cleanup goal“. Dojde k vybrání technologií z množiny přípustných technologií pro dané médium a kontaminant. Nastavení parametrů technologie, tak aby kontaminanty byly minimální a zároveň výstupní parametry technologie odpovídaly vstupním kriteriím. Technologie se zařadí do sekvence. Formální zápis algoritmu:

M:= empty

min:= cleanup goal

do until $\forall K_f \in P_j$ = min

for k $\in T_k$; kde T_k je přípustná pro (M_i, P_j)

vypočítej (V_b, H_r) tak, aby K_f = min and $V_b = S_q$

$M \leftarrow T_k$

uspořádej posloupnost M

done

6 Závěr

Pro úspěšné vyřešení problematiky dekontaminační strategie bylo nutné nejprve formalizovat modelu systému. Následně ve zvoleném systému navrhnout algoritmus pro výběr sekvence přípustných technologií s danými hodnotami parametrům, které při daných hodnotách znečištěného území minimalizují množství kontaminantů v území s ohledem na vstupní kritéria.

Dekontaminační strategie slouží jako podpora v rozhodování při likvidaci znečištěného území, které má negativní vliv na lidské zdraví. Jelikož je řešení dekontaminace komplexní a nákladný problém, přináší formalizace potencionální zefektivnění celého procesu a umožňuje výpočet řešení, které by bylo jinak výpočetní obtížně řešitelné [3].

7 Literatura

[1] Mahutová, K., Pavlovič, J. Energy Management at Waste Clean up Sites, Avoiding Secondary Air Impacts to Human Health, First Biennial Central & Eastern European Environmental Health Conference., 2004.

[2] Pavlovič, J. Multidimensional Decision Support Tool for Remedial Technologies, In Proceedings of the ENVIROINFO BRNO 2005 – Informatics for Environmental Protection., 2005.

[3] Pavlovič, J., Hřebíček, J. Finding Optimal Solutions of Energetic Remedial Equations with Genetic Algorithms, Enviroinfo2006., 2006.

[4] U. S. EPA Center for Exposure Assessment Modeling. The Multimedia, Multi-pathway, Multi-receptor Exposure and Risk Assessment. http://www. epa.gov/., 2003

Využití CSP při odhadování nákladů workflow procesů

Jana Sedláčková

Vysoké učení technické v Brně
Fakulta informačních technologií
Božetěchova 2, 612 66 Brno
jana.sedlackova@gmail.com

Abstrakt
Příspěvek představuje myšlenku, jak by bylo možné využít formální specifikaci komunikujících sekvenčních procesů (CSP) v oblasti softwarového měření. Přesněji jak využít CSP při stanovení odhadů nákladů na vyvíjené workflow procesy. Pro odhadování nákladů bude v příspěvku použita metoda funkčních bodů FPA a tento postup bude následně prezentován na konkrétním příkladu workflow procesu.

Abstract
This paper describes an idea how to take use of communicating sequential processes (CSP) formal specification in the software measurement. Concretely how to use CSP for cost estimation of developing workflow processes. In this paper for the cost estimation will be used FPA - function point analyses method and presented it on one practical example of workflow process.

Klíčová slova
Komunikující sekvenční procesy (CSP), měření, metoda funkčních bodů (FPA), workflow procesy, diagramy chování.

Keywords
Communicating sequential processes (CSP), Measurement, Function point analyses method (FPA), Workflow processes, Behaviour diagrams.

1 Úvod

Procesní přístup k řízení institucí a podniků se stává stále běžnějším. Nejde jen o velké firmy a společnosti, jež na tento přístup postupně přecházejí, ale i o střední a malé firmy a instituty veřejné správy. Tento způsob vytlačuje starší techniku funkčního pohledu, jež dělila práci mezi funkční jednotky vytvořené na základě příslušných dovedností. Takto vzniklé jednotky vytvářely jen svoji konkrétní činnost nezávisle na celkovém výsledku.

Podobně jako u vývoje informačních systémů i workflow projekty sebou přinášejí nemalá finanční rizika spojená s jejich vývojem. Je třeba co nejpřesněji analyzovat náklady, které sebou vyvíjený produkt přináší. Pro stanovení odhadů nákladů budeme v tomto příspěvku používat

metodu funkčních bodů (FPA). Při odhadování se však pokusíme využít formálního jazyku komunikujících sekvenčních procesů CSP, jímž workflow proces popíšeme.

2 Charakteristika CSP

CSP (Communicating Sequential Proccess) je formální jazyk, jež popisuje chování a vzájemnou interakci v souběžných systémech. Tato algebra je podporována různými softwarovými nástroji, které umožňují analýzu a verifikaci systémů. CSP model je založen na myšlence několika regulárních sekvenčních procesů, které běží paralelně. Během svého trvání může proces provádět různé události nebo akce. Tyto události jsou pak viditelnou částí chování procesu.

Množina událostí, které daný proces využívá, se nazývá abeceda procesu, nebo také interface procesu. Během aktivace procesu může událost z abecedy nastat jednou, vícekrát a nebo nemusí nastat vůbec. Do abecedy procesu se mohou vložit jen ty události, které zachycují aspekty chování procesu, které nás zajímá.

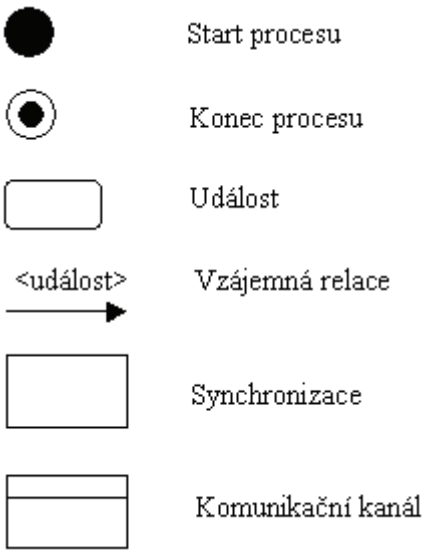
Nejjednodušší možné chování má proces, který nedělá nic. Takový proces se zapisuje jako STOP. Pokud chování nějakého systému dospěje do tohoto procesu, znamená to, že nastal deadlock - proces nic nedělá a v této činnosti pokračuje donekonečna. Kromě procesu STOP existuje ještě jeden předdefinovaný proces SKIP. Podobně jako proces STOP nedělá nic, ale na rozdíl od něj se ukončí. Používá se pro indikaci korektně ukončeného chování procesu.

Základní CSP algebraické operátory:

- $a \rightarrow P$ - operátor prefix, provede se vstupní nebo výstupní událost *a* a poté se přejde do procesu *P*
- $(a \rightarrow P) \,\square\, (b \rightarrow Q)$ - deterministický výběr větvení běhu podle událostí
- $(a \rightarrow P) \sqcap (b \rightarrow Q)$ - nedeterministický výběr větvení běhu podle událostí
- $P \,|||\, Q$ - operátor prokládání, nezávislá paralelní kompozice
- $(a \rightarrow P) \,||\{\alpha\}||\, (a \rightarrow Q)$ - paralelní rozhraní, paralelní kompozice se sdílenou událostí *a*
- $(a \rightarrow P) \setminus \{a\}$ - operátor skrytí události *a*

3 Diagramy chování

Pro popis chování systému lze rovněž využít tzv. diagramů chování, jež jsou založeny na UML specifikaci. V UML specifikaci tyto diagramy popisují jednotlivé stavy systému a události, které vyvolávají přechod z jednoho stavu do jiného. Tento způsob zápisu je možné použít pro vizuální zápis specifikace v jazyce CSP. V diagramech chování jsou zobrazeny přímo sledy událostí místo stavů systému. Jsou používány následující grafické symboly:



- *Start procesu* – symbol reprezentující začátek chování procesu
- *Konec procesu* – symbol reprezentující ukončení chování aktuálního procesu a specifikuje zároveň následující proces určující další chování systému

[1] *Událost* – objekt specifikující událost, kterou daný proces provedl

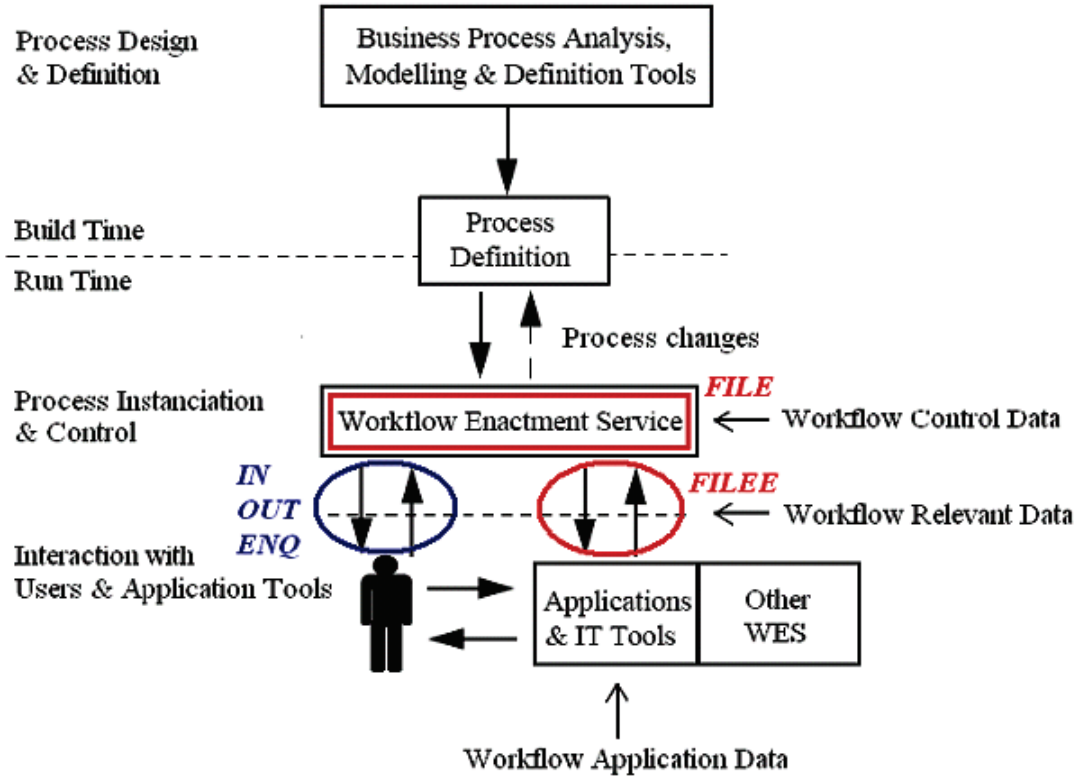
- *Vzájemná relace* – symbol pro vzájemnou relaci ukazující vztah mezi jednotlivými událostmi a procesy
- *Synchronizace* – objekt pro vyjádření vzájemné interakce procesů, obsahuje seznam událostí
- *Komunikační kanál* – vyjadřuje vzájemný vztah mezi procesy. Seznam událostí specifikuje typ komunikačního kanálu, tedy události které je kanálem možné zasílat

4 Stanovení odhadu nákladů workflow procesů metodou FPA

Metoda FPA zkoumá aplikační oblast a neměří zdrojový kód. Při samotném výpočtu odhadu nákladu workflow procesu je třeba proto nejprve spočítat tzv. neupravené funkční body, tedy identifikovat základní funkční typy workflow procesu (IN – externí vstupy, OUT – externí výstupy, ENQ – externí dotazy, FILE – vnitřní logické soubory, FILEE – soubory vnějšího rozhraní) a jejich příslušné složitosti. Počet neupravených funkčních bodů *NFP* je pak dán součtem všech váhových hodnot. Z neupravených funkčních bodů se dále počítají tzv. upravené funkční body, jež jsou přizpůsobeny danému konkrétnímu workflow procesu a prostředí, ve kterém se vyvíjí. Stanoví se vlivy 14ti faktorů obecných charakteristik systému včetně příslušné hodnoty vlivu na aplikaci. Tímto způsobem se vypočte faktor přizpůsobení a vynásobí hodnotou neupravených funkčních bodů. Tento výsledek dá počet upravených funkčních bodů

FP, na jehož základě se stanoví pracnost a cena jednoho funkčního bodu a dále i celkové náklady pro vývoj workflow procesu.

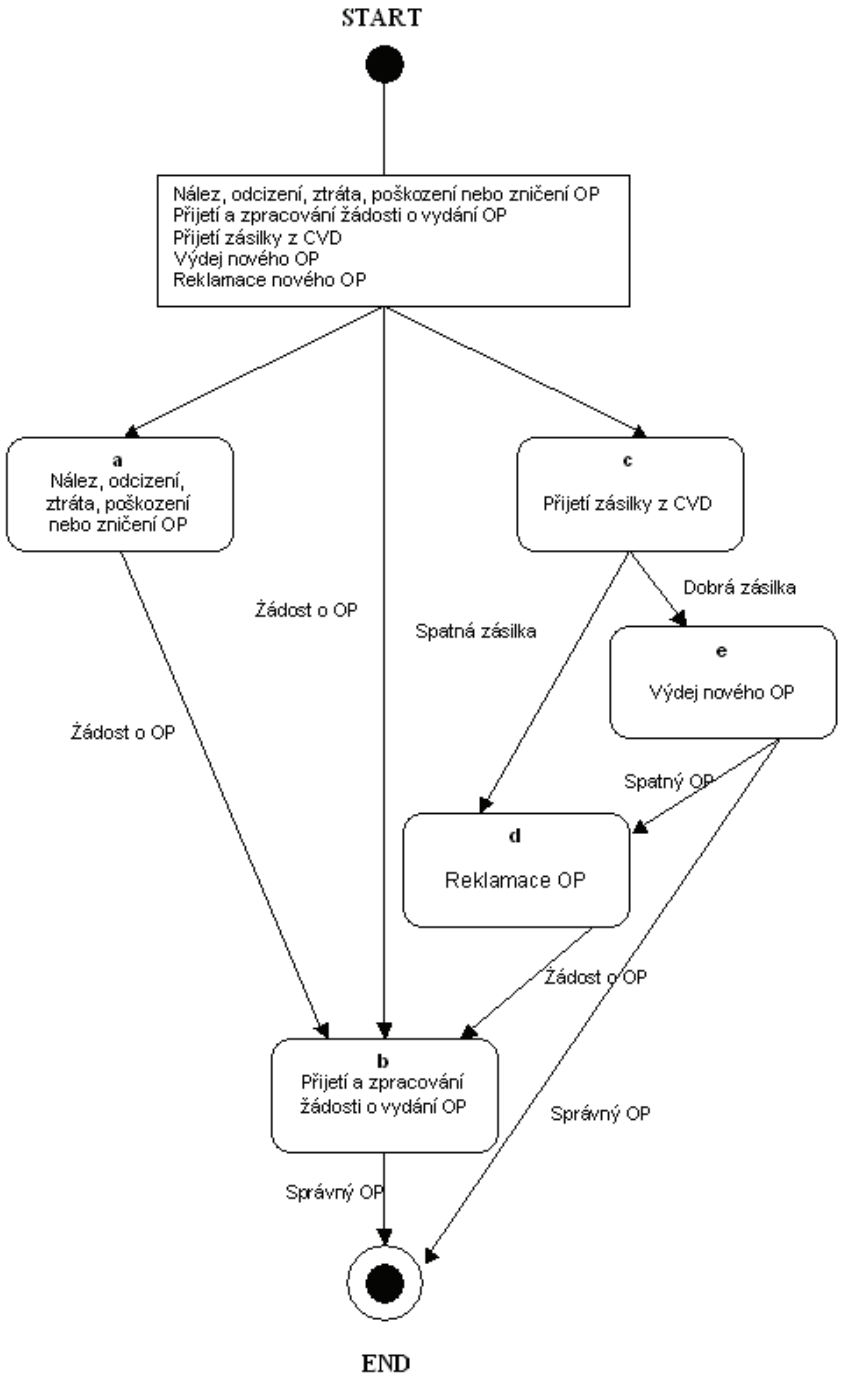
V případě odhadů workflow procesů je nejobtížnější identifikovat základní funkční typy – IN, OUT, ENQ, FILE, FILEE. Na následujícím obrázku, na němž je zobrazen meta-model workflow procesu, jsou vymezeny oblastí, kde můžeme tyto funkční typy rozpoznat.



Obr.1: Meta-model workflow procesu

5 Využití CSP pro popis workflow procesu

Nyní bude popsán workflow proces evidence a vydávání občanských průkazů. Tento workflow proces popisuje dílčí činnosti, podle kterých postupují pracovníci úseku občanských průkazů a evidence obyvatel. V úseku jsou evidovány a vydávány občanské průkazy a potvrzení o občanském průkazu. Proces na nejvyšší úrovni dekompozice obsahuje následujících pět činností: nález, odcizení, ztráta, poškození nebo zničení OP; přijetí a zpracování žádosti o vydání OP; přijetí zásilky z Centrální výroby dokladů (CVD); výdej nového OP; reklamace nového OP.



Obr.2: Diagram chování workflow procesu

Pro jednodušší zápis jsou události znázorněny symboly *a* – *e*. Nyní bude workflow proces evidence a vydávání OP zapsán pomocí CSP algebraických operátorů.

$$\begin{aligned} \mathrm{START} = \quad & (a \rightarrow b \rightarrow END) \\ & \square\ (b \rightarrow END) \\ & \square\ (c \rightarrow d \rightarrow b \rightarrow END)\ \square\ (c \rightarrow d \rightarrow e \rightarrow b \rightarrow END)\ \square\ (c \rightarrow e \rightarrow END) \end{aligned}$$

Nyní bude stanoven počet externích vstupů IN, externích výstupů OUT, externích dotazů ENQ, vnitřních logických souborů FILE a souborů vnějšího rozhraní FILEE a tyto budou ohodnoceny. Tím dostaneme počet neupravených funkčních bodů. Podle postupu, jež je uveden výše, bude stanoven počet upravených funkčních bodů a následně lze stanovit náklady na vývoj celého workflow procesu pro evidenci a výdej OP metodou FPA.

6 Závěr

V příspěvku bylo ukázáno, jak je možno využít formální specifikaci souběžných komunikujících systémů CSP v oblasti odhadování nákladů na vývoj workflow procesů. Pro odhadování byla použita metoda funkčních bodů FPA, jež se často používá pro stanovení odhadů nákladů klasických softwarových systémů.

Postup stanovení odhadu byl nastíněn na konkrétním workflow procesu pro evidenci a vydávání občanských průkazů. Pro popis tohoto workflow procesu byly použity diagramy chování, jež jsou založeny na UML specifikaci. Následně byl tento workflow proces rovněž popsán pomocí CSP algebraických operátorů.

Lze tedy říci, že i při odhadování nákladů na vývoj workflow procesů lze využít možností, které jsou nabízeny algebrou komunikujících sekvenčních procesů CSP.

7 Literatura

[1] Hoare, C.A.R: Communicating Sequential Processes, 2004.

[2] Bailey, J. W., Basili, V. R.: A meta-model for software development resource expenditures. USA, Piscataway, IEEE Press 1981.

[3] International Function Points User Group: Function Point Counting Practices Manual: Release 4.1. 1999.

[4] Bert C., Dumke R., Bundschuh M., Schmietendorf A.: Best Practices in Software Measurement, Springer, 2005.

[5] Sedláčková J.: Odhady času a úsilí workflow projektů, Rigorózní práce, FI MU Brno, 2005.

[6] Sčuglík F.: Uživatelské rozhraní formální specifikace vestavěných systémů, FIT VUT Brno, 2003.

[7] Brookes S.D., Hoare C.A.R, Roscoe A.W.: A Theory of Communicating Sequential Processes, Oxford University, England, 1984.

[8] Jacquet J.P., Abran A.: From Software Metrics to Software Measurement Methods: A Process Model, Universite do Quebec a Montreal, Canada.

Maple jako asistent při výuce matematiky

Pavel Kříž

Masarykova univerzita, Ústav matematiky a statistiky
Janáčkovo nám. 2a, 602 00 Brno
kriz@math.muni.cz

Abstrakt
Příspěvek je zaměřen na využití systému počítačové algebry Maple ve výuce matematiky. Popisuje historický vývoj samotného systému v souvislostech spojených s přímým začleněním do výuky. Ukazuje možnosti pro interaktivní zapojení Maple do výuky. V závěru se věnuje dostupným materiálům na internetu.

Klíčová slova
Výuka matematiky, Maple, Maplet, MapleNet

Abstract
The paper focuses on the use of computer algebra system Maple in teaching mathematics. It describes the historical development of the system itself in contexts associated with direct involvement in teaching. It shows the possibilities for interactive engagement Maple in teaching. In the end, we discuss the available material on the Internet.

Keywords
Education od mathematics, Maple, Maplet, MapleNet

1 Úvod

Obrovský rozmach informačních technologií v současné době značně ovlivňuje vývoj mnoha vědeckých disciplín a směrů. Jinak tomu není ani v matematice. Systém počítačové algebry Maple během svého vývoje zasáhl do mnoha matematických disciplín. V následujícím textu se zaměříme na významné mezníky ve vývoji systému Maple, které jsou úzce spojeny právě s využitím jeho možností ve výuce matematiky. Dále uvedeme aktuálně dostupné pomůcky pro studenty a učitele, jako jsou např. průvodci pro začínající uživatele Maple, výukové materiály pro seznámení s novým dokumentovým prostředím Maple a internetové portály zaměřené na tuto tématiku.

2 Systém Maple

Během vývoje systému Maple, jehož počátky sahají do počátku osmdesátých let minulého století, lze pozorovat odlišný přístup ve vývoji. Systém byl sice od začátku veden k co nejširšímu použití, s velikým důrazem na využití v oblasti vzdělání, nicméně hlavní cílovou skupinou byli především vědci.

Nástup Maple verze 8 s sebou přinesl dvě různá grafická uživatelská prostředí. Prostředí známé z minulých verzí bylo zachováno pod názvem „Classic worksheet“. Novinkou bylo prostředí, které mělo odstartovat novou etapu k vývoji. Zpočátku bylo jeho využití velmi sporadické, a to zejména díky své náročnosti na počítačové vybavení. Kromě „klasického“ rozšíření výpočetních schopností o nové knihovny byl kladen důraz i na jejich začlenění do výuky. Začíná vývoj knihovny *Student*, která je navržena jako pomůcka učitelům a studentům matematiky. Obsahuje řadu příkazů pro výpočty, výuku a grafické znázornění matematické teorie funkcí jedné proměnné (*Calculus1*), lineární algebry (*LinearAlgebra*) a středoškolské matematiky (*Precalculus*). Je zde kladen důraz nejen na konečný výsledek, ale i na postup řešení metodou „krok za krokem“. Jejich didaktický přínos je zřejmý.

Následující verze pokračují v započaté strategii a postupně odstraňují těžkopádnost a nestabilitu nového uživatelského rozhraní. Nechybí ani forma vytváření grafických aplikací – Maplets. Jejich implementace je však pracná a vyžaduje dostatečnou znalost programování a syntaxe Maple. Lze tedy tvrdit, že využití našla jen u skupiny zkušených uživatelů.

Maple verze 10 nabízí vytvářet matematické dokumenty s využitím interaktivního klikacího kalkulu. Jedná se o novou formu práce se systémem Maple. Není zde kladen takový důraz na syntax a znalost příkazů. Tímto postojem je tak schopen oslovit širší skupinu uživatelů v různých směrech zaměření. Verze 10 disponuje doplněnou verzí knihovny *Student*, rozšířenou o knihovnu pro výuku vektorového počtu *VectorCalculus* a knihovnu *MultivariateCalculus* zaměřenou na analýzu funkcí více proměnných. Součástí knihovny *Student* je řada řešených příkladů, vzorových dokumentů a výukových průvodců.

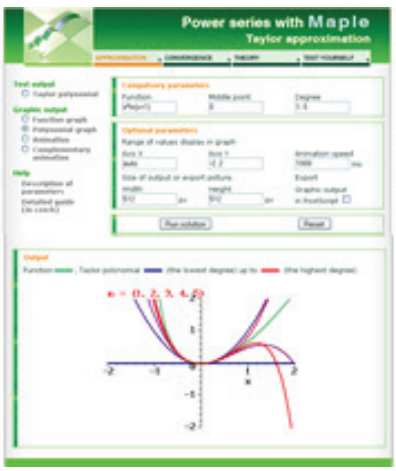
Aktuální verze Maple 11 nabízí novinky, které vnášejí do výuky zcela nový rozměr. Pro méně zkušené uživatele jsou k dispozici průvodci a nechybí ani množství řešených příkladů. Nedílnou součástí didakticky zaměřených knihoven jsou grafické nástavby příkazů, které poskytují uživateli větší komfort práce se systémem. Tyto nástavby jsou založeny na technologii Maplet. Uspořádání nápovědy společně s interaktivními nástroji působí kompaktním dojmem a jsou pro nového i zkušeného uživatele přínosem.

3 Interaktivní výuka se systémem Maple

V následujícím textu ukážeme nové přednosti systému Maple, přímé a interaktivní zapojení do výuky matematiky. Jako v předešlé kapitole, budeme postupovat chronologicky. Uvedeme metody pro interaktivní publikování matematiky na internetu v závislostech na možnostech, které nám Maple ve svém vývoji nabízel.

Počátky jsou spojené s grafickou interpretací či vizualizací mnohých matematických problémů. Jedná se o pouhé grafické výstupy buď ve formě obrázků či animací, které mnohdy doplňují běžný matematický text. Zde ovšem nelze hovořit o interaktivním přístupu.

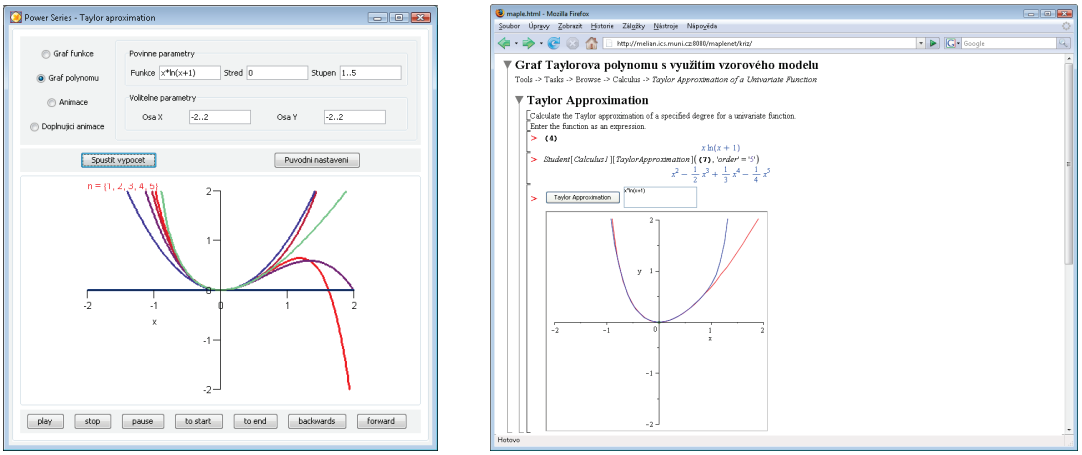
Slovo interaktivní lze použít až s metodou, která využívá webového serveru a serverové instalace systému Maple. Jde o metodu, kde koncový uživatel potřebuje pouze internetové připojení a běžný internetový prohlížeč. Svou variabilitou je tato metoda velice silná ovšem samotná implementace je náročná. Uživatelské prostředí a návrh komunikace mezi serverem a koncovým uživatelem je nedílnou součástí implementace a vyžaduje tak znalost některého ze skriptovacích jazyků. Na druhou stranu nevyžaduje žádnou interaktivní podporu ze strany Maple nebo další doplňující software.



Obr.1: Interaktivní aplikace *Mocninné řady s Maple* využívá serverové instalace Maple

Mladším a poněkud vyspělejším dojmem působí systém MapleNet. Jedná se komerční systém sloužící pro prezentování a publikování práce v Maple na internetu. V porovnání s předchozí metodou klade prakticky stejné požadavky na koncového uživatele. Značný náskok ovšem získává v části implementační. Díky svému návrhu

několikanásobně snižuje složitost implementace a tím i nároky kladené na tvůrce interaktivní prezentace. Systém MapleNet podporuje čtyři prezentační formy. Na serveru lze provozovat aplikace typu Java Applet, Maplet, dynamicky generované JavaServer Pages (JSP) a konečně technické dokumenty vytvořené programem Maple – *Rich technical document*.



Obr.2: Maplet *Power series* a technický dokument řešený v Maple, oba publikované systémem MapleNet

4 Motivace pro začlenění Maple do výuky

S nástupem nového uživatelského prostředí Maple přichází i nová forma dokumentu. Technický dokument neboli *Rich technical document* s sebou přináší mnohé vylepšení a inovace, jeho možnosti jsou v porovnání s klasickým zápisníkem bohatší. Přímé vkádání interaktivních prvků, grafických výstupů nebo výpočty pomocí kontextové nabídky jsou důkazem jeho široké variability. Obrovskou předností je vlídnost k novým uživatelům. Jednoznačně lze tvrdit, že tento směr umožní přímé zapojení Maple do výuky. Důkazem tomu je kurz vyučovaný na Přírodovědecké fakultě Masarykovy univerzity s názvem Matematická analýza s Maple pro obor Matematická biologie. Tento kurz je plně vyučován v Maple a paralelně doplňuje probíranou látku matematické analýzy v prvním ročníku.

Technologie klikacího kalkulu s sebou přináší značné zjednodušení a zpřehlednění práce. Mimo to je součástí instalace množství nástrojů, průvodců a asistentů. K dispozici jsou vzorové šablony základních matematických problémů.

Závěr této kapitoly věnujeme podpoře programu Maple. Na oficiálnách stránkách firmy Maplesoft je k dispozici řada výukových materiálů a návodů. Uveďmě si některé z nich.

Materiály technického charakteru pro seznámení s Maple 11:

Maplesoft Documentation Center[a] – dokumentace, návody, manuály, příručky programátora.

Maplesoft Training[b] – demonstrativní materiály pro začínající uživatele ve formě videokurzů a prezentací.

Maple Quick Start Training[c] – přehled základních funkcí a dovedností systému Maple.

Using Embedded Components[d] – vstupní přehled vizuálních komponent technického dokumentu.

[a] www.maplesoft.com/documentation_center/
[b] www.maplesoft.com/support/training/

[c] www.maplesoft.com/support/training/coursecontent/Quickstart_training.pdf
[d] www.maplesoft.com/applications/app_center_view.aspx?AID=1857

Materiály zaměřené na výuku matematiky:

Vrámci akademické sekce[a] byl integrován program „Clickable Calculus“[b]. Jedná se o sbírku hotových aplikací[c] pro výuku matematické analýzy. Nechybí ani seriály záznamů on-line seminářů[d].

Sekce studijních materiálů eBooks & Study Guides[e] obsahuje placené elektronické sbírky. Sbírka *Precalculus*[f] je dělena do jedenácti kapitol a řeší víc než šedesát problémů převážně ze středoškolské matematiky. Sbírka *Calculus*[g] je zaměřena na diferenciální a integrální počet funkce jedné proměnné. Obsahuje celkem 31 kapitol rozdělených do pěti tematických celků. Obě sbírky poskytují pro učitele matamatiky dostatek materiálu a užitečné inspirace pro tvorbu výukových materiálů.

[a] www.maplesoft.com/teachers_center/
[b] www.maplesoft.com/products/Maple11/academic/clickable_calculus/
[c]
www.maplesoft.com/products/Maple11/academic/clickable_calculus/applications.aspx
[d] www.maplesoft.com/demo/recorded-seminars.aspx
[e] www.maplesoft.com/products/studyguides/
[f] www.maplesoft.com/products/studyguides/precalculus/
[g] www.maplesoft.com/products/studyguides/CalculusStudyGuide/

5 Závěr

Dnešní „asistenti“ matematiky, ať už Maple či jiné systémy pro matamatické výpočty, zaujaly v současné době důležitou roli a staly se tak našimi partnery. Ovšem i přes neustálé zdokonalování systémů počítačové algebry nelze opomíjet klasickou formu výuku matematiky. Dostatečné zvládnutí teorie a pochopení probírané látky je nezbytnou součástí pro úspěšné prohlubování znalostí a uvědomování si dalších zkušeností.

6 Odkazy

[1] Kříž, P.: www.math.muni.cz/~kriz/pseries, Mocninné řady s Maple, 2006
[2] Kříž, P.: melian.ics.muni.cz:8080/maplenet/kriz/pseries, 2006
[3] Kříž, P.: melian.ics.muni.cz:8080/maplenet/kriz/taylor.mw, 2007
[4] Maplesoft: www.maplesoft.com, 2007

Podpora aplikací matematických disciplín systémem Maple při řešení současných problematik globálního charakteru

Zuzana Chvátalová

Fakulta podnikatelská Vysokého učení technického v Brně
Ústav informatiky
Kolejní 4, 612 00 Brno
chvatalova@fbm.vutbr.cz

Abstrakt

Článek se zabývá využitím počítačového systému Maple při aplikacích kvantitativních metod ve vědních oborech a ve výuce matematiky jako významného aspektu při modelování skutečných jevů, jejich analýze a jako podpory pro řešení současných problémů globálního charakteru, rozhodování, jejich klasifikací apod. Jsou uvedeny tři jednoduché případy.

Abstract

The paper deals with using computer system Maple in applications of quantitative methods in scientific disciplines and in the teaching of mathematics as an important aspect in modeling of real phenomena, in analyses of phenomena as a support for present problem of global character solving, decision making, classification of phenomena etc. Three simple cases are presented.

Klíčová slova

Kvantitativní metody, matematické disciplíny, počítačový systém Maple, ekonometrie, modelování, výuka, problémy globálního a environmentálního charakteru.

Key words

Quantitative methods, mathematical disciplines, computer system Maple, econometrics, modeling, teaching, problems of global and environmental character.

1 Úvod

Zrychlený životní styl, často se měnící vnější i vnitřní podmínky, nezbytná informovanost a komunikaceschopnost překonávají časoprostorové bariéry ve společnosti a korespondují s šíří a rychlostí vývoje informačních a komunikačních technologií (ICT). Toho lze využít mj. jako kvalitní prostředek podporující **aplikační sílu kvantitativních metod**, tj. **metod založených na bázi výstupů vesměs matematických disciplín**, v praxi i v procesu přípravy na ni, **při řešení současných aktuálních odborných problematik globálního charakteru, k nimž neodmyslitelně patří problémy environmentální povahy a problémy s tím související.**

2 Kvantitativní metody a ICT

Potřeba užití kvantitativních metod stále častěji proniká do většiny vědních oborů, a to nejen technických, nýbrž i ekonomických a společenskovědních, a tím se stále více podílí na řešení problémů soudobých globálních společenských trendů i megatrendů. Rozumný stupeň matematizace společnosti si vynucuje potřeba kvantifikace, odůvodňování faktů v časové dynamice, předvídání vývoje událostí, umění abstrahovat pro modelování a simulování reálných jevů při analýze problémů, jejich řešení, a následně pro jednání, rozhodování i řízení. Navzdory neoptimistické tendenci v České republice, zejména po revoluci 1989, kdy matematika a matematické disciplíny v pozici samostatných oborů (v procesu vzdělávání i v praxi) jsou často vnímány jako ryze teoretické, nezáživné, obtížné, málo použitelné a v neposlední míře obtěžující, je nasnadě **posílit aplikace matematiky a matematických disciplín pro praxi**, a to již kvalifikovaným systémem vzdělávání.

Implementace prostředků ICT přispívá k zatraktivnění, zkvalitnění, k podpoře prohloubení logického pochopení sledované problematiky, v nemalé míře ke zvýšení produktivity práce, k vedení k samostatnosti při rozhodování, ale zvyšuje i možnosti odborné komunikace a výměny zkušeností, a tím k tak preferované týmové práci. Nasazení prostředků ICT by rovněž mělo zohledňovat výrazné faktory vývojové perspektivy, jejich užití z pohledu potřeb (případné variability výběru) daného oboru, vypěstování vhodných návyků pro setrvání v budoucí praxi, zejména při snaze sofistikovaného přístupu k dané problematice, jejich možné expanzi a inovacím. Stagnace v tomto ohledu či přecenění nežádoucího aspektu na úkor jiného mohou být předzvěstí neúspěchu, často s výrazně negativními důsledky, zejména v prostředí environmentálních vztahů a souvislostí v nejrůznějších oblastech společenských aktivit.

Není však nutné přeceňovat nerozumnou participaci ICT v lidských činnostech a předávat jim kompetenci v krocích, které jsou doménami počínání lidských zdrojů. Přílišné mechanizování, formalizace a automatizace úkonů mohou podnítit nežádoucím způsobem eliminaci toho, co je lidem vlastní – myšlení. Implementace a vhodná volba ICT by měly být velmi silnou stránkou i příležitostí ke zkvalitnění a zajištění patřičných podmínek pro jejich realizaci.

3 Volba systému Maple pro aplikace kvantitativních metod

Není novou informací, že produkt Maple počítačové společnosti Maplesoft výrazně proniká do mnoha komerčních, výzkumných i vzdělávacích sfér a v konkurenci dalších vynikajících počítačových systémů zaujímá celosvětově významné a všeobecně uznávané postavení (zejména s ohledem na své široké implementační možnosti v praxi). Za téměř třicet let jeho vývoje od prvotních projektů, při zkvalitňování jak pracovního, tak obecně uživatelského prostředí zdokonalováním servisních, informačních i komunikačních nástrojů, posilováním interaktivních vazeb, reakcemi na horké podněty praxe, výuky a vědy, poskytuje komplexní, dynamické a otevřené prostředí pro cílové skupiny různých uživatelů, různých stupňů odbornosti, v různých oborech kvantitativního i kvalitativního charakteru. Přičemž důležitou devizou jeho nových verzí je stále komfortnější a samostatnější uživatelská obslužnost. Důkazem toho je na mnoha fórech prezentovaná současná verze Maple 11.

4 Maple v aplikacích matematických disciplín pro praxi

V dalším stručně zmiňme tři případové situace pro využití systému Maple jako užitečné podpory při aplikacích matematických disciplín, které hrají úlohu jak při sledování kvantitativních charakteristik, tak při sledování kvalitativních vlastností jevů. A to v různých aplikačních oborech:

(A) *Při analýze atypických jevů v ekonomii.*

(B) *Při názorném zobrazení objektů ve složitých, byť bohatě aplikovatelných matematických teoriích.*

(C) *Při výuce v základním kurzu matematiky na vysoké škole.*

4.1 Případ 1

(ad A) *Při analýze atypických jevů v ekonomii – identifikace a charakteristiky atypické poptávky po jisté komoditě může zachytit jak kvantitativní znaky, tak kvalitativní znaky sledovaného jevu.*

Bylo-li zmíněno, že kvantitativní metody stále častěji pronikají i do disciplín společenskovědního charakteru, ekonomie je toho typickým představitelem. Je třeba znát nejen statistiky nejrůznějších ekonomických jevů, nýbrž zejména v tržních podmínkách je stále častěji třeba ekonomické jevy sledovat v souvislostech, modelovat a kvantifikovat je. A tak vznikají příbuzné obory zohledňující tato fakta. U nás zejména po revoluci 1989 výrazně proniká do povědomí obor zvaný ekonometrie, který ve světě (v USA) si v podmínkách tržního hospodářství vynutil samostatnou profilaci již ve 30. letech 20. století.

Ekonometrie jako kvantitativní ekonomická disciplína se zabývá především měřením v ekonomice na základě analýzy reálných statistických dat pomocí ekonometrických metod a modelů (Hušek, 1999).

Matematika a matematické disciplíny tedy mají prostor v jistém slova smyslu „modelovat a měřit“ ekonomické veličiny, ekonomické skutečnosti, sledovat tak jejich zákonitosti, pokoušet se odhalit jevy nestandardní, okrajové apod. V komplexnosti různých podmínek je potřebné nastavit rozhraní srozumitelné komunikace odborníků – nematematiků s takto vymezeným prostředím plným matematických aplikací. K tomu může sloužit pevná základna vymezená volbou vhodného počítačového systému. To vše může sehrát významnou roli podporující řešení problematik globálního charakteru související např. s logistickými řetězci, externalitami aj., a tím vstupující jako významné charakteristiky či ukazatelé pro environmentálního rozhodování.

Záměrem dále uvedeného případu je charakterizovat komodity s atypickou poptávkou (Chvátalová[2], 2005).

Poptávka je chápána jako jednoduchý model dvou endogenních proměnných, ceny P a poptávaného množství Q, tj. $Q = D(P)$, ceteris paribus. Poptávka se obecně řídí známým zákonem klesající poptávky. Atypickou poptávkou tedy rozumíme poptávku, jejíž nestandardnost je dána porušením zákona klesající poptávky na sledovaném intervalu cen.

Při předběžném výzkumu zaměřeném k identifikaci atypické poptávky dle konkrétních empiricky zjištěných diskrétních hodnot vyhodnocením vizuálního modelu byly ve spotřebním koši vytypovány dvě komodity[3] M a S (při srovnatelných obecných podmínkách), s předpokládanou vlastností. Dle druhu komodit lze (budeme) předpokládat, že jde o substituty, tj. komodity vzájemně „zastupitelné“.

---

[2] Případ je upravenou verzí úvahy v (7).

[3] Pro jednoduchost nebudeme uvažovat „fyzický“ obsah obou komodit, nýbrž je pouze označme a popišme.

Aby bylo možné atypickou poptávku vymodelovat jako funkční závislost, byl uvažován ekonomický model ve smyslu přirozeného jazyka (závislost poptávaného množství Q na ceně P), tj. opačně, než bývá historickou zvyklostí při prezentaci poptávky v ekonomické literatuře. Přitom matematický model byl požadován ve tvaru jednoduché spojité funkce, která je schopna dostatečně zachytit případný atypický průběh poptávky na sledovaném intervalu cen.

Dále byly určeny grafické modely pružností obou poptávek a analyzovány ceny, při nichž dané poptávky byly pružné, jednotkově pružné a nepružné.

Cenová pružnost poptávky (krátce pružnost) je významnou mikroekonomickou charakteristikou, která zachycuje míru změny poptávaného množství jako reakci na změnu ceny (případně jako důsledek lze dále sledovat dopad těchto změn na změnu celkového příjmu TR). Platí $E = E(P) = \left| \frac{\mathrm{P}}{\mathrm{Q}} \cdot \frac{dQ}{dP} \right|$. Pro pevně zvolenou cenu P jde o pružnost poptávky při této ceně, $E(P)$ pak vyjadřuje rychlost změny poptávaného množství při ceně P. Dle hodnoty $E(P)$ vůči číslu 1 pak lze rozhodnout, zda je poptávka pružná, nepružná či jednotkově pružná při ceně P.

Situaci analyzujme v pracovním prostředí Maple.

Modely poptávek

Pro určení obou poptávek (označme odpovídající poptávané množství QM a QS) po zmíněných komoditách M a S byla v příkazu v Maple v metodě nejmenších čtverců požadována, a tedy předdefinována, kvadratická funkce:

```
>eq_fit:=fit[leastsquare[[P,Q],Q=a*P^2+b*P+c,{a,b,c}]]([Pv
 alues,Qvalues]);
```

a tím byly určeny kvantitativní modely obou poptávek:

získaný výstup:

$$QM = -9{,}66017\,P^2 + 97{,}7844\,P + 859{,}524,$$

$$QS = 0{,}217571\,P^2 - 60{,}3392\,P + 1560{,}03.$$

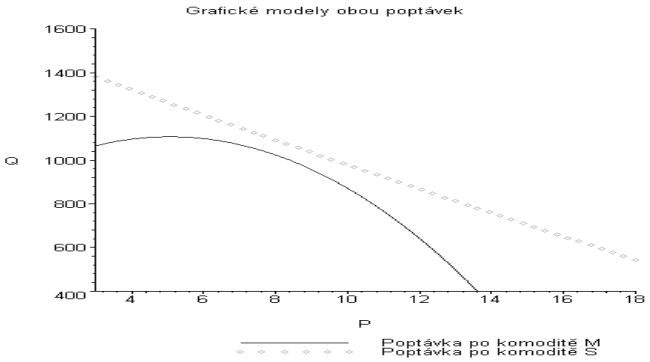
Dále byly výstupy definovány jako procedury a následně vykresleny přímo grafické modely těchto poptávek. Oba grafy pro možnost snazšího vizuálního vyhodnocení a porovnání jsou umístěny do téže soustavy souřadnic:

```
> QM:=proc(P) -9.66017*P^2+97.7844*P+859.524 end;

> QS:=proc(P) 0.217571*P^2-60.3392*P+1560.03 end;

> plot({QM(P),QS(P)}, P=3..18, Q=400..1600, style=[line,
 point], title="Grafické modely obou poptávek");
```

Získaný výstup:



Obr. 1: Grafické modely poptávek

Zhodnocení

Je vidět (viz Obr. 1), že poptávka po komoditě M má atypický průběh – nejde o klesající funkci na uvedeném intervalu cen, zatímco poptávka po komoditě S se chová standardně. Parabola, která charakterizuje poptávku po komoditě M je otočená „dolů“, zatímco parabola vyjadřující poptávku po komoditě S je otočená „nahoru“, je zachycena částí pouze jedné její větve, dle proporcí vizuálním vyhodnocením křivky by bylo možné tento model vyjádřit i jako model lineární.

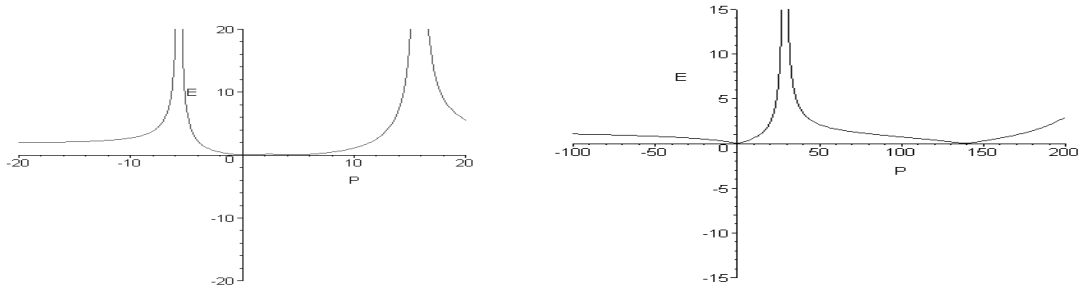
Cenové pružnosti poptávek – grafické modely

Cenové pružnosti poptávek není třeba v Maple předpočítávat, stačí aplikovat naznačené operace pro jejich určení ze zadaných poptávek (obě poptávky jsou příhodně vyjádřeny jako funkce proměnné *P*):

```
>E:=proc(P)abs(P*diff(Q(P),P)/Q(P)) end;
> plot(E(P),P=0..20,E=0..20);
> plot(E(P),P=0..15,E=0..200);
```

Získaný výstup:

Grafický model cenové pružnosti poptávky po komoditě M (vlevo) a S (vpravo):



Obr. 2: Grafické modely pružností poptávek

Analýza pružností poptávek

Analýza pružností poptávek v Maple bude provedena sadou příkazů řešících jednu rovnici a dvě nerovnice vzhledem k ceně *P*:

```
>JEDNOTKOVĚ PRUŽNÁ:=solve(E(P)=1,P);
>PRUŽNÁ:=solve(E(P)>1,P);
>NEPRUŽNÁ:=solve({E(P)<1,P>0,P<960.8560187},P);
```

Získaný výstup:

- Komodita M

  Při ceně 9,780666429 je poptávka jednotkově pružná, $E(P) = 1$.

  V intervalu cen (9,780666429;15,76597936) je poptávka pružná, $E(P) > 1$.

  V intervalu cen (0;9,780666429) je poptávka nepružná, $E(P) < 1$.

- Komodita S

  Při ceně 13,98500324 je poptávka jednotkově pružná, $E(P) = 1$.

  V intervalu cen (13,98500324;28,85697555) je poptávka pružná, $E(P) > 1$.

  V intervalu cen (0;13,98500324) je poptávka nepružná, $E(P) < 1$.

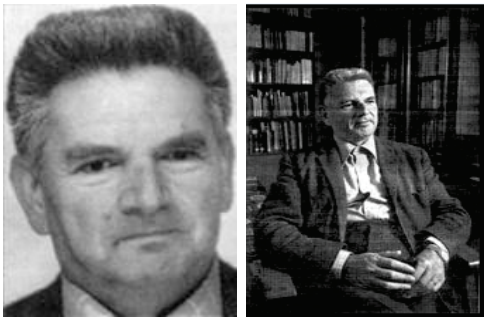
Vyhodnocení

- Před modelováním poptávky jsme předpokládali o komoditách M a S, že jde o substituty. Vzhledem k zásadní odlišnosti charakteru jejich poptávek - v případě komodity M jde o výrazně atypickou poptávku (od cenové hladiny cca 6 níže – pro spotřebitele pravděpodobně signalizující od této ceny podezření na špatnou kvalitu), v případě komodity S jde o typickou poptávku (jakoby tato značka spotřebiteli garantovala kvalitu i při nízkých cenách, a proto se ji rozhodne kupovat i za nejnižší ceny na rozdíl od komodity M) - předpoklad o substitutech je tedy diskutabilní.
- Je vidět, že grafické modely poptávek i jejich pružností velmi usnadní orientaci v celkové situaci a **dokonce kvantitativní výstupy** mohou poukázat na preference spotřebitelů, a tedy **signalizovat výstupy kvalitativního** charakteru. V Maple jsou tyto výstupy přehledné, jednoduše a pochopitelně konstruovatelné a modifikovatelné.
- Podotkněme, že **Maple matematicky korektně vyřeší** výše uvedené rovnice i nerovnice, tak, že nabídne další hodnoty či intervaly cen. Proč je nezahrneme do následných výsledných úvah? Je to velmi důležitý okamžik. Protože tehdy přistupuje **lidský faktor** a nezbytná **ekonomická interpretace získaných výsledků**, je nutno vybrat pouze **ekonomicky smysluplné hodnoty**. (Někdy lze tyto skutečnosti v Maple nastavit předem přímo do příkazů. Ale tím se syntéze matematiky a ekonomie nevyhneme, jen ji přesuneme. Je otázka, kdy je to snazší.) Tedy ekonomická rozvaha rovněž musí participovat při interpretaci problému i výsledků při využití kvantitativních metod a počítačového systému.

4.2 Případ 2

(ad B) *Při názorném zobrazení objektů v teorii katastrof – v této části bude jen velmi volně a okrajově (spíše pro zajímavost) podtržena výhoda pracovního prostředí a možností sytému Maple při aplikaci náročné matematické disciplíny ve společenskovědních oborech (což by zřejmě přivítal zakladatel teorie katastrof ve své době).*

Jak již bylo v úvodu řečeno, v řadě vědních oborů je stále více třeba užívat matematických metod. Lze říci, že právě biologie byla faktickou základnou pro vznik velmi zajímavé a náročné matematické teorie, **teorie katastrof**, kterou založil v 60. letech 20. století francouzský matematik René Thom. Volně řečeno termínem „katastrofa“ Thom označuje (v určité korespondenci s přirozeným jazykem) náhlou a neočekávanou kvalitativní změnu stavu systému při malých spojitých změnách parametrů, na nichž systém závisí.



Obr. 3: René Thom[4] (1923-2002)

Ve své slavné větě Thom dokázal, že za jistých dostatečně obecných podmínek existuje pouze konečný počet elementárních katastrof, které lze popsat dostupným matematickým aparátem. Výrazným pozitivem teorie katastrof je možnost názorné interpretace, která je zajištěna geometrickým přístupem k problematice (Votroubková-Chvátalová, 1983). Přitom aparát k tomu je dosti složitý, využívá netriviálních výsledků různých matematických oborů. Systém Maple je výrazným pomocníkem pro grafické výstupy prezentovaných výsledků v této teorii a jejich nejrůznější modifikace, polohování, analýzy dalšími matematickými prostředky apod. Rovněž vzhledem k souvisejícím výstupům vztahujícím k identifikaci tzv. rovnovážných stavů systémů, nacházející oporu v teorii diferenciálních rovnic, Maple nabízí vhodné podmínky pro modelování jevů této zajímavé teorie.

Thomova teorie skýtá možnost aplikací v mnoha oborech (např. biologie, psychologie, medicína, ekonomie, sociologie aj.). Například autorem známých aplikací je anglický matematik E. C. Zeeman, který poznatky této teorie kloubil s výsledky pokusů se stresem a agresivitou zvířat.

René Thom zemřel v roce 2002 a do konce svého života se věnoval svým výzkumům. V současnosti se teorie katastrof těší oblibě na řadě odborných pracovišť nejen ve světě, nýbrž i u nás.

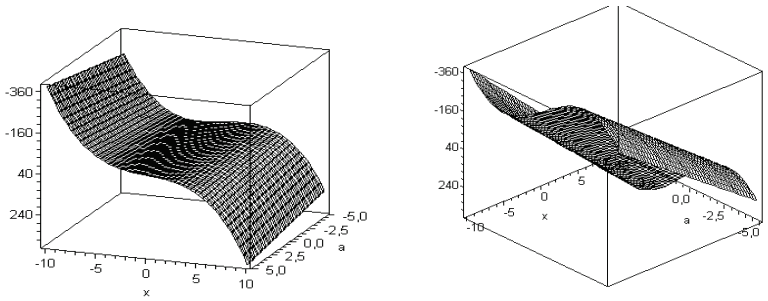
Cílem této části příspěvku není kategorizace katastrof nebo systematický výklad určité problematiky v tomto oboru, jde o poukázání na smysluplnou aplikaci kvantitativních metod v konfrontaci s **komfortem systému Maple pro služby významné teorii** a snad zamyšlení se nad časově-historickými souvislosti a možnostmi doby v kontextu úžasných poznatků vynikajících vědců.

Jako ukázku uveďme v Maple 3D grafickou interpretaci alespoň jedné ze sedmi katastrof „Fold“ („Záhyb“) s jednou stavovou veličinou x a jedním řídícím parametrem a. Příkazem a snadnou manipulací v Maple vytvořme grafický model této katastrofy ve dvou polohách:

[4] Zdroj: http://www.ihes.fr/EVENEMENT/Thom/Thom1.html

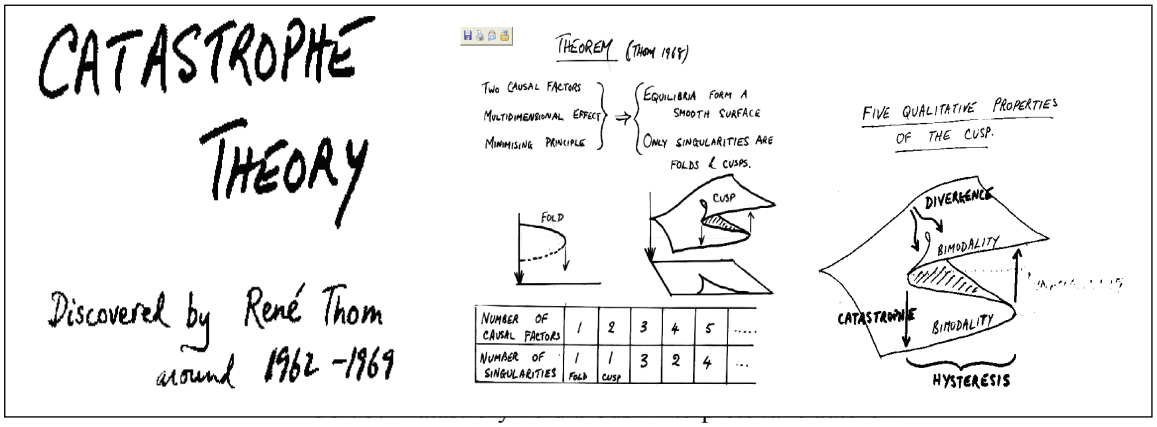
```
plot3d((1/3) * x^3 + a * x, x = -10..10, a = -5..5, grid = [50, 50], axes
    = boxed, color = "SkyBlue");
```

Získaný výstup:



Obr. 4: Katastrofa Fold v Maple ve dvou pohledech

Pro zajímavost srovnejme s interpretacemi katastrof samotným autorem v 60. letech 20. století[5]:



[5] Zdroj: http://zakuski.utsa.edu/~gokhman/ecz/c.html; Catastrophe theory *by E.C. Zeeman* Trinity University, San Antonio, March, 1995

4.3 Případ 3

(ad C) *V základním kurzu matematiky se stále častěji jeví potřeba inovovat výukový proces, ve smyslu inovací matematické výchovy jako celkové filozofie a možnostmi její implementace do následné praxe. Nutným komplementem tohoto přístupu je racionální nasazení vhodných prostředků ICT.*

Důraz při výuce matematických disciplín je kladen zejména na:

- nepřetěžování teoretickým aparátem,
- aplikační možnosti matematiky pro odborné disciplíny,
- osvojení si obecně platných postupů,
- zohlednění skutečnosti, že studentská klientela nemá stejnorodou „matematickou základnu“, jak z hlediska hloubky a šíře znalostí, tak z hlediska vztahu k matematice.

Současné společenské trendy předurčují nové požadavky formalizace i obsahovosti výuky, a to především při zohlednění širokého uživatelského nasazování a dostupnosti prostředků ICT. Je třeba dbát mnohem více na porozumění principům a smluveným či platným pravidlům, vývojové logice, ošetření kolizí, nastavení smysluplných podmínek, zařazováním jednodušších variant příkladů, nežli žádat nepřehledné a časově náročné ruční výpočty rozsáhlých matematických struktur s hrozbami triviálních numerických chyb, a tím úniku podstaty řešeného problému.

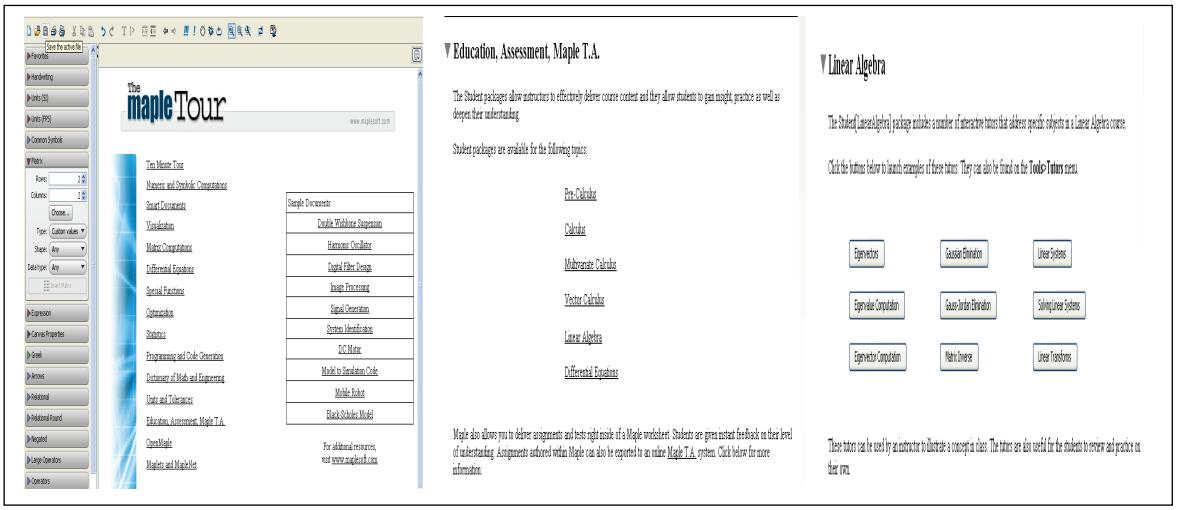
Systém Maple poskytuje příznivé podmínky pro komplexní výuku ze strany studenta, učitele i tutora, a to ve fázi:

- přípravné,
- prezentační,
- kontrolní a vyhodnocovací.

Zaměřme se však na detail (detail z pohledu celkové nabídky systému) představující komfort pracovního prostředí současné verze Maple 11 při osvojování učiva podporující samostatnost a variabilitu rozhodování při výpočtu.

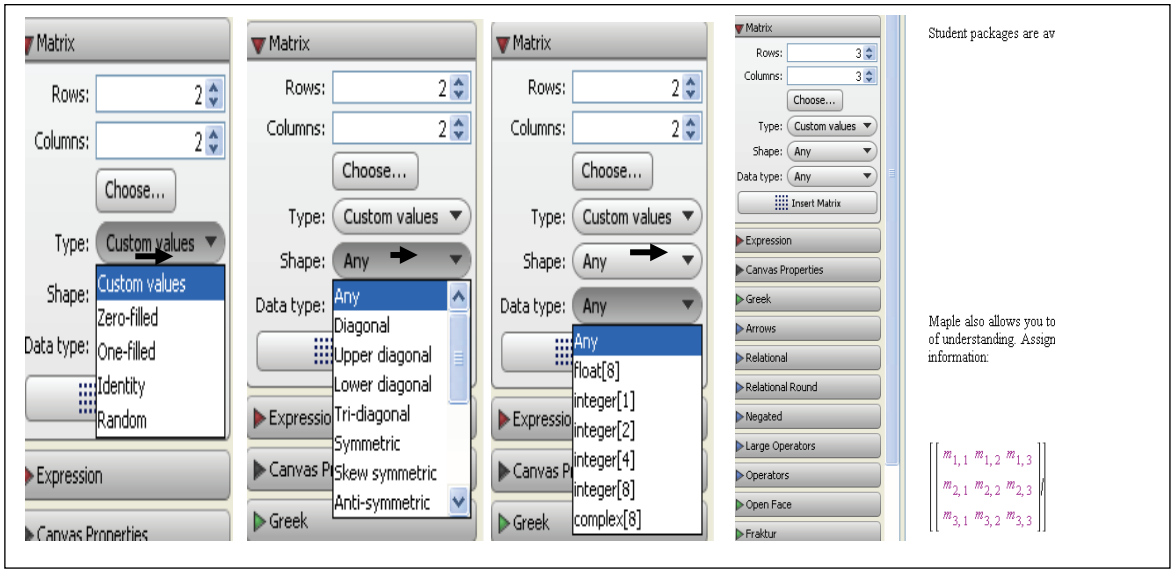
Mnoho problémů z praxe vede k sestavení soustav lineárních rovnic, v souvislosti s tím pak při hledání jejích řešení např. k potřebě určení inverzní matice k matici soustavy. K tomu existuje několik metod. Pro ruční výpočty se doporučuje nejčastěji metoda Jordanova (volně řečeno, jde o „připojení“ jednotkové matice zprava k matici soustavy a pomocí tzv. řádkových úprav její převedení „doleva“). Princip této metody je jednoduchý, avšak početně i pro nevelké matice činí časté numerické problémy. Ve výuce to vede k zbytečnému strachu, protože podstatě metody studenti dobře rozumí, a přesto se jim někdy správný výsledek získat nedaří.

Prezentujme, jak je vybaven systém Maple pro výpomoc při řešení takového problému. Bude užit asistent v sekci Education, Assessment, Maple T.A., v podsekci Linear Algebra, v odstavci Matrix Inverse:



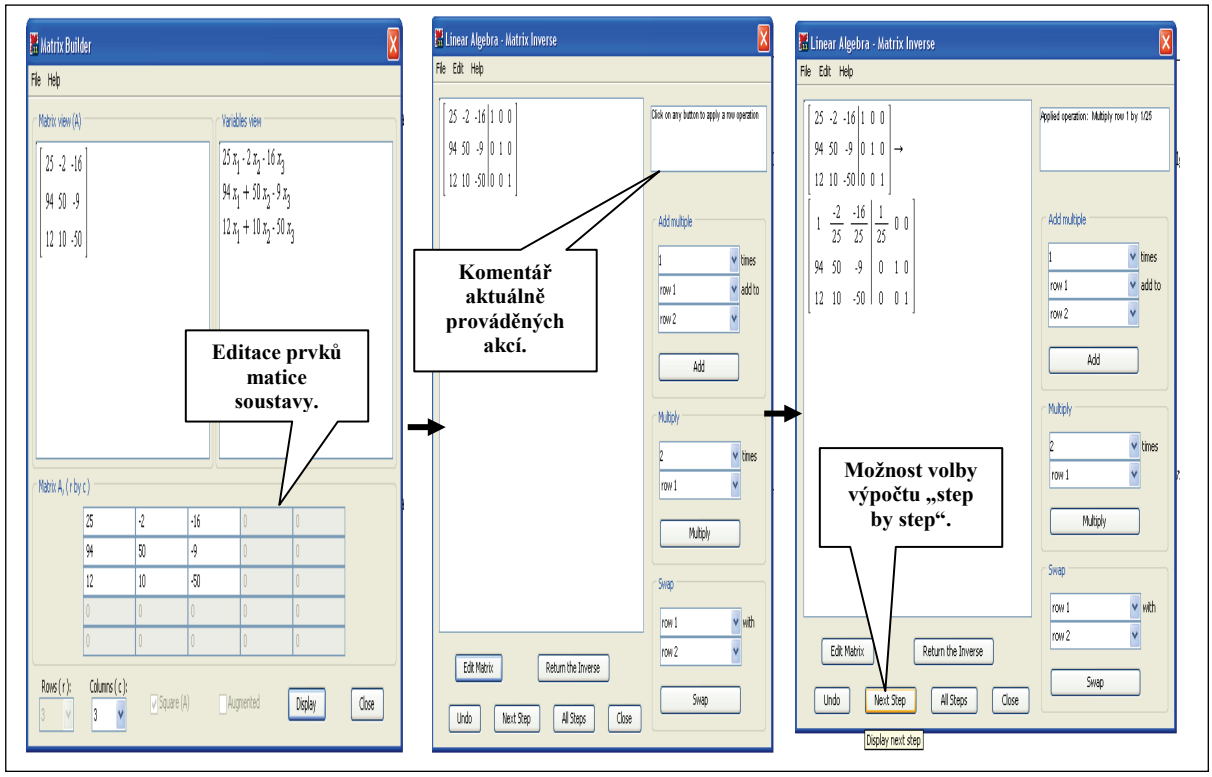
Obr. 6: Výběr Maple-asistent k výpočtu inverzní matice

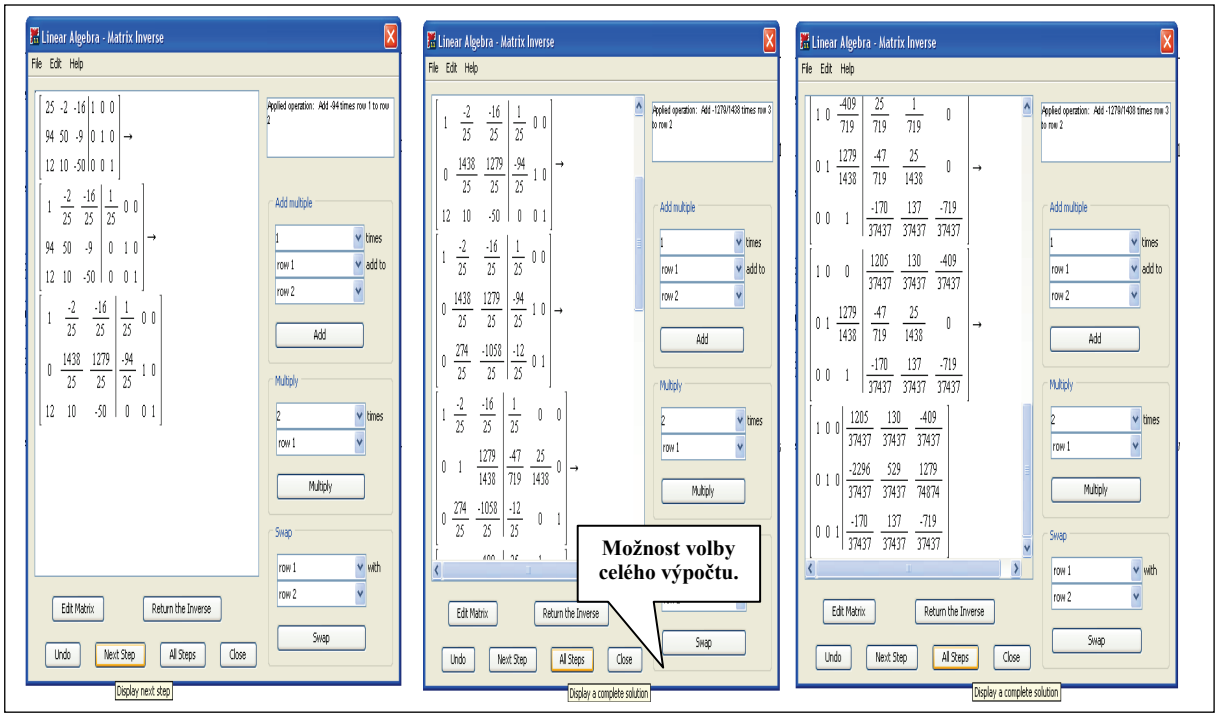
K pohodlí systému přispívají rychle dostupné palety nástrojů pro editaci:



Obr. 7: Palety nástrojů k editaci matic

Vhodnou prezentaci průběhu výpočtu zajišťuje posloupnost interaktivních a srozumitelných asistentů (v akci):


Matrix Builder
Editace prvků matice soustavy.
Linear Algebra - Matrix Inverse
Komentář aktuálně prováděných akcí.
Možnost volby výpočtu „step by step“.


Linear Algebra - Matrix Inverse
Možnost volby celého výpočtu.

Výhodou současné verze Maple 11 je i to, že obsahuje samostatný prezentační mód, takže nejen studentovi takto vedená výuka přináší zvýšení produktivity práce, možnost hlubšího pochopení vlastním ovládáním výpočtu, ale i učitel má možnost pohodlně vysvětlovat problematiku v živě řízené přednášce nebo ji dokumentovat v elektronické či fyzické formě.

5 Závěr

- Řešení problematik determinovaných globálními společenskými trendy (nezbytně otázky environmentálního charakteru) vyžadují implementace nástrojů ICT do nejrůznějších oblastí života každého člověka. Jejich vhodná volba v praxi, v nejrůznějších oborech, v systému vzdělání, ve vědě i ve výzkumu tak podpoří využívání kvantitativních metod, a tedy širokých možností aplikací matematiky a matematických disciplín. A tím lze stále více odborně, řešit problémy, avšak i hledat společné rozhraní či propojovat společenské oblasti kvalitativního a kvantitativního charakteru, sofistikovaně a produktivně řešit jejich problémy.
- Rovněž požadavky managerů na pracovní tým mají v současnosti jiný obsah než dříve. Je kladen důraz na skloubení vědomostí a schopností do dovedností učinit rozhodnutí, často interdisciplinárního charakteru, pod vlivem času, závisející na korektní informovanosti apod. To si bez využívání nástrojů ICT nelze představit.
- Systém Maple nabízí kompletní ICT k zlepšení výuky matematických disciplín a jejich aplikaci v praxi. Umožňuje řešit problémy mnoha (i netradičními) formami výuky (e-learning aj.). Uživatelské, pracovní a komunikační prostředí systému Maple jednoznačně zkvalitňuje výuku matematických disciplín, podporuje hlubší a snazší pochopení probíraného učiva, zohlednění jevové dynamiky v čase a v souvislostech vizualizacemi, animacemi a simulacemi. Podporuje aktivity, které propojují teoretické disciplíny s praxí.

6 Literatura

[1] Hřebíček J. and Žák V. *Nové možnosti systému Maple ve výuce*. *In:* Sborník čtvrtého ročníku konference o e-learningu SCO 2007, Brno, 2007.

[2] Hušek R. *Ekonometrická analýza.* Praha: EKOPRESS, s.r.o., 1999.

[3] Chvátalová Z. *Teaching of Econometrics with Support System Maple. In:* Proceedings: Conference ApliMat 2007. Bratislava (SR), 2007.

[4] Chvátalová Z. *Možnosti užití Maple v matematických disciplínách v ekonomických studijních programech. In: Proceedings:* Festive Scientific International Conference 2007, FP VUT v Brně, Brno 2007.

[5] Chvátalová Z. and Kříž J. *Příprava výuky ekonometrie s využitím systému Maple.In:* Sborník 5. matematického workshopu s mezinárodní účastí, FAST VUT v Brně, Brno 2006.

[6] Chvátalová Z., Kříž J. and Ondrák V. *Výuka matematických disciplín a ekonometrie užitím Maple. In:* Sborník 6. matematického workshopu s mezinárodní účastí, FAST VUT v Brně, Brno 2006.

[7] Chvátalová Z. *Atypické chování poptávky vybraných komodit s podporou metod matematického modelování.* Dizertační práce. FP VUT v Brně. Brno 2005.

[8] Votroubková-Chvátalová Z. *Elementární teorie katastrof.* Diplomová práce. MU v Brně. Brno 1983.

[9] Varian H. R. *Mikroekonomie*. Přeložil Ing. Libor Grepa. Praha: Victoria Publishing, 1995.

[10] http://www.vladimirzak.com/sco2007.zip

[11]http://www.fi.muni.cz/~hrebicek/maple/cas/

[12]http://www.maplesoft.cz

[13]http://www.maplesoft.com

[14]http://zakuski.utsa.edu/~gokhman/ecz/c.html

[15]http://www.ihes.fr/EVENEMENT/Thom/Thom1.html

Současné metody vyhledávání informací a prezentace jejich výsledků

Matěj Štefaník

Masarykova univerzita
Botanická 68a, 602 00,Brno
stefanik@iba.muni.cz

Abstrakt

Příspěvek popisuje současný stav a technologie, které se používají při hledání informací v prostředí internetu. Dále uvádí nové přístupy, které se začínají v problematice vyhledávání uplatňovat. Součástí je rovněž popis požadavků, které jsou na vyhledávání kladeny z celé řady směrů, a které je možné obecně zahrnout pod pojem personalizace vyhledávacích nástrojů. Příspěvek se pak zaměřuje na popis možných způsobů řešení zmíněných problémů a to zejména za použití moderních prostředků a nástrojů sémantického webu. Vzhledem ke skutečnosti, že vize plně sémantického webu je však stále ještě vzdálená, je navrhované řešení kombinací sémantických a klasických vyhledávacích postupů.

Abstract

This paper presents current state and technologies that are used for information retrieval in the Internet. The paper further describes new approaches that are being used in these problems and also describes requirements that may affect information retrieval. These requirements may flow from variety of stimuli but in general we can use for overall description a term "personalization" of searching tools and methods. The paper focuses on the description of possible means of solving mentioned problems and especially using modern tools of semantic web. With regard to a situation that the vision of fully implemented semantic web is still far away the proposed solution is a combination of semantic and classical searching methods.

Klíčová slova

Vyhledávání, hledání informací, ontologie, kategorizace, personalizace, RDF;

Keywords

Searching, Information retrieval, Ontology, Clustering, knowledge, personalization, RDF;

1 Úvod

Informační komunikační technologie (ICT) jako nástroje umožňující přístup k informacím se prosazují na všech úrovních lidské činnosti. Zejména pak v posledních letech, kdy s rozvojem Internetu a rozšiřováním jeho služeb dostupnost informací prudce roste. S větším objemem dostupných informací současně roste počet zájemců o tyto informace, což zpětně vyvíjí tlak a vytváří požadavky na zahrnutí nových, potenciálně zajímavých informací, které doposud dostupné nebyly.

S příchodem nových technologií souvisí nové problémy, které nemusely být v počátečních fázích implementace patrné. Jedním z těchto problémů je přístup ke kvalitním, ale především relevantním datům a informacím.

V oboru vyhledávání informací se velmi často setkáváme s problémem, kdy podmínkám dotazu při hledání informací vyhovuje velké množství dokumentů (či jiných informačních zdrojů) a tyto výsledky jsou navíc často nevhodně prezentovány uživatelům. To v konečném důsledku znepřehledňuje a ztěžuje vlastní identifikaci hledaných informací. Nepřehledný výstup je způsoben často příliš vágními a nepřesnými definicemi použitými uživateli při vlastním vyhledávání, ale především pak technickou stránkou nástrojů, které se k vyhledávání používají.

Požadavky na vyhledávání vyplývají obecně z heterogenity webového prostředí, informačního přetížení a specifických informačních potřeb jednotlivých uživatelů, což jsou všechno problémy, které mohou v důsledku komplikovat efektivní vyhledávání relevantních informací. Velkou výzvou pro současné technologie tedy je, přinést svým uživatelům možnosti, jak jednoduchým a přehledným způsobem přistupovat k informacím dostupným elektronickými prostředky. Aktuálním problémem již není samotný nedostatek informací, ale spíše jednoduchost vyhledávacího procesu, vhodná vizualizace nalezených výsledků či jejich srozumitelná kategorizace. Z pohledu budoucího vývoje se dá očekávat, že požadavky na relevantnost výstupů vyhledávacího procesu porostou.

Vedle metod pro vyhledávání se v současnosti dostávají do popředí také požadavky na zvýraznění individuální role jednotlivců či skupin a jejich preferencí. Tato skutečnost právě souvisí s umožněním individuálních přístupů k vyhledávacím nástrojům, ale také k možnostem pro vizualizaci výsledků vyhledávání. Zde by mělo být uživatelům umožněno definovat strukturu a tvar výstupních dat. Právě touto problematikou se tento článek bude zabývat

2 Základní koncepty

Aktuální druhy vyhledávání, které je možné v současnosti využít (byť v některých případech stále v minimálním množství) jsou následující:

- **Syntaktické vyhledávání** – klasické vyhledávání založené na prohledávání indexů, které srovnává slova pouze na základě jejich syntaxe.
- **Sémantické vyhledávání** – vyhledávání, které se snaží vyhledávat i na základě významu a kontextu hledané informace a tím přispívá ke zvýšení relevance výsledků.
- **Hybridní vyhledávání** – využívá kombinaci obou výše popsaných. Dotazem není množina klíčových slov, ale množina konceptů. Každý koncept je pak převeden na množinu klíčových slov (někdy může obsahovat rovněž váhu), která představuje vstup pro syntaktický vyhledávač.

V současnosti je právě problematika zanesení individuálních prvků do procesu vyhledávání okrajovým tématem, ačkoli se ukazuje, že právě *personalizace* (zohlednění uživatelských

požadavků) je jedním z klíčových atributů budoucích vyhledávacích nástrojů (a nejenom vyhledávacích).

Současné implementace umožňují uživateli minimální flexibilitu a dovolují mu modifikovat pouze omezené množství atributů. Nejčastějším atributem je pak omezení velikosti prohledávané domény.

Dalšími významnými prvky, který spadají do problematiky vyhledávání, jsou možnosti zprostředkování výstupů vyhledávání uživateli. Potenciálně je použitelná široká škála přístupů. První možností je jednoduchý *seznam* objektů bez jakékoliv vnitřní struktury (vhodný při hledání v malých doménách), následně je možné využít různých *indexačních metod* (ranking), které mohou vyjadřovat např. úroveň relevance dokumentu. Samozřejmě existují další způsoby jak data prezentovat, významnou skupinu pak tvoří metody pro *kategorizaci* výstupu do skupin. Tyto metody mají potenciál zpřehlednit výsledky uživateli a poskytnout mu celkový přehled o obdržených výsledcích.

3 Současný stav

S uvedenými problémy úzce souvisí nové možnosti a trendy ve vyhledávání, které by měli být rovněž reflektovány. V současnosti se začínají rozvíjet a uplatňovat přístupy které podstatným způsobem obohacují současné vyhledávací nástroje a metody a přináší tak nové možnosti. Mezi tyto nové přístupy můžeme zařadit např:

- Metasearch vyhledávání – je princip vyhledávání založený na agregaci výsledků z několika paralelních vyhledávacích nástrojů. Na tyto nástroje nejsou kladeny žádné dodatečné požadavky nebo podmínky (může se jednat o zcela nezávislé vyhledávací nástroje).
- Linear search – představuje zcela opačný přístup než metasearch. Linear serach použijeme v případech, kdy je zájmová doména poměrně malá a kdy je možné zohlednit rozdílné požadavky na vyhledávání pouze vymezení prohledávané oblasti. Linear search tedy představuje zúžení vyhledávacího prostoru. Kvalitnějších výsledků se v tomto případě dosahuje prohledáváním pouze vybraných datových zdrojů. Předpokladem je, že tyto datové zdroje jsou pro konkrétní hledání relevantnější a můžeme je tak označit jako důvěryhodnější. Velmi často je Lienar search tvořen seznamem těchto důvěryhodných datových zdrojů.
- Shlukování (klastrování) – Dalším častým rozšířením vyhledávání je následné zpracování výsledků a kategorizace jednotlivých odkazů (nebo dokumentů) do skupin. Tato kategorizace do skupin by měla být pro uživatele vodítkem a měla by mu získané výsledky podstatně zpřehlednit a usnadnit následné hledání relevantních informací.
- Vizualizace (Visual Search) – Nástroje pro seskupování a následnou prezentaci vyhledaných obsahově souvisejících dokumentů.

- Inkorporace ontologií a řízených slovníků – zapojení ontologií, oborových taxonomií a řízených slovníků do procesu vyhledávání.

Všechny zmíněné přístupy, a zvláště pak poslední dva, se již dotýkají poměrně bouřlivě se rozvíjející části výzkumu ICT, který můžeme obecně pojmenovat jako sémantické technologie či web 2.0. Pojem web 2.0 je poměrně široký a zahrnuje celou řadu metod a přístupů. Z našeho pohledu se k vyhledávání informací nejvíce vztahují následné prvky:

- Personalizace – se dotýká problému, jak zajistit individuální funkcionalitu či přístup k datům na základě požadavků jednotlivých uživatelů. Tato problematika se může dotýkat jak vizuální stránky, tak také získávání informací, prezentovaní vlastních informací atd.
- Long tail – s personalizací je rovněž úzce spojen problém Long tail. Tento termín popisuje proces, kdy s narůstajícím množstvím informací vzrůstá i počet zájemců o informace, které netvoří hlavní část zájmu uživatelů. Jinak řečeno, se vzrůstajícím počtem informací a služeb vzrůstá zájem o okrajová témata. Problém při vyhledávání tohoto typu informací samozřejmě souvisí s jejich velmi úzkou vazbou, což vnáší do výsledků vyhledávání značný šum a nadbytečnost. (rozdíl mezi tématy bývá pouze velmi malý)
- Reputační systémy – s heterogenitou dat a uživatelů rovněž souvisí garance kvality nalezených informací a v moderních systémech a architekturách je potřeba se na tuto problematiku soustředit. Řešením pak bývá implementace reputačních systémů, které umožňují kvalitu informací zachytit a doručit uživatelům.
- Mashups – dalším důležitým prvkem je pak umožnit uživatelům využívat dostupné softwarové komponenty volně a bez omezení. Ideou je zveřejňovat API tak, aby bylo co nejjednodušší tyto moduly libovolně (co nejvíce) využívat či je libovolně kombinovat a vytvářet tak zcela nové aplikace nebo funkcionalitu.

4 Hybridní metody

Potenciální implementací či rozšířením uvedených technik je navržení architektury, která bude používat znalost z konkrétní domény (formalizované vhodným způsobem) a bude následně na základě této znalosti optimalizovat vyhledávání informací.

Důvody pro takovou modifikaci již byly zmíněny dříve a jedná se především o nedostatečné zapojení uživatelů do vyhledávacího procesu. V současnosti jsou jejich požadavky realizovány především seznamem prohledávaných zdrojů, které mají být přednostně analyzovány (viz. Linear Search). Z tohoto důvodu by bylo vhodné rozšířit tento způsob a zachytit a následně reprezentovat znalosti o objektech a jejich relacích na základě představy uživatele. Vlastní vyhledávací proces by byl následně řízen touto zaznamenanou znalostí (tedy na základě požadavků uživatele).

Nejobecnější koncepce rozšíření je následující: Uživatel by zadal libovolný dotaz (ať už založen pouze na textové podobě s následnou konverzí do ontologie nebo spíše s asistencí založenou např. na generování ontologie pomocí uživatelské aplikace). Tento zpracovaný dotaz by pak sloužil jako centrální řídící jednotka vyhledávacího procesu a rovněž by představoval i předlohu pro kategorizaci výsledků vyhledávání.

Existující implementace využívají předdefinovaných množin konceptů, které jsou založeny na ontologiích. Uživatel pak přistupuje k těmto uloženým konceptům např. pomocí webového prohlížeče a zvolí si takové, které podle něj nejvíce odpovídají jeho požadavkům. Po výběru dostatečného množství konceptů se spustí vlastní vyhledávání. Vyhledávací mechanismus převede uživatelem vybrané koncepty na množinu klíčových slov (jednotlivé koncepty nemusí mít disjunktní klíčová slova). Tato množina je následně předána klasickému vyhledávacímu stroji, který provede standardní syntaktické prohledání webového obsahu. Z publikovaných výsledků je patrné [1], že se tento přístup ukázal jako efektivní a použitelný.

Výhoda popsaného přístupu je ve skutečnosti, že znalost o datech vyhovujících požadavkům uživatele je popsána ontologií. Po uživatel tak není vyžadována znalost konrétní domény, aby byl schopen zadávat skutečně smysluplné dotazy (např. legislativa atd.). Tento fakt vede k podstatně lepším výsledkům při hledání než obecný prohledávací algoritmus, který hledá pouze na základě uživatelem vložených klíčových slov. Použití ontologie jako šablony slouží ke zjemnění vyhledávacích požadavků a působí jako filtr, který předchází problémům se zahlcením uživatele nerelevantními informacemi, které by klasický vyhledávací proces mohl produkovat. V případě použití konceptů nemusí dotyčný uživatel vědět o doméně příliš velké detaily, ale stále je poměrně efektivně schopen provádět vyhledávání, aniž by utrpěl ztrátu důležité informace jenom z důvodu neznalosti dané terminologie či použití nevhodné kombinace klíčových slov (klíčová slova se z konceptů generují automaticky a zahrnují např. také synonyma).

V jednoduchosti je tady možné říci, že zmiňovaný přístup pomocí konceptů reprezentovaných v ontologii pouze vytváří rozhraní mezi uživatelem a vlastním vyhledávacím strojem. A v případech kdy uživatel nezná doménu příliš podrobně je tento přístup schopen usnadnit vyhledávání tím, že za uživatele doplní vhodná klíčová slova, která s daným konceptem souvisí a která uživateli nemusela být známá.

Detailnější proces vyhledávání se u zmíněného přístupu skládá z několika kroků:

- Nejprve jsou vybrány koncepty a na jejich základě je definována doména a vytvořena celková ontologie výsledného konceptu.
- Následně se pomocí transformačních pravidel vygeneruje množina klíčových slov, která reprezentuje daný dotaz
- Tato množina je předána vyhledávacímu stroji (např. Google). (nebo v případě použití přístupu Metasearch do několika paralelních vyhledávacích strojů)
- Agregace obdržených výsledků a jejich prezentace uživateli

S tímto přístupem pak souvisí celá řada problémů

- Jak zaručit, že daný koncept odpovídá požadavkům uživatele
- Jak zaručit že v případě změny domény se provede relevantní změna i v pravidlech pro generování množiny klíčových slov
- Jak dále kategorizovat příchozí výsledky vyhledávání. (muselo by být provedeno jejich další zhodnocení a klastrování, provedení stejné práce dvakrát)
- Jak umožnit uživatelům hledání i v případech, kdy není dostupný předpřipravený koncept.
- Jak umožnit uživatelů přidávat váhu jednotlivým konceptům

Dalším nedostatkem výše popsaného procesu je fakt, že se v průběhu hledání ztratí souvislost mezi původní ontologií a výsledkem vyhledávání. Vzhledem k potřebě výsledky vyhledávání nějakým způsobem dále kategorizovat a neztrácet tedy kontakt mezi dotazem a jeho výsledky je tento přístup nedostačující a vyžaduje rozšíření. Vize je, nerozdělovat vlastní proces vyhledávání od procesu kategorizace a spojit je do jednoho kroku tak, aby vyhledávací stroj vyhledával a následně rovněž kategorizoval na základě stejné předložené ontologie.

5 Rozšířený model

Jak již plyne z výše zmíněných nedostatků, je potřeba vyhnout se statickému řešení pomocí předefinovaných konceptů. Ty mohou být součástí návrhu, ale neměly by se stát klíčovým bodem. Důraz by tedy měl být kladen na personalizaci a podpoření uživatelského přístupu k prohledávacímu stroji a zároveň vyřešit problémy spojené s kategorizací, což se v současnosti ukazuje jak klíčový prvek.

Obecně by bylo možné použít při prohledávání dva zdroje znalostí:

- Znalosti uložené v daném zdroji (vytvořené autorem zdroje) – centralizované řešení, v kterém by zdroj nabízel koncepty uživatelům (potenciálně lepší znalost domény, jednoznačně lepší znalost obsahu)
- Znalost dodaná do vyhledávání od uživatele – dynamické řešení (lepší kontrola nad vyhledáváním "vím co hledám", jasná definice relací mezi objekty v doméně na základě uživatelovi vize)
- Jejich kombinace – kombinované řešení

Oproti předchozímu přístupu pak bude vyhledávání rozšířeno a bude se skládat z následujících kroků:

- Definici hledací ontologie – bude probíhat na základě tří výše zmíněných přístupů.
- Následně bude hledací ontologie předložena vyhledávacímu stroji

- Zpracování ontologie vyhledávacím strojem –
    1. vyhodnocování bude probíhat přímo na úrovni ontologie, kdy váha jednotlivých slov v ontologii bude dána metrikou (např. vzdálenost podkonceptu od kořene).
    2. Vlastní prohledávání pak bude probíhat za pomoci klasických textových indexů. Bude se procházet index dokumentu a jednotlivým nalezeným slovům bude přiřazena hodnota na základě metriky definované v ontologii. Výstupem bude rozhodnutí o zařazení nebo nezařazení informačního zdroje do výsledné množiny.
    3. Paralelně s prohledáváním bude probíhat kategorizace a v případě, kdy je prohledávaný dokument zařazen do výstupu, bude rovněž na základě předložené ontologie zařazen do odpovídající kategorie.
- Prezentace obdržených výsledků

Klíčová je samozřejmě metrika pro hodnocení relevantnosti daného informačního zdroje vzhledem k zadanému dotazu. Ta musí být do procesu hodnocení zahrnuta a musí hodnotit vztahy mezi objekty ontologie a nalezenými objekty v dokumentu. Metrika může být komplexnější a může zahrnovat např.:

- množství slov které se shodují s konceptem a které se v dokumentu objevuje
- vzájemná vzdálenost těchto konceptů v ontologii
- počet podkonceptů obsažených je v dokumentu
- atd.

Další otázkou je výpočetní složitost současných vyhledávacích a kategorizačních procesů. Vyřešení této otázky by mohla přinést MAS technologie. V tomto kontextu by řešebím bylo navržení multi-agent prostředí implementující popsaný způsob vyhledávání. V tomto prostředí by bylo možné složitost vlastního prohledávání informací (které by se u současných přístupů vyhodnocovalo vždy v jednom uzlu) distribuovat mezi celou množinu uzlů v síti. Zde by opět vzhledem k jednotnému pohledu (popsaného v ontologii) mohl velmi jednoduše prohledávání probíhat v jednotlivých uzlech a částečné výsledky by mohli být v konečné fázi shromážděny u uživatele a velmi jednoduchým způsobem agregovány

6 Závěr

Cílem práce je navrhnou takový postup a metodiku, které usnadní vyhledávání a přinese uživatelům potenciální zisky v podobě relevantnějších dat a ménšího objemu času stráveného při vyhledávání.

Základem našeho přístupu je reprezentace uživatelovi vize o objektech a jejich vztazích pomocí ontologií. Tato ontologie současně slouží jako řídící prvek vlastního vyhledávacího procesu a také jako předpis pro kategorizaci nalezených výsledků

Důležitou skutečností týkající se našeho návrhu je fakt, že předložený model by byl jednotný pro všechny vyhledávací stroje. Tímto způsobem by byla již na úrovni datové struktury zajištěna interoperabilita a vzájemná komunikace.

Další fáze vývoje se zaměří na identifikaci vhodné domény pro pilotní implementaci. Důležitými prvky vývoje budou navržení vhodných znalostních domén pro vybranou oblast společně s navržením vhodných metriky pro hodnocení relevance informačního zdroje vzhledem k předloženým požadavkům a vhodné způsoby generování struktury kategorií.

7 Literatura

[1] D. Berrueta, J. E. Labra, L. Polo; Searching over Public Administration Legal Documents Using Ontologies, CTIC Foundation, Gijón, 2005

[2] J. Hřebíček, J. Ráček; Environmental Information Space I. Theory and Practice. In 6th International Symposium on Environmental Software Systems. Guelph : The International Federation for Information Processing, 2007

[3] J. Hřebíček, J. Ráček, K. Kisza, M. Štefaník; Environmental Information Space II. In Proceedings of the 3rd Internationl Summer School on Computational Biology. Brno, Masaryk University, 2007

[4] J. Hřebíček; The Semantic Interoperability Tools. In Interop-soft. Informační systémy a technologie, základ interoperability krizového řízení ochrany obyvatelstva. Brno : Universita obrany, 2007.

[5] D. Ravishankar, K. Thirunarayan, and T. Immaneni; A modular approach to dokument indexing and semantic search. In ACTA Press, editor, Web Technologies, Applications, and Services, 2005.

[6] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, A. Kirilov.; KIM: a semantic platform for information extraction and retrieval, 2004.

Metodický postup zpracování rámcového procesního modelu

Jan Ministr

Vysoká škola báňská – Technická univerzita Ostrava,
Ekonomická fakulta, katedra Aplikovaná informatika
Sokolská tř. 33, 701 21 Ostrava
jan.ministr@vsb.cz

Abstrakt
Základní podmínkou úspěšné implementace a udržení požadované úrovně procesního řízení v organizaci je kvalitní zpracování a aktualizace rámcového procesního modelu. Při implementaci procesních principů je často managementem opomíjen význam a role rámcového procesního modelu. To v konečném důsledku může vést k nevyužití nabízeného potenciálu procesního řízení, případně neúspěchu celého projektu implementace procesního řízení. Cílem tohoto článku je popsat koncepci rámcového procesního modelu a metodický postup jeho tvorby.

Abstract
The basic precondition of successful implementation and sustain of required level of process management in company is the processing and updating of the model process frame in good quality. The management often forgot about meaning and role of model process frame by implementation of process principles. It can be lead to not use the offered potential of process management in final result or failure the whole project of process management implementation. The goal of this paper is to describe the conception of process model frame and the method of its creation.

Klíčová slova
BPM – Business Process Management, mapa procesu, externí a interní zákazník

Keywords
BPM – Business Process Management, process map, external and internal customer

1 Úvod

Implementace BPM je jedním ze způsobů optimalizace činností firem, který je v současnosti moderním trendem jak úspěšně reagovat na rychle měnící se podmínky podnikání v ČR. Jedná se o kvalifikovaný způsob uchopení této problematiky, aby činnosti dané firmy byly uspořádány, monitorovány a řízeny v rámci zákaznicky orientovaných procesů. Při této činnosti má nezastupitelnou úlohu zpracování rámcového procesního modelu, který přehledně a srozumitelně popisuje identifikované procesy probíhající v celkovém kontextu konkrétní firmy. Hlavní výstup úvodních procesních analýz, které jsou prováděny ve firmách, tvoří

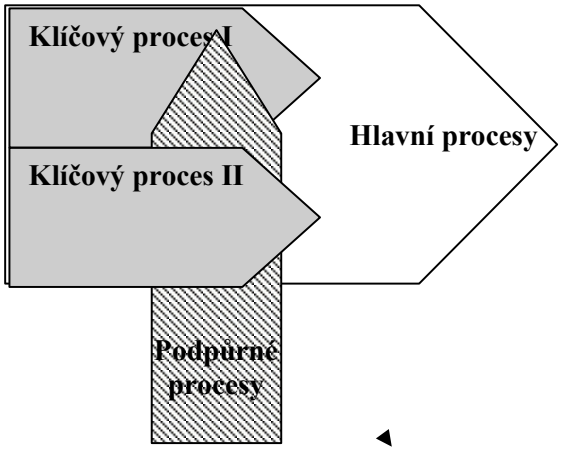
zpravidla sady procesních map, které dokumentují stávající průběh identifikovaných procesů. Jednotlivé aktivity zmapovaných procesů jsou v dobře zpracovaných procesních mapách ohodnoceny koeficienty výkonnosti vzhledem k cílům, které jsou určeny managementem firmy. Takto vytvořená sada procesních map je ale pro management firmy až příliš podrobná a nepřehledná, a proto ji vrcholový management většinou nevyužívá, jak blíže popsáno v článku MINISTR, KUHN (2). Pro své rozhodování potřebuje vrcholový management jednoduchý a přehledný procesní model, který pokrývá komplexně činnost firmy tak, aby na základě jeho obsahu bylo jasné všem úrovním managementu: které procesy jsou pro danou firmu hlavní, a které mají pouze podpůrný charakter pro průběh hlavních procesů. Takovýmto modelem, který splňuje roli komunikačního rozhraní mezi vrcholovým a ostatními úrovněmi managementu při procesním řízení, je rámcový procesní model, jehož úloha je nezastupitelná jak na počátku projektu implementace BPM, tak i v průběhu následného průběžného vylepšování procesů. Opomenutí tohoto významu rámcového procesního modelu může vést od zvýšení nákladu na projekt implementace BPM v dané firmě až po celkový nezdar využívání principů procesního řízení.

2 Rámcový procesní model a jeho zjednodušení

Prvním krokem, který je nutné provést na počátku jakéhokoli projektu implementace BPM, je vytvoření nebo revize stávajícího rámcového procesního modelu. V literatuře se můžeme setkat s mnoha kategoriemi procesů (hlavní, klíčové, podpůrné, vedlejší, řídící sdílené apod.), ale většina projektů využívá rozdělení identifikovaných procesů ve firmě do dvou hlavních skupin, kterými jsou:

- *Hlavní procesy,* které vytvářejí hodnotu pro externího zákazníka. Jejich výsledkem je produkování výstupů, které požaduje externí zákazník. Hlavní procesy podporují nosnou činnost dané firmy, která představuje naplnění její strategické vize a poslání. Na základě konkrétní šíře obsahu poslání a vize lze dále dekomponovat hlavní procesy na *klíčové procesy*.
- *Podpůrné procesy,* jejichž výstupem je tvorba podmínek podporujících funkce hlavních procesů. Vyznačují se tvorbou přidané hodnoty pro interního zákazníka.

Ostatní známé kategorie procesů jako jsou *vedlejší, řídící a sdílené procesy* se vzhledem k jejich celkovému podílu na realizaci cílů dané firmy obvykle pro přehlednost v rámcového procesního modelu neuvádějí. Vzniká tak zjednodušená a přehledná forma rámcového procesního modelu, který je uveden na obrázku č.1.

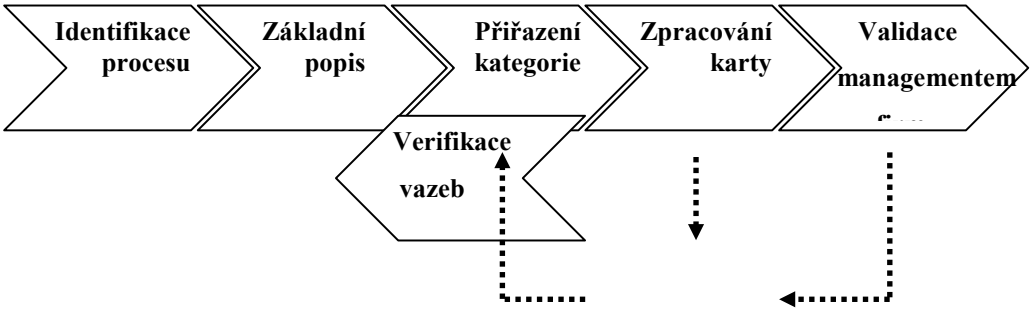


Obr. 1: **Zjednodušený rámcový procesní model**

Počet identifikovaných procesů v jednotlivých kategoriích záleží na zaměření činnosti firmy, u středních firem, které působí v sektoru služeb nebo výroby, se jedná maximálně o 5 klíčových procesů a 15 až 20 podpůrných procesů z závislosti na zvoleném stupni dekompozice procesů. Při tvorbě rámcového procesního modelu je často nutné eliminovat přesvědčení zaměstnanců, že vše co dělají, patří do kategorie hlavních procesů, protože jejich činnost je náročná na čas.

3 Postup tvorby rámcového procesního modelu

Většina autorů se zabývá u rámcového procesního modelu zpravidla pouze rozdělením procesů do jednotlivých kategorií, ale neuvádí to nejtěžší a to je postup jeho tvorby spojený s pracnou identifikací jednotlivých procesů a dodržení jednotného komplexního pohledu na činnost firmy, která je dána strategickými cíly firmy. Celý tento časově náročný proces lze rozdělit do několika kroků, jak je znázorněno na obrázku č.2.



Obr. 2: Postup tvorby rámcového procesního modelu

a. Identifikace procesu

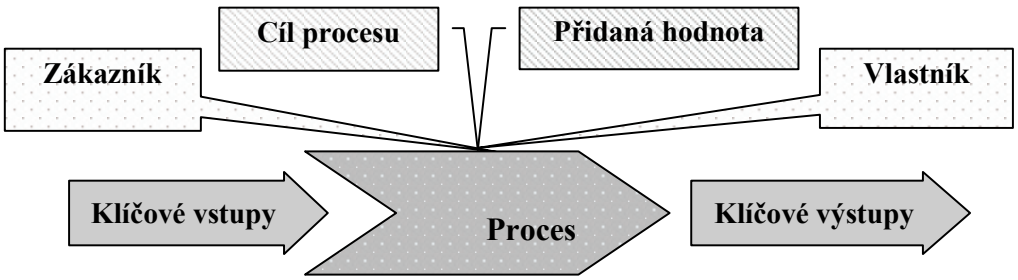
Tento krok tvoří základ celého postupu. Procesní analytik zde musí prostudovat základní dokumenty firmy, ve kterých jsou definovány strategické cíle firmy a prověřit jejich platnost s vrcholovým managementem. Na základě „prověřených“ strategických cílů firmy musí dále procesní analytik:

- Identifikovat klíčové výstupy pro externího zákazníka (výsledné produkty činnosti firmy) a z jakých vstupů jsou postupně tvořeny.
- Identifikovat nutné podmínky, činnosti a produkty, které musí firma zajistit, aby transformace klíčových vstupů na klíčové výstupy proběhla tak, aby byly uspokojeny požadavky externího zákazníka a tak identifikovat proces.
- Pojmenovat proces tak, aby vyjadřoval pomocí slovesné vazby charakteristiku transformace.

b. Základní popis procesu

V tomto kroku je proveden popis základních charakteristik identifikovaných procesů, kterými jsou:

- *Cíl procesu,* který představuje k čemu nám daný proces slouží a čeho průběhem tohoto procesu dosahováno. Nelze jej automaticky zaměnit za cíl existence organizační jednotky, ve které proces probíhá.
- *Přidaná hodnota,* která představuje formu naplnění cíle daného procesu.
- *Zákazník procesu,* který představuje odběratele daného procesu, může jím být např.jiný organizační útvar, individuální zákazník, jiná firma.
- *Vlastník procesu*, který představuje pracovníka, který je zodpovědný za celý průběh daného procesu, za jeho korektnost a správnost. V případě, že daný proces probíhá přes více organizačních útvarů, je nutné navíc uvést pracovníky zodpovědné za danou část procesu.
- *Klíčové vstupy,* které představují veškeré vstupy potřebné ke spuštění daného procesu, tyto vstupy mohou být materiálního charakteru (např. suroviny, polotovary, apod.), nebo nemateriálního charakteru (data, rozhodnutí, apod.).
- *Klíčové výstupy,* které přestavují to, co je produktem daného procesu, co vzniklo jeho průběhem (produkt, služba, písemné vyjádření apod.).



Obr. 3: **Základní charakteristiky procesu**

Tento krok musí být prováděn ve spolupráci v vrcholovým managementem tak, aby bylo dosaženo rozpracování strategických cílů do cílů jednotlivých procesů.

c. Přiřazení kategorie procesu

Zde jsou zkoumány vazby mezi jednotlivými procesy na základě:

- *Typu zákazníka* (externí nebo interní).
- *Typu vazby,* kterou tvoří klíčové vstupy a výstupy.
- *Souvislosti* danými strategickým cílem firmy a cílem procesu.

V tomto kroku může dojít k případnému sloučení nebo rozdělení procesů na základě zkoumaných charakteristik.

d. Zpracování karty procesu

Daný krok podrobněji popisuje identifikovaný proces pomocí tzv. „karty procesu“, která rozšiřuje základní popis procesu o charakteristiky:

- Klíčová legislativa související s procesem
- Hlavní produkty, které jsou používány uvnitř procesu
- Posloupnost činností (základní kroky procesu)
- Základní charakteristiky jednotlivých činností procesu:
  - odpovědnost zaměstnanců
  - význam dané činnosti pro organizační jednotku (klíčová nebo podpůrná)
  - časové trvání průběhu činnosti
  - režim aktivity (průběžný, sezónní, nárazový)
- Základní indikátory (metriky) procesu (výskyt, FTE, indikátory výkonnosti a kvality)

- Spolupráce organizačních útvarů při průběhu aktivit procesu

Zde je vhodné, pokud není k dispozici některý CASE nástrojů, použít Excel, který pro vytvoření rámcového modelu u středních firem zpravidla stačí.

e. Verifikace vazeb rámcového procesního modelu

Karty jednotlivých procesů umožňují lépe zařadit proces do dané kategorie a tím upřesnit primární rámcový procesní model. Úlohou řešitelského týmu v této fázi vývoje rámcového procesního modelu je verifikovat všechny vazby mezi procesy v jednotlivých kategoriích za účelem potvrzení přiřazené kategorie procesu. Tato činnost je prováděna tak dlouho, dokud nedojde k odstranění všech diskrepancí mezi jednotlivými procesy a úspěšné validací managementem firmy.

f. Validace managementem firmy

V tomto kroku procesní analytik vytvoří zpravidla zjednodušený rámcový procesní model firmy v grafické formě, který využívá pouze kategorie hlavních a podpůrných procesů, doplněný kartami jednotlivých procesů. Vytvořený rámcový procesní model musí nakonec projít finální validací, kterou provádí vrcholový management. Tento odsouhlasí strukturu a obsah modelu vzhledem k definované misi a cílům dané firmy, jak blíže diskutováno v článku MINISTR, KUHN (2).

4 Závěr

Odsouhlasený rámcový procesní model slouží k upřesnění rozsahu, strategie a postupu celého projektu implementace procesního řízení, které dále představuje podrobné zmapování stávajícího průběhu jednotlivých procesů a na základě stanovených kritérií navržení zlepšeného průběhu daného procesu.

U většiny projektů na základě požadavku snížení nákladů na projekt není v rámci úvodní analýzy vytvořen nebo aktualizován rámcový procesní model dané firmy jako celku. Následná integrace výsledků pilotního projektu do celkového projektu implementace procesního řízení je pak velmi pracná a nákladná. Autor příspěvku doporučuje na základě rozsahu projektu procesního řízení vytvořit nebo aktualizovat rámcový procesní model firmy jako celku, a tak získat komplexní pohled na průběh procesů dané firmy v souladu se strategickými cíly managementu, a tím harmonizovat její veškeré činnosti firmy.

Kvalitně zpracovaný rámcový procesní model navíc tvoří vhodné komunikační rozhraní mezi jednotlivými úrovněmi managementu při vylepšování průběhu jednotlivých procesů.

5 Literatura

[1] Fiala, J., Ministr, J. Průvodce analýzou modelováním procesů. Ostrava: VŠB-Technická univerzita Ostrava, 2003. 109 s. ISBN 80-248-0500-6

[2] Ministr, J., Kuhn, M. The Role of model process framework in process analysis. In Informační a komunikační technologie pro praxi. Ostrava: VŠB-Technická univerzita Ostrava, Ekonomická fakulta, 2007. s.85-91. ISBN 978-80-248-1522

[3] Jäger, R., Ráček, J. Methodology of the Unified Environmental Information System Development Services Management. In Proceedings of the International Academic Conference on Increasing Competitiveness or Regional, National and International Markets Development - New Challenges. Ostrava : VŠB - Technical university of Ostrava, 2007. s. 89-93. ISBN 978-80-248-1457-5

Metodický rámec implementace servisně orientované architektury

Martin Pochyla

Vysoká škola báňská – Technická univerzita Ostrava,
Ekonomická fakulta, katedra Aplikovaná informatika
Sokolská tř. 33, 701 21 Ostrava
martin.pochyla@vsb.cz

Abstrakt
Tento příspěvek se zaměřuje na návrh metodického rámce implementace servisně orientované architektury. Služby a kompozitní aplikace jsou moderním trendem integrace informačních systémů a podnikových procesů. První část článku je zaměřena na základní prvky, funkce a vlastnosti SOA. Následuje porovnání existujících implementačních postupů a modelů zralosti SOA. Na základě existujících implementačních postupů a modelů zralosti SOA, je navržen metodický rámec implementace SOA, který je směřován především do oblasti malých a středních organizací Tento metodický rámec svými šesti fázemi pokrývá celý životní cyklus implementace SOA do organizace. V rámci vytvořeného metodického rámce jsou strukturovaně popsány náplně jednotlivých fází a činností.

Abstract
This paper focus on field of implementation based on service oriented architecture integration solutions. Firstly, you can compare different SOA definitions. Next part of paper deals with description of main SOA functions and comparison of existing implementation methods and SOA maturity models. Based on existing implementation methods and SOA maturity models new methodical framework of SOA implementation is designed. Methodical framework is targeted mainly to small and middle businesses. Implementation consists of six independent phases that covers full SOA lifecycle implementation. Each phase is fully described by phase purpose, main activities, key documents and critical factors.

Klíčová slova
Servisně orientovaná architektura, služby, webové služby, modely zralosti SOA

Keywords
Service Oriented Architecture, services, web services, SOA maturity model

1 Úvod

U řady českých podniků a organizací je ICT struktura složitým mixem mnoha technologií od různých dodavatelů, které nikdy nebyly stavěny na vzájemnou komunikaci. Správa a náklady na udržení takové architektury často tvoří většinu výdajů na ICT rozpočet firmy. V rámci dynamického vývoje na trhu a s rostoucími požadavky zákazníků a partnerů se tak dostávají

podniky čím dál častěji do složité situace. Narážejí na skutečnost, že díky závislosti na starých systémech a aplikacích se tvoří velká propast mezi tím, kam by se měl podnik strategicky ubírat a tím, co je ještě technologicky možné, proveditelné a financovatelné.

V minulých letech byl nastolen trend integrace podnikových systému a aplikací, který je založen na kompozitním charakteru samotných systémů. Celá oblast je dnes zastřešována servisně orientovanou architekturou (SOA – Service Oriented Architecture). SOA je dnes vnímána jako další vývojový krok směrem k budování nových podnikových informačních systémů a v dnešní době prakticky představuje jedinou alternativu pro návrh, vývoj, provoz a integraci aplikací do ucelených celků založených na kompozitním modelu.

Hlavním důvodem pro nové požadavky na integrace aplikací je důsledné zacílení na firemní procesy a jejich optimalizaci. Základním kamenem tedy již nejsou jednotlivé aplikace, ale procesy, které je využívají. Jednotlivé procesy také překračují jednotlivé aplikace a vnitřní podnikové hranice. Procesy jsou zapojeny do dodavatelských řetězců a tak je žádoucí spolehlivá funkčnost a řízení aplikací, které je zastřešují a realizují. Moderní nasazení informačních systémů a klíčových aplikací musí být každodenně efektivně řízeno a spravováno.

Architektura služeb je důležitá, protože představuje základní rámec, který sjednocuje podnikový model s informačními technologiemi a realizuje funkcionalitu zajišťující efektivní podnikání. Servisně orientovaná architektura se ukazuje jako nejvýznamnější posun za posledních 10 let ve způsobu, jak jsou podnikové aplikace navrhovány, vyvíjeny a nasazovány.

2 SOA – Service Oriented Architecture

Servisně orientovaná architektura umožňuje díky svému principu propojení dříve nesourodých informačních zdrojů, což je věc, která není nijak převratná z pohledu integrace. Princip samotný existuje již řadu let, ale velkého rozšíření se dočkal díky rozšíření jazyka XML a webových služeb.

Při bližším zkoumání dostupné literatury jsem zjistil, že existuje velké množství zdrojů, které vnímají SOA rozdílnými pohledy. V určitých případech dochází u vývojových firem k takovým zjednodušením, že často zapomínají na samotný fakt, že SOA jako taková není produkt, který se dá prodat v jedné krabici, ale jedná se celkový návrh řešení koloběhu a výměny informací ve firmě i mimo firmu.

SOA je definována autory následujícím způsobem. "*SOA je forma technologické architektury, která je založena na principu služeb. Protože je realizována na technologické platformě webových služeb, vytváří SOA potenciál, jak podporovat a rozšiřovat tyto principy v business procesech a nejrůznějších oblastech.*"

Jiná definice SOA „*Architektura zaměřená na služby je architektonický koncept, založený na dobře definovaných, volně vázaných, obchodně zaměřených, opakovaně použitelných sdílených službách.*"

Na SOA je nutné se dívat jako na koncept a návrh architektury podniku. V rámci této oblasti existuje velké množství metodologií a systémů, které popisují tvorbu podnikových architektur. Tyto metodologie popisují podnikovou architekturu pomocí pohledů, abstrakcí nebo modelů. Mezi nejznámější patří Zachmanův rámec [6]podnikové architektury. Zachmanův rámec je zajímavý, protože zahrnuje i procesní model a jádro zlepšování procesů. Pro vyrovnání procesních potřeb a SW zdrojů nabízí tak nový potenciál pro SOA.

Linthicum definuje SOA jako strategicko-technologický rámec, který umožňuje všem podnikovým systémům vystavovat a přistupovat k dobře definovaným službám uvnitř nebo vně podniku. Tyto služby mohou být rovněž abstrahovány na orchestrační vrstvy a kompozitní aplikace.

SOA vychází z kompozitního modelu návrhu, vývoje a provozu aplikací. Kompozitní model je možné charakterizovat jako model, který se skládá z několika samostatných částí a každá tato část je považována za samostatný kompozitní model. Kompozita je složena z jedné nebo více komponent, které nesou příslušnou funkcionalitu daného modulu. Na kompozitním modelu je založena servisně komponentní architektura (SCA – Service Component Architecture), která poskytuje model pro stavbu systémů a aplikací s použitím SOA. Je založena na stávajících zkušenostech s tímto architektonickým stylem a snaží se o systematický přístup k implementaci služeb na principu komponent, kterou poskytují své služby prostřednictvím rozhraní orientovaného na služby.

Kompozitní model umožňuje distribuovaným SW komponentám (službám), aby byly na síti vystaveny, nalezeny, vzájemně využívány/volány, spravovány, komponovány a slaďovány. Co je zde skutečně nového, oproti předchozím technologiím, je kombinace následujících vlastností a okolností:

- Důsledné využívání standardů (XML, SOAP, WSDL, UDDI, HTTP/S, WS-*).
- Míra adopce a podpory těchto standardů ze stran různých dodavatelů i příznivé přijetí ze stran zákazníků.
- Uplatňování volně spojené architektury využitím některé z middlewarových technologií. To je zcela opačný přístup proti těsně spojeným architekturám, které vedou k neudržovatelnému „špagetovému“ efektu.
- Hrubozrnné rozhraní vystavených softwarových komponent.

Architektura orientovaná na služby je postavena na třech klíčových principech. První princip – „podnikové procesy řídí služby a služby řídí technologii“ – znamená, že služby tvoří abstraktní vrstvu, která umožňuje vytvářet vztah mezi podnikovými procesy a technologií. Druhý princip – „podniková agilita“ – znamená schopnost IS/ICT rychle odpovídat na změny požadavků podnikání. Třetí princip předpokládá, že architektura orientovaná na služby se neustále vyvíjí.

[6] The Zachman Institute for Framework Advancement – www.zifa.com

Základní technologie a trendy, které ovlivňují organizaci při implementaci SOA a přeměně organizace na servisně orientovanou, jsou ve své podstatě čtyři. Jsou to webové služby, jazyk XML, architektura založená na službách a řízení podnikových procesů.

3 Existující postupy jak implementovat SOA

V rámci zavádění SOA jsou používány běžné postupy založené na postupných krocích a naplnění cílů. Druhou variantou jsou modely zralosti, které se nezaměřují detailně na postupy, ale stanovují milníky, kterých se má dosáhnout při budování SOA.

a. Implementační postupy

Základní postup implementace v pěti krocích popsal Barry. Autor velmi detailně popisuje jednotlivé kroky uvnitř jednotlivých stupňů implementace nové architektury. Podle mého názoru je tento postup velmi transparentní a jasně vyjadřuje možnosti a cesty jakým může být implementace webových služeb a SOA dosaženo.

Další z možných alternativ popisuje Erl v rámci svého podnikového rámce XWIF. Erl se zaměřuje po technologické stránce na jazyk XML a jeho práce je na tomto jazyku prakticky založena. Jeho metodologie se mi zdá mírně komplikovaná s možná až přílišným zacházením do technických detailů v rámci jednotlivých částí popisu řešení implementace SOA.

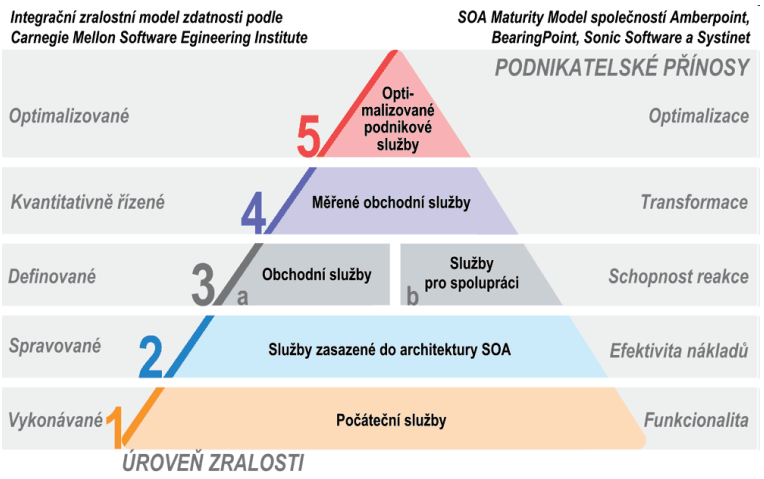
Mírně odlišný je implementační postup firmy BEA, která jako systémový integrátor nabízí řešení vytvořené na základě svých zkušeností získaných u zákazníků. Jako jediná také kombinuje nejenom základní postup implementace SOA, ale také model zralosti jako podpůrný prostředek identifikace stávajícího stavu ve firmě.

Životní cyklus SOA podle firmy IBM, se díky jejímu vlivu a možnostem, jeví jako velmi života schopný. Jeho čtyři základní fáze úplně pokrývají potřebné kroky vedoucí k SOA. Na stránkách firmy IBM[7] je také možnost získat obsáhlé zdroje informací, společně s modelovými příklady realizací životního cyklu SOA.

Ani jeden z výše jmenovaných postupů se však nezaměřuje na konkrétní situaci organizace, ve smyslu její velikosti nebo typu trhu, na kterém podniká. Dle mého názoru není možné plošně využívat implementační postup, bez specifických úprav pro konkrétní situaci trhu, obchodní model, velikost společnosti a ekonomického prostředí. Důležitou roli, také hraje současný stav integračních postupů SOA v rámci organizace, které jsou reprezentovány modely zralosti SOA popisované v následující kapitole.

[7] http://www-306.ibm.com/software/solutions/soa/gov/lifecycle/

b. Modely zralosti SOA



Obr. 1 - SOA Model zralosti – zdroj: (Štumpf 2006)

V rámci implementace SOA existují také přístupy, které jsou založeny na hodnocení vyspělosti podnikové infrastruktury a zralosti organizace pro další krok do vyššího implementačního stádia. Jedná se o modely zralosti nebo také nazývané modely dospělosti (maturity models). Modely zralosti SOA vychází většinou z modelu zralostí CMMI (Capability Maturity Model Integration). Integrační model zralosti CMMI spojuje a rozšiřuje modely zralosti, vytvářené od začátku 90 let v Institutu pro softwarové inženýrství (Software Engineering Institute[8]). Jedná se zejména o Capability Maturity Model for Software, Systems Engineering Capability Model a Hitegrated Product Development Capability Maturity Model.

Nejčastěji zmiňovaným je model zralosti SOA (SOAMM), vytvořený společností Sonic Software se svými partnery – společnostmi AmberPoint, BearingPoint a Systinet. V porovnání se základním modelem CMMI na obrázku č. 1. Model ukazuje zvyšující se přínosy pro podnikání při zavedení vyšší úrovně zralosti SOA a ukazuje, jak se na jednotlivých úrovních zralosti mění cíle, charakteristiky a předpoklady vlivu SOA na podnikání – od nové funkcionality, přes snížení nákladů, dopad na podnikání, transformaci podnikání až po optimalizaci podnikání.

Každá úroveň zralosti definuje cíle, vliv na podnikání, rozsah, důležité průmyslové standardy, klíčové praktiky a kritické faktory úspěchu, jak technologické, tak organizační. SOAMM tak poskytuje návod, jak zavést SOA a měřit pokrok jejího zavádění. SOAMM specifikuje, co je potřeba pro zavedení dané úrovně zralosti SOA – tedy jaké musí být dovednosti, metody, technologie, infrastruktura. Jde zejména o tyto základní prvky:

[8] http://www.sei.cmu.edu/

- metodiky pro analýzu, návrh a implementaci SOA,
- modelovací a vývojové nástroje, které umožní specifikaci a vytváření služeb a jejich propojení na podnikové procesy,
- podniková sběrnice služeb pro poskytování spolehlivé, rozšiřitelné, distribuované komunikace a transformace dat mezi službami a napojení na původní systémy,
- registr a úložiště služeb a politik jako jediné místo pro uspořádání a řízení SOA informací včetně katalogu dostupných služeb, definice jejich rozhraní a politik pro používání služeb,
- nástroje pro provoz a řízení infrastruktury včetně monitorování využívání služeb a zjišťování metrik.

Modely zralosti se liší od předchozích postupů především tím, že nepopisují jednotlivé kroky implementace SOA, ale změny v architektuře podniku a podnikových systémů. Většina těchto modelů pracuje s pěti a šesti úrovněmi, které se liší podle stupně využívání služeb a kompozitních aplikací.

Modely zralosti lze rozdělit podle jejich autorů do tří skupin. První skupinou jsou modely vytvořené softwarovými firmami, které se aktivně podílejí na zavádění SOA do firem – Oracle model zralosti, IBM – Service Integration Maturity Model. Tyto modely jsou často poznamenány nástroji nebo procesy, které firmy používají, a výsledný model je na tyto prvky určitým způsobem naroubován.

Druhou skupinu tvoří konzultantské firmy, které vytváří modely obsahující prvky napomáhající v odkrytí požadavků zákazníků. Typickým příkladem je model zralosti firmy BEA, která založila tvorbu modelu zralosti SOA na dotazníkovém průzkumu u svých potencionálních klientů.

Poslední skupinou jsou modely jednotlivých analytiků nebo nezávislých organizací CBDI Web Services Maturity Model, Linthicum model zralosti, které se mohou soustředit pouze na jedno odvětví průmyslu nebo určité typy organizací. Tyto modely většinou neinklinují ke konkrétním softwarovým nástrojům nebo technologiím, ale na druhou stranu nebývají příliš detailní nebo propracované.

U většiny modelů je možné sledovat rozdělení jednotlivých fází na základní skupiny, které se věnují technologické integraci a obchodní integraci. Nejnižší stupně modelů zralosti se právě věnují převážně integraci technické. Integrace zaměřená na podnikání je definována v závěrečných stupních. Úspěšnost zavedení SOA na základě těchto modelů bude odrážet, jakým způsobem se podařilo propojit tyto technologické a podnikové stupně.

4 Metodický rámec implementace SOA

Na základě prostudování existujících implementačních postupů a modelů zralosti SOA, jsem se pokusil o vytvoření základního metodického rámce implementace webových služeb respektive servisně orientované architektury. Na základě svých předchozích průzkumů, jsem tento metodický rámec připravil především pro malé a střední organizace, které jsou také tlačeny svým okolím a potřebami do využívání služeb a SOA samotné.

Před každou implementací je velmi důležité věnovat pozornost možným konfliktům, které mohou vzniknout na různých úrovních organizace. Většina problémů je způsobena různými pohledy na určitou problematiku. Nejvíce problematická je často komunikace s ICT oddělením, které často nevnímá problematiku ICT a podnikání zároveň. Do většiny problémů je zatažen především vedoucí ICT oddělení, protože musí zodpovídat za investice do ICT. Ty jsou často jen velmi těžce přesně doložitelné vzhledem k očekávaným přínosům. Podobně je tomu při implementaci SOA, kde jsou počáteční investice velmi vysoké, ale jen velmi složitě vyčíslitelná návratnost.

ICT projekty pro jednotlivá oddělení si mohou často konkurovat nebo doplňovat a je velmi důležité, aby svou roli sehrál vedoucí ICT oddělení jako koordinátor projektů a ovlivňoval tak jednotlivá řešení. Příkladem může být znovupoužitelnost, kde u nových projektů je důležité zjistit, jestli neexistují hotová řešení pro některé služby nebo funkce. Projektoví manažeři, ale jen velmi neradi otevírají již uzavřené projekty, aby byly nalezeny některé formy znovupoužitelnosti. K řízení těchto situací velmi dobře napomáhá procesní a znalostní management.

a. Základní struktura fází metodického rámce

Základní implementační projekt SOA jsem definoval na základě šesti hlavních fází, které jsou zobrazeny na obrázku č. 2. Jednotlivé fáze na sebe přímo navazují a není možné při prvním spouštění implementace fáze libovolně vynechat nebo provádět v jiném pořadí. Protože je však implementace SOA kontinuální proces, existují dvě zpětné vazby, které zajišťují přírůstkový charakter této metodiky. První koloběh spočívá v opakování fázi druhé až čtvrté, které zajišťují pilotní projekt, návrh, tvorbu a implementaci nových služeb. Tato zpětná vazba funguje za předpokladu, že organizace se stále pohybuje na stejné úrovni, co se týče zralosti služeb a celé SOA architektury. Právě pro přechod na novou úroveň implementace SOA je určena druhé zpětná vazba, která zajišťuje propojení mezi fází poslední a fází první.

NULTÁ FÁZE – PŘÍPRAVNÉ PROCEDURY

CÍL FÁZE: Hlavním cílem této fáze je, uvědomit si, jestli je nutné služby a SOA implementovat nebo postačí jiný typ integračního řešení.

ÚČEL a OBSAH: V rámci této fáze je nutné odpovědět na základní otázky, které dopomůžou v orientaci jaké řešení zvolit a jakým směrem se bude pokračovat v integraci stávajících a nových systémů v organizaci. V rámci specifikace musí být zřejmý směr a odpovědnost za

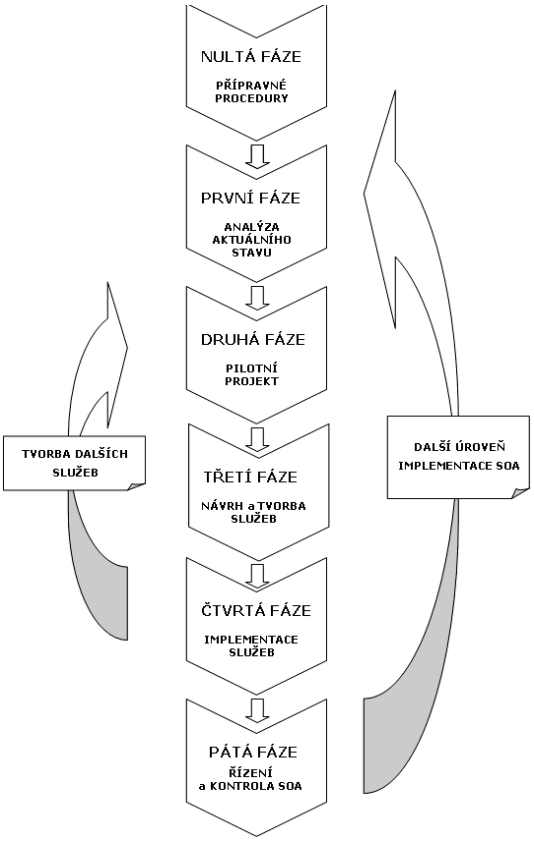
následující kroky implementace SOA. Součástí této fáze jsou také první kroky vedoucí k získání podpory u všech zainteresovaných stran.

ČINNOSTI:

1 Definování krátkodobých a dlouhodobých cílů.

2 Zajištění rozpočtu pro plánované cíle v rámci první etapy implementace.

3 Zajištění podpory pro implementaci SOA u vedení organizace.



Obr. 2 - Metodický rámec implementace SOA

.

PRVNÍ FÁZE – ANALÝZA AKTUÁLNÍHO STAVU

CÍL FÁZE: Důkladná analýza aktuálního stavu v organizaci a tím vytvoření podmínek pro naplánování kroků vedoucích k dosažení vytýčených cílů a vhodné sestavení implementačního týmu SOA.

ÚČEL a OBSAH: V rámci této fáze jsou popsány všechny primární systémy a jejich funkce, které organizace využívá. Popis systému by se neměl omezit jen na výčet funkcí, ale také na popis stávajících technologických řešení a propojení systémů mezi sebou. Analýza se týká pouze těch systémů, které se podílejí nebo budou podílet na dosažení cílů, které byly definovány v předchozí fázi. Získané informace dopomůžou k stanovení aktuální pozice na stupni modelu zralosti SOA.

ČINNOSTI:

1 Ustanovení hlavních členů implementačního týmu.

2 Analýza a popis informačních systémů.

3 Analýza a popis propojení systémů mezi sebou.

4 Analýza informací a dat do systému vstupujících.

5 Analýza informací a dat ze systému vystupujících.

DRUHÁ FÁZE – PILOTNÍ PROJEKT

CÍL FÁZE: Stanovení rámce, obsahu a detailní naplánování pilotního projektu implementace SOA. Zároveň je v rámci této fáze založen implementační tým zodpovědný za naplnění jednotlivých fází pilotního projektu a fází následujících.

ÚČEL a OBSAH: Pilotní projekt je impulsem pro nastartování implementace SOA a počátek využívaní služeb v rámci podnikové infrastruktury. Součástí pilotního projektu, který bude vybrán, půjde o aplikaci architektury služeb na vybrané segmenty organizace nebo na skupinu funkcí, které podnik využívá pro naplnění svých obchodních cílů. V následných hlavních cyklech implementace SOA, půjde v této fázi o definování dalších cílů a oblastí vhodných pro implementaci služeb a kompozitních aplikací. Fáze svým obsahem vždy naplňuje iniciační a koncepční potřeby hlavního cyklu implementace SOA.

TŘETÍ FÁZE – NÁVRH a TVORBA SLUŽEB

CÍL FÁZE: Navrhnout a vytvořit nové služby, které budou sloužit k naplnění cílů SOA projektu.

ÚČEL a OBSAH: V rámci této fáze členové implementačního týmu společně s řešitelskou firmou provedou detailní analýzu vytypované oblasti, a na základě analýzy navrhnou a vytvoří služby, které budou využívány v rámci pilotního projektu. Jednotlivé činnosti této fáze, by se měly striktně držet životního cyklu návrhu a tvorby služeb.

ČINNOSTI:

1 Analýza podnikových procesů v rámci řešených modulů a činností.

2 Tvorba katalogu priorit a potřeb jednotlivých služeb.

3 Návrh a tvorba služeb.

4 Testování služeb na neprovozním prostředí.

ČTVRTÁ FÁZE – IMPLEMENTACE SLUŽEB

CÍL FÁZE: Implementace a reálné spuštění služeb v rámci aplikačního rámce podnikových informačních systémů nebo portálových řešení.

ÚČEL a OBSAH: V rámci této fáze je nutné vytvořené služby postupně zavést do reálného provozu. To představuje jejich implementaci do struktury rámce informačního systému nebo napojení na portálová řešení, které využívá organizace nebo její partneři. Jednotlivé činnosti této fáze, by se měly striktně držet životního cyklu služeb.

ČINNOSTI:

1 Začlenění služeb do struktury portálového řešení.

2 Zabezpečení služeb.

3 Tvorba kompozitních aplikací a orchestrace služeb.

4 Testování služeb v reálném provozu.

5 Hodnocení výkonnosti služeb.

6 Školení uživatelů a tvorba dokumentace.

PÁTÁ FÁZE – ŘÍZENÍ a KONTROLA SOA

CÍL FÁZE: Hlavní prioritou této fáze je konsolidace a ustálené fungování nově implementovaných služeb a kompozitních aplikací. Kompletní kontrola nad funkcemi, poskytováním a bezpečností jednotlivých služeb.

ÚČEL a OBSAH: Primárním účelem této fáze je zavedení a provádění organizačních politik, pravidel a procesů, které pomohou s řízením služeb a architektury SOA. Mezi hlavní operace patří sledování komunikace a přenos zpráv mezi službami a zobrazení těchto informací v kontextu technickém, infrastrukturálním nebo obchodním. Jednotlivé činnosti jsou prováděny opakovaně a kontinuálně.

ČINNOSTI:

1 Definice politik, pravidel a procesů pro řízení SOA.

2 Zjištění poskytovatelů a konzumentů služeb.

3 Správa registrů a metadat jednotlivých služeb.

4 Mapování toku zpráv a sledování vazeb mezi službami.

5 Detekce a eliminace nebezpečných služeb.

b. Hlavní rizikové oblasti implementace SOA

Součástí každého implementačního projektu by také měla být oblast, která řeší případná rizika a jejich odstranění. Mezi všeobecná pravidla, a jež se dotýkají oblastí, které je nutné sledovat, patří:

1 V jednu chvíli se soustředit pouze na jednu věc. Pokud je to možné, omezit provádění několika změn ve stejný čas.

2 Snažit se minimalizovat rozsah. Implementovat jen po malých segmentech organizace tak, že se problémy řeší s menší skupinou.

3 Koordinovat změny společně s obchodním cyklem firmy, čímž se zamezí kritickým stavům nebo extrémnímu zatížení organizace v nevhodném časovém období.

4 Připravit uživatele na změnu. Změna často předpokládá také nové mechanismy a procesy a je důležité, aby tyto změny byly pochopeny a lidé připraveni na dopady změn.

Z pohledu implementačního projektu, který je úzce zaměřen na SOA, je dále možné narazit na tyto problémové oblasti:

1 Opoždění konečného termínu – v případě, že není naplněn stanovený termín na začátku projektu, je potom nový termín několikrát posunut.

2 Nabobtnání projektu – během průběhu projektu se často stává, že jsou přidávány nové funkcionality, které dělají projekt nepřehledným a často mají za důsledek první výše zmiňovanou rizikovou oblast.

3 Zastaralá technologie a standardy – v rámci déletrvajících projektů je nutno počítat se zastaráním používaných technologií. Velkým rizikem je pro mnohé firmy právě velmi rapidní vývoj na poli standardů, které se dotýkají SOA a webových služeb.

4 Vazby na technologie – v rámci implementací řešení založených na SOA, je nutné zásadně oddělit technologie od podnikání a nesnažit se pouze o implementaci technologie.

Bez vhodné struktury SOA je riziko provádění změn velké, protože prováděné změny mohou být velmi nákladné nebo naopak málo účinné. Při využití SOA v organizaci, a tím dosažené možnosti provádět změny flexibilně a rychle, získá podnik ještě další výhody. Jeden z hlavních omylů, které jsou často v souvislosti se SOA zmiňovány, je fakt, že SOA se nerovná pouhému využití webových služeb. V rámci SOA mohou webové služby představovat jen jednu z forem komunikace v rámci této architektury. Toto je častý omyl, který může ve svém důsledku přinášet velké problémy.

5 Závěr

Implementace SOA není jednoduchá a také nedělá oblast informačních technologií jednodušší. Velmi však pomůže v situacích, kdy je potřebná velká schopnost adaptace změn na trhu, a časté napojení na informační zdroje obchodních partnerů nebo zákazníků.

Pro zrychlení procesu implementace změn a usnadnění pokrytí celých firemních procesů jsou vyvíjeny různé postupy a metodologie, který ve spolupráci s vyššími jazyky jako je WS-BPEL a událostně řízenou architekturou umožňují vytvořit pružný informační systém, podporující firemní procesy na principu jednoduše přístupných služeb a kompozitních aplikací. Ne vždy jsou však tyto technologie realizovatelné nebo vhodné pro obchodní modely, které využívají malé a střední organizace.

Větší rozšíření webových služeb a SOA samotné, je však dnes stále limitováno některými nedotaženými aspekty implementace. Právě díky této skutečnosti, jsem si jako cíl své práce stanovil vypracování všeobecného metodického rámce implementace SOA řešení, který by měl být nezávislý na technologických variantách řešení SOA. Své řešení jsem směřoval na segment malých a středních organizací, které vnímají integraci z mírně odlišného pohledu než velké korporátní společnosti. Metodický rámec svou náplní pokrývá celý životní cyklus implementace služeb a dalších prvků tvořících architekturu založenou na službách.

Dnešní doba se vyznačuje neustálými změnami. To samozřejmě platí i pro ekonomické prostředí, ve kterém se pohybují české společnosti. Každou změnu vnějšího prostředí je nutno ve většině případů promítnout i do vnitřního fungování společnosti. Právě rychlost provádění změn a použité technologie jsou ty faktory, která odlišuje ty nejlepší od těch ostatních. Věřím, že metodický rámec implementace SOA bude přínosem pro organizace, které hledají řešení a způsob, jak využít SOA při budování své strategické výhody.

6 Literatura

[1] Arsanjani, A.: *Increase flexibility with the Service Integration Maturity Model (SIMM)*, September 2005, [online] Dostupné na Internetu [cit. 1. 2.2007] http://www-128.ibm.com/developerworks/webservices/library/ws-soa-design1/

[2] Bachman, J. *Movin' SOA On Up* [online] Dostupné na Internetu [cit. 18.10.2006] http://www.sonicsoftware.com/solutions/learning_center/soa_insights/

[3] Barry, D. *Web services and Service-Oriented Architecture: The Savvy Manager's Guide*, Morgan Kaufmann Publishers, 2003, 245s. ISBN: 155-8609-067

[4] Bloomberg, J. Schmelzer, R. *Service Orient or Be Doomed!,* 2006, New Jersey, Wiley, ISBN: 0-471-76858-8

[5] Crotty, C. Oracle *Defines an SOA Maturity Model* [online] Dostupné na Internetu [cit. 10.01.2007] http://internet.ziffdavis.com/oraclefz/article_soa_maturity_model.html

[6] Fiala, J., Ministr, J. *Průvodce analýzou a modelováním procesů*. VŠB-Technická univerzita Ostrava, 2003. 110 s. ISBN 20-248-0500-6

[7] Groves, D. *Successfully Planning for SOA*, [online] Dostupné na Internetu [cit. 11.02.2007] http://dev2dev.bea.com/pub/a/2005/11/planning-for-soa.html

[8] High, R. IBM's SOA *Foundation An Architectural Introduction and Overview*, November 2005, [online] Dostupné na Internetu [cit. 18.02.2007] http://www.ibm.com/developerworks/webservices/library/ws-soa-whitepaper/#download

[9] IBM *Composite Models* [online] Dostupné na Internetu [cit. 20.06.2007] http://publib.boulder.ibm.com/infocenter/rtnlhelp/v6r0m0/index.jsp?topic=/com.ibm.xtools.comparemerge.doc/topics/ccompmods.html

[10] Krafzig, D. Banke, K. Slama, D. *Enterprise SOA: Service-Oriented Architecture Best Practices*, Prentice Hall PTR, 2004, ISBN: 0-13-146575-9

[11] Newcomer, E., Lomow, G. *Understanding SOA with Web Services*, Addison Wesley, 2004, ISBN: 0-321-18086-0

[12] Pochyla, M. SOA pro malé a střední podniky. In *MEKON 2007*, Výsledky vědecké práce studentů doktorského studia. Ostrava: VŠB-TU Ostrava, 2007. vol. 9, s. 171., ISBN: 978-80-248-1324-0

[13] Ráček, J. Strukturovaná analýza systémů. Brno : Masarykova univerzita, 2006. 104 s. ISBN 80-210-4190-0

[14] Wolf, P., Wolf, J. 2005. Procesní a znalostní management v organizaci. In Acta academica karviniensia. Opava; Slezská univerzita v Opavě, Obchodně podnikatelská fakulta v Karviné, 2/2005, s. 187-193. ISSN 1212-415X.

Návrh metodického postupu použití objektových databází při tvorbě webových aplikací na platformě Java Enterprise Edition

Vítězslav Novák

VŠB-TU Ostrava, Ekonomická fakulta
Katedra aplikované informatiky
Sokolská třída 33, 701 21, Ostrava
vitezslav.novak@vsb.cz

Abstrakt
Webové aplikace jsou jedním z mnoha typů aplikací, které lze pomocí programovacího jazyka Java vytvářet. Tyto webové aplikace dnes využívají nejrozšířenější typ databází, tedy databáze relační, a tomu jsou přizpůsobeny i nástroje, které tyto aplikace používají. Protože ale programovací jazyk Java je jazyk objektově orientovaný, bylo by mnohem přirozenější používat i databáze objektově orientované, aby nemuselo docházet k mapování objektů do relačních tabulek. Tento článek by měl načrtnout možnosti a postupy pro použití objektově orientovaných databází při vývoji webových aplikací na platformě Java Enterprise Edition.

Abstract
Web applications are some of the application type, which is possible to create by the help of programming language Java. These web applications use mostly relational database systems and adjusted development tools. Because programming language Java is object oriented programming language and it would be more natural use object oriented databases to by unnecessary mapping object into relational tables. This article outline possibilities and processes for using object oriented databases at development web application on Java Enterprise Edition platform.

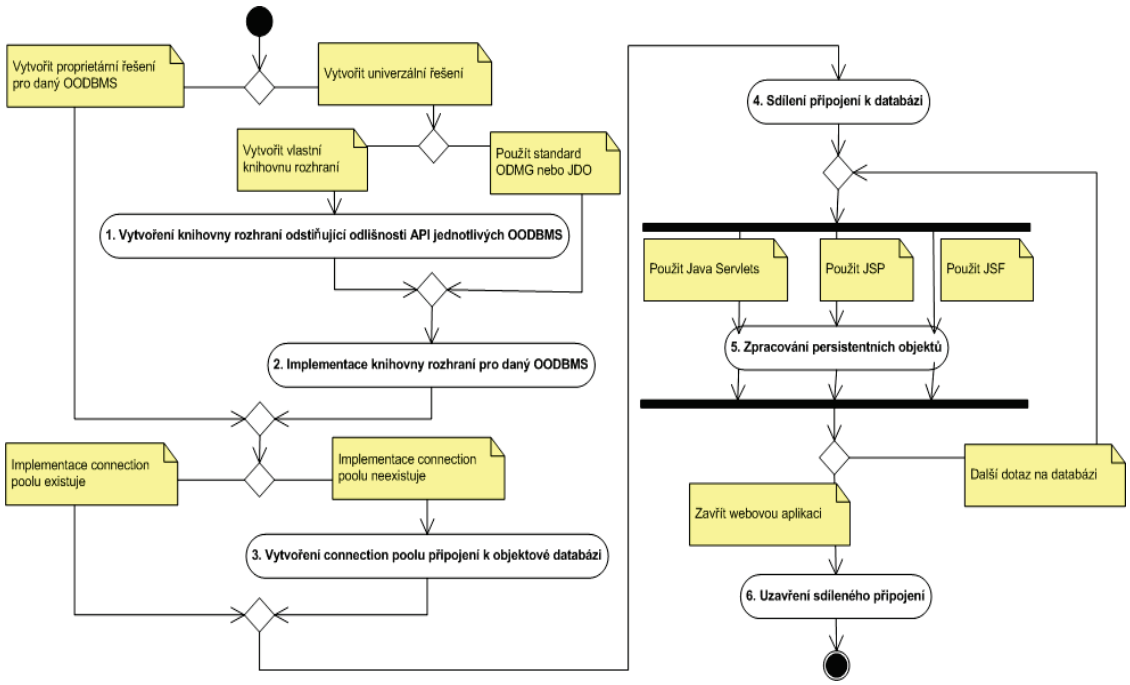
Klíčová slova
Java Enterprise Edition, webová aplikace, relační databáze, objektově orientovaná databáze, třída, objekt, Java Servlet, JavaServer Pages (JSP), JSP Standard Tag Library (JSTL), JavaServer Faces JSF), vlastní značky, ObjectDB, db4objects, Caché.

Keywords
Java Enterprise Edition, web application, relational database, object-oriented database, class, object, Java Servlet, JavaServer Pages (JSP), JSP Standard Tag Library (JSTL), JavaServer Faces (JSF), custom tags, ObjectDB, db4objects, Caché

1 Úvod

V současné době se bohužel v nástrojích na platformě Java Enterprise Edition, které používají webové aplikace, ať už se jedná např. o knihovnu JSTL, komponenty JSF nebo Java EE aplikační servery, s objektově orientovanými databázemi nepočítá. Tyto nástroje jsou přizpůsobeny pro použití hlavně s relačními databázemi. Přesto zase není až tak velký problém ve webových aplikacích Java EE objektově orientované databáze použít. Tento článek si právě klade za cíl navrhnout metodický postup, jak použít objektově orientovanou databázi při tvorbě webové aplikace na platformě Java Enterprise Edition.

Celý návrh metodického postupu přirozeně vychází ze způsobu použití relačních databází ve webových aplikacích a pouze jej přizpůsobuje pro objektově orientované databáze. Celý metodický postup lze znázornit následujícím diagramem aktivit:



Obr.1: Návrh metodického postupu.použití objektových databází při tvorbě webových aplikací na platformě Java Enterprise Edition

Stručně lze postup popsat: vývojář webové aplikace se musí nejdříve rozhodnout, zda-li bude vytvářet proprietární řešení pro zvolený OODBMS nebo bude chtít vytvářet universální řešení přenositelné mezi různými OODBMS. Pokud se rozhodne pro tvorbu universálního řešení, musí vytvořit jakousi softwarovou mezivrstvu, která odstíní jednak rozdíly v API jednotlivých OODBMS, jednak ale také odstíní vývojáře od složitosti API jednotlivých OODBMS. Jako softwarovou mezivrstvu může zvolit existující specifikace ODMG pro programovací jazyk Java

nebo JDO. Tyto specifikace jsou však dosti složité a proto je pro vývojáře webové aplikace určitě snadnější vytvořit si vlastní softwaru mezivrstvu.

Ve webových aplikacích je důležité s každým HTTP požadavkem neustále nevytvářet další a další připojení k databázi, což je náročné na zdroje a čas, ale tato připojení si předalokovat do tzv. connection poolu. Některé objektově orientované databáze coonection pool poskytují, některé ne. U těch je pak nutné connection pool implementovat. Máme-li vytvořeno připojení k databázi, musíme jej nasdílet pro všechny komponenty webové aplikace. Ty si jej pak půjčují, použijí jej podle typu použité technologie a pak jej zase vrací do connection poolu. Při ukončení běhu webové aplikace je nutné všechna předalokovaná připojení connection poolu uzavřít.

2 Vytvoření knihovny rozhraní odstiňující odlišnosti API jednotlivých OODBMS.

V případě, že nevyhovují existující standardy, je výhodnější vytvořit vlastní aplikační rozhraní podle návrhového vzoru Fasáda, který odstíní jak nejednotnost v API jednotlivých OODBMS, tak složitosti API jednotlivých OODBMS. Toto vlastní rozhraní bude obsahovat pouze potřebné metody pro potřeby webových aplikací. Z principů používaných při práci s relačními databázemi, ze studia standardů ODMG, JDO i některých OODBMS a také vlastní praxí jsem dospěl k následující základní struktuře požadovaného aplikačního rozhraní (základní rozhraní mají název tučným písmem) viz Obr. č. 2.:

Jak lze z obrázku zjistit, jedná se o malou knihovnu několika rozhraní a tříd, která by byla snadno distribuovatelná do webových aplikací jako Java Archiv. Já jsem si tuto knihovnu nazval JODBC (Java Object Database Connectivity). Navrhovaná knihovna JODBC obsahuje podobná rozhraní, která obsahuje jak technologie JDBC, tak každý OODBMS. Následně popíšu jednotlivá rozhraní atřídy této knihovny a jejich metody.

`ObjectDatabase` – odpovídá rozhraní **`java.sql.Connection`** technologie JDBC. Reprezentuje připojení k objektově orientované databázi a umožňuje manipulaci s daty, čímž nahrazuje jazyk OML (Object Manipulation Language). Umožňuje základní operace s databází, jako je získávání dotazu, řízení transakcí, vytváření, ukládání a odstraňování objektů v databázi a uzavření připojení.

Poslední dvě metody nejsou příliš běžné. Souvisejí s problémem používání jedinečného identifikátoru objektů OID. V případě používání relačních databází obsahuje každá tabulka primární klíč, podle kterého je možné záznamy v tabulkách vyhledávat. Tento primární klíč je jedním z atributů každé tabulky, takže jej lze zjistit naprosto běžně jako hodnotu každého dalšího atributu tabulky. Uobjektově orientovaných databází je situace poněkud jiná. Programátor se o jedinečný identifikátor objektů, tedy o něco, co je podobné primárnímu klíči u relačních databází, vůbec nemusí starat, protože jej každému objektu přiřadí sama objektově orientovaná databáze. To je určitě výhoda. Ve webových aplikacích nelze ale do webové stránky umístit například do odkazu, který má identifikovat nějaký objekt, samotný objekt, ale pouze řetězec, který jej jednoznačně identifikuje a podle kterého lze tento objekt z databáze

získat. A pro tento účel jsou zapotřebí metody, které jsou schopné vrátit jedinečný identifikátor objektu jako řetězec a podle tohoto řetězce zase tento objekt z databáze vrátit.

`ObjectDatabasePool` – odpovídá rozhraní **`javax.sql.DataSource`** technologie JDBC a reprezentuje connection pool na předalokovaná připojení k objektově orientované databázi.

`ObjectDatabaseQuery` – odpovídá rozhraní **`java.sql.Statement`** technologie JDBC, reprezentuje dotaz na objektově orientovanou databázi a umožňuje tyto dotazy spouštět.

Z výpisu kódu výše lze odvodit dva problémy týkající se dotazů. Prvním problémem je, že každý OODBMS používá jiný a naprosto rozdílný objektový dotazovací jazyk, druhým pak je datový typ, který vrací metoda provádějící dotaz.

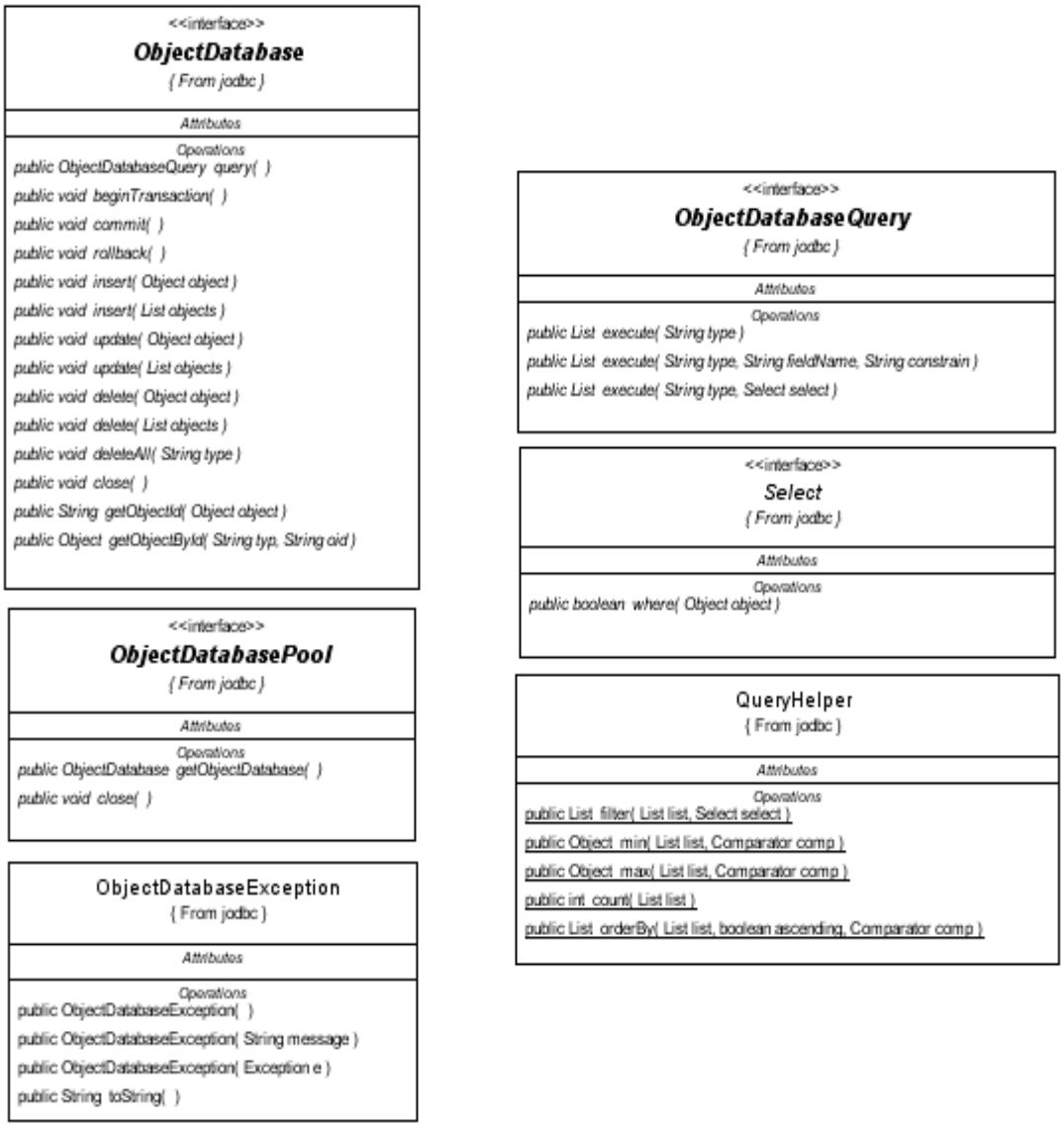
Při hledání řešení těchto problémů jsem využil toho, že jednotlivé objektové dotazovací jazyky mají přece jen něco společného. Každý dotaz totiž obsahuje:

- název třídy, jejichž instance budou z databáze vybírány,
- název atributu, na kterém bude výběr prováděn,
- omezení pro filtrovací atribut,
- dotaz vrací nějaký typ kolekce.

Těchto shodných prvků lze využít pro vytvoření univerzálního dotazovacího jazyka, který jsem si zase soukromě nazval JODBC Query Language (JODBCQL).

Dotazovací jazyk JODBCQL by měl splňovat následující podmínky:

- musí být implementovatelný na libovolný objektový dotazovací jazyk,
- pro snadnost použití ve webových aplikacích Java EE by všechny parametry dotazovacích metod měly být pokud možno typu **`java.lang.String`**,
- bude vracet jednotný typ kolekce – **`java.util.List`**.



Obr. 2: Základní rozhraní a třídy nutné pro odstínění rozdílů mezi API jednotlivých OODBMS

V objektovém dotazovacím jazyku JODBCQL je možné dotazy zadávat pomocí následujících tří metod rozhraní **`ObjectDatabaseQuery`**:

```
java.util.List execute(java.lang.String type)
                                        throws
ObjectDatabaseException
```

Tato metoda vrátí extent daného datového typu. Má následující parametr:

- **type** – plný název třídy, jejichž instance budou z databáze vybírány, např: "cz.vsb.ekf.nov21.BeznyUcet".

***Příklad použití metody**: vybrat všechny knihy z databáze Knihkupectví (fiktivní databáze).*

```
ObjectDatabase db = pool.getObjectDatabase();
ObjectDatabaseQuery query = db.query();
List knihy = query.execute("cz.vsb.ekf.nov21.knihkupectvi.Kniha");
```

Jak je vidět z příkladu, tato metoda je vhodná pouze pro zadávání velmi jednoduchých dotazů, kde výsledkem dotazu má být vždy celý extent daného datového typu.

```
java.util.List execute(java.lang.String type,
                       java.lang.String fieldName,
                       java.lang.String constrain)
                                        throws
ObjectDatabaseException
```

Tato metoda vrátí seznam objektů vyhovující zadanému kritériu. Má následující parametry:

- **type** – plný název třídy, jejichž instance budou z databáze vybírány, např: "cz.vsb.ekf.nov21.BeznyUcet".
- **fieldName** – název atributu, na kterém bude výběr prováděn, např: nazevUctu.
- **constrain** – výraz, vyjadřující kritérium výběru, např: "Novák". Protože se ale způsob zadávání kritérií u jednotlivých OODBMS značně liší, je pravděpodobné, že toto omezení bude závislé na daném OODBMS a s přechodem na jiný OODBMS jej bude nutné upravit.

***Příklad použití metody**: vybrat všechny knihy autora "Pitner, T." z databáze Knihkupectví.*

```
ObjectDatabase db = pool.getObjectDatabase();
ObjectDatabaseQuery query = db.query();
List objekty = query.execute("cz.vsb.ekf.nov21.knihkupectvi.Kniha",
                             "autor","Pitner, T.");
```

Tato metoda vychází z jazyka JDOQL verze 1.0, který je také velmi jednoduchý. Jak je vidět z příkladu, tato metoda umožňuje použití pouze jednoduchého výběrového kritéria, které se navíc může lišit podle použitého OODBMS.

```
java.util.List execute(java.lang.String type,
                       cz.vsb.ekf.nov21.jodbc.Select select)
                       throws ObjectDatabaseException
```

Tato metoda vrátí seznam objektů vyhovující zadanému kritériu. Má následující parametry:

- **type** – plný název třídy, jejichž instance budou z databáze vybírány, např: "cz.vsb.ekf.nov21.BeznyUcet".

- **select** – implementace rozhraní **`cz.vsb.ekf.nov21.jodbc.Select`**. Toto rozhraní má deklarovánu jedinou metodu **`boolean where(java.lang.Object)`**, ve které je nutno při jejím překrytí implementovat vyhledávací podmínku a to pomocí výrazu v programovacím jazyce Java.

Tato metoda vychází z metody Native Queries, kterou používá OODBMS db4o, a spočívá v tom, že je z databáze vybrán základní extent typu daného parametrem type a v něm je každý prvek porovnám metodou **`boolean where(java.lang.Object object)`** s nastaveným kritériem výběru v implementaci rozhraní **`cz.vsb.ekf.nov21.jodbc.Select`**. Prvek, který tomuto kritériu vyhovuje, je vložen do výsledného seznamu. Pomocí této metody lze provádět i složité dotazy na objektově orientovanou databázi a to dokonce bez znalosti nějakého speciálního dotazovacího jazyka, jakým je SQL nebo JDOQL, nýbrž pouze se znalostí programovacího jazyka Java.

Pro další úpravu výsledku dotazu je možno využít statických metod třídy **`java.util.Collections`**. Tato třída je knihovní třídou programovacího jazyka Java a obsahuje statické metody, které umožňují takové operace s kolekcemi, jako je řazení, nalezení minima a maxima, míchání atd. Takovouto pomocnou třídu je také možné si vytvořit (viz třída **`cz.vsb.ekf.nov21.jodbc.QueryHelper`**) a doplnit o další potřebné metody jako např. možnost dofiltrovat si výsledný seznam objektů a jiné.

Většina metod rozhraní JODBC vyhazuje výjimku **`ObjectDatabaseException`**, kterážto by také měla být součástí knihovny JODBC. Knihovnu JODBC je samozřejmě možné dále doplnit o další rozhraní a jejich metody, pokud by si webová aplikace vyžádala nějakou dodatečnou funkčnost.

3 Implementace knihovny rozhraní odstiňující odlišnosti API jednotlivých OODBMS pro daný OODBMS.

Vybranou knihovnu odstiňující odlišnosti v API jednotlivých OODBMS, např. knihovnu JODBC, je dále nutné implementovat pro vybraný OODBMS a vytvořit tak vlastně jakýsi JODBC driver, což bude také Java Archiv s potřebnými třídami, které implementují rozhraní zvolené knihovny. Výhoda tohoto JODBC driveru spočívá v tom, že při přechodu aplikace na jiný OODBMS stačí vyměnit pouze tento JODBC driver a zbytek kódu aplikace se prakticky nezmění.

Vytvoření a implementace vybrané knihovny odstiňující odlišnosti API jednotlivých OODBMS se může zdát pracná, na druhou stranu ale nejspíše pro tvorbu webových aplikací bude vybrán jeden OODBMS a ten bude používán ve více webových aplikacích. Z toho vyplývá, že vytvoření a implementace takovéto knihovny, případně implementace connection poolu pro vybraný OODBMS bude prováděna pouze jednou a vytvořené řešení pak bude recyklováno v dalších webových aplikacích s tím, že stále zůstává otevřena možnost snadno přejít na jiný OODBMS. Stačilo by pouze implementovat vybranou knihovnu pro tento jiný OODBMS.

Pokud ale bude rozhodnuto vytvářet proprietární řešení pro konkrétní OODBMS, není nutné implementovat žádná rozhraní. Je ale nutné vytvořit connection pool připojení k dané objektově orientované databázi, pokud není v daném OODBMS implementován, a to podle principů uvedených dále, jen s použitím API vybraného OODBMS.

4 Vytvoření connection poolu připojení k objektové databázi.

Vytvořit vlastní connection pool není zcela triviální činnost. Jako základní vzor je možno použít implementaci uvedenou v [6] a upravit ji pro potřeby objektově orientovaných databází. Tuto implementaci zde pro svůj rozsah neuvádím.

5 Sdílení připojení k objektové databázi a uzavření sdíleného připojení k objektové databázi.

Vytvoření sdílení připojení k objektově orientované databázi a uzavření tohoto sdíleného připojení spolu velmi úzce souvisejí a proto budou popsány na jednom místě.

První problém, který je nutné při vytváření webové aplikace vyřešit, je vytvoření sdílení jedné kopie instance connection poolu pro všechny komponenty webové aplikace při startu webové aplikace a při ukončení webové aplikace tento objekt zase korektně uzavřít. Podle mého názoru pro vyřešení tohoto problému můžeme využít následující tři možnosti, které budou dále vysvětleny:

- využít metodu **`void init()`** a **`void destroy()`** controller servletu,
- využít metod posluchače typu **`javax.servlet.ServletContextListener`**,
- implementovat objekt typu **`ObjectDatabasePool`** jako singleton.

Využití metody `void init()` a `void destroy()` controller servletu pro vytvoření sdílení connection poolu

Základním principem vytvoření sdílení connection poolu pro všechny komponenty webové aplikace je jeho uložení jako atributu v rozsahu application. K tomuto řešení je nutné podotknout, že jej nelze použít spolu s frameworkem JavaServer Faces. Webovou aplikaci sice řídí controller servlet typu **`javax.faces.webapp.FacesServlet`**, ten je ale součástí API knihovny JSF a není možné jej měnit.

Využití metod posluchače typu `javax.servlet.ServletContextListener` pro vytvoření sdílení coonection poolu

Základním princip vytvoření sdílení connection poolu pro všechny uživatele webu je podobný předchozímu způsobu s tím, že controller servlet aplikace není nutný. Využije se metod posluchače typu **`javax.servlet.ServletContextListener`**. Celý proces lze popsat v těchto krocích:

- je nutné vytvořit třídu implementující rozhraní **`javax.servlet.ServletContextListener`** a překrýt metody tohoto rozhraní: **`void contextInicialized(ServletContextEvent evt)`** a **`void contextDestroyed(ServletContextEvent evt)`**,
- tuto třídu je nutno zaregistrovat jako posluchače události typu **`javax.servlet.ServletContextEvent`** v popisovači webové aplikace web.xml.
- v metodě void **`contextInicialized(ServletContextEvent evt)`** je nutno implementovat vytvoření instance connection poolu pomocí jeho konstruktoru a tu následně uložit jako atribut v rozsahu application.

Při zavírání webové aplikace lze zase využít toho, že je volána metoda **`void contextDestroyed(ServletContextEvent evt).`**

Implementace connection poolu jako singletonu

Použití connection poolu jako singletonu je asi nejjednodušší možností, jak sdílet connection pool mezi všemi uživateli webové aplikace, nevyžaduje ukládání žádných atributů. Využívá principu návrhového vzoru singleton.

6 Zpracování persistentních objektů.

V tuto chvíli je ve webové aplikaci zajištěno připojení k objektově orientované databázi. Pomocí tohoto připojení je již možné začít pracovat s persistentními objekty. Implementace kódu pro práci s persistentními objekty se bude lišit podle použité technologie.

Zpracování persistentních objektů při použití technologie Java Servlets

HTTP požadavky v Java Servlets vyřizují převážně metody **`void doGet(HttpServletRequest req, HttpServletResponse res), void doPost(HttpServletRequest req, HttpServletResponse res)`**. Tyto metody budou vyžadovat připojení k objektové databázi. To jim poskytne connection pool, odkaz na nějž musíme v servletu uložit. Pro získání a uložení odkazu na connection pool je nejlépe použít metodu **`void init()`** servletu. Způsob získání connection poolu záleží na tom, zda je uložený jako atribut v rozsahu application nebo zda jej používáme jako singleton.

V samotných metodách `void` **`doGet(HttpServletRequest req, HttpServletResponse res)`** a **`void doPost(HttpServletRequest req, HttpServletResponse res)`** a jiných je pak možno využít uložený connection pool k získání připojení k objektově orientované databázi a metod tohoto připojení k práci s perzistentními objekty.

Základní kostra kódu provádějícího operace ukládání, změny a odstraňování perzistentních objektů může vypadat takto:

```
ObjectDatabase db = null;
try {
    //získáme připojení k databázi
    db = pool.getObjectDatabase();
    //zahájíme transakci
    db.beginTransaction();
    //uložení objektu
    db.insert(object);
    //změna objektu
    db.update(object);
    //odstranění objektu
    db.delete(object);
    //potvrzení úspěšné transakce
    db.commit();
} catch (ObjectDatabaseException ex) {
    //v případě výjimky transakci odrolujeme zpět
    try {
        db.rollback();
    } catch (ObjectDatabaseException e) {}
    //reakce na výjimku
} finally {
    //uzavřeme připojení k databázi
    try {
        if(db!=null)
            db.close();
    } catch (ObjectDatabaseException ex) {}
}
```

Jak lze z výpisu kódu vidět, veškerá práce s databází se odehrává pouze na úrovni programovacího jazyka Java, není zde nutná znalost žádného dalšího jazyka, např. SQL.

Zpracování persistentních objektů při použití technologie JavaServer Pages

Pokud by tvůrci webové aplikace používali JSP skriptovací značky, jako např. deklarace nebo scriplety, pak by mohli při práci s databází postupovat podle postupu uvedeného v předchozí části, jen s ohledem na technologii JSP. Cílem tvůrců webových aplikací ale u stránek JSP bývá veškerý kód programovacího jazyka Java přesunout mimo JSP stránky. Za tímto účelem mají vývojáři k dispozici několik technologií. Zaprvé je to na platformě Java EE knihovna značek JavaServer Pages Standard Tag Library (JSTL), která zapouzdřuje nejčastější operace prováděné na JSP stránkách. Další možností je použití jiné externí knihovny značek, případně si vytvořit nové značky pomocí technologie Custom Tags.

Pro práci s relační databází je v knihovně JSTL skupina značek s prefixem SQL, které umožňují hlavně získávat výsledné sady záznamů na JSP stránce, vykonávat SQL příkazy typu INSERT, UPDATE, DELETE a seskupovat tyto příkazy do jednoduchých transakcí. Tyto značky lze s úspěchem využit v případě, že vytváříme jednoduchou webovou aplikaci, která se bude skládat pouze z JSP stránek bez použití controller servletu. V opačném případě je samozřejmě

lepší, když je tato funkčnost zapouzdřena právě do tohoto řídícího servletu. Pro použití s databázemi objektově orientovanými je ale tato skupina značek nepoužitelná.

Proto je nutné pro tento účel vytvořit novou knihovnu značek, která bude obsahovat funkčnost skupiny značek s prefixem SQL knihovny JSTL, ale se zaměřením na objektově orientované databáze, avšak zůstane kompatibilní s ostatními značkami knihovny JSTL. Základní skupinu těchto značek by měly tvořit tyto značky:

<jodbc:setObjectDatabase> - značka, která v rozsahu page uloží odkaz na objekt typu ObjectDatabase. Má jeden atribut:

- **poolName** – název atributu, pod kterým je uložen connection pool. Povinný atribut.

<jodbc:query> – značka, která vrátí seznam objektů jako výsledek dotazu na databázi. Má několik atributů vyplývajících z dotazovacího jazyka JODBCQL:

- **type** – datový typ třídy, jehož seznam objektů chceme získat. Povinný atribut.
- **fieldname** – název atributu, u kterého chceme zadávat kritérium výběru. Nepovinný atribut.
- **constrain** – kritérium výběru. Nepovinný atribut.
- **var** – název proměnné, do které se uloží výsledný seznam objektů. Povinný atribut.

<jodbc:transaction> – značka umožňující řízení transakcí. Nemá žádné atributy.

<jodbc:insert> – značka umožňující uložit nový objekt do databáze. Má jeden atribut:

- **object** – object, který chceme uložit do databáze. Povinný atribut.
- **<jodbc:update>** – značka umožňující změnit existující objekt v databázi. Má jeden atribut:
- **object** – object, který chceme změnit v databázi. Povinný atribut.
- **<jodbc:delete>** – značka umožňující odstranit objekt z databáze. Má jeden atribut:
- **object** – object, který chceme odstranit z databáze. Povinný atribut.
- **<jodbc:oid>** – značka, která vrátí jedinečný identifikátor objektu jako java.lang.String. Má jeden atribut:
- **object** – object, jehož OID jako java.lang.String chceme získat. Povinný atribut.

Zpracování persistentních objektů při použití technologie JavaServer Faces

Při použití objektově orientované databáze s frameworkem JSF je nutno vyřešit ještě další dva problémy, které nebyly řešeny v předchozích kapitolách. Jsou to:

- **problém získání odkazu na objektově orientovanou databázi v backing beans:**

Problém získání odkazu na objektově orientovanou databázi v backing beanech se prakticky týká jediné věci a to, jak v metodách backing beans získat odkaz na connection pool, pokud byl uložen jako atribut v rozsahu application. Řešení je následující:

```
javax.faces.context.FacesContext faces =

javax.faces.context.FacesContext.getCurrentInstance();
javax.servlet.ServletContext context = (javax.servlet.ServletContext)
                              faces.getExternalContext().getContext();
ObjectDatabasePool pool = (ObjectDatabasePool) context.getAttribute("pool");
```

- **problém použití objektově orientované databáze v JSF komponentách:**

Tento problém se týká komponent, které zobrazují tabulková data, typicky tedy například komponenta **`<h:dataTable>`**. Jedná se vlastně jen o změnu zvyku, jaký používat datový typ hodnot u atributu value těchto komponent. Vývojáři používající relační databáze tradičně používají datový typ **`java.sql.ResultSet`** nebo **`javax.servlet.jsp.jstl.sql.Result`**, ale naprosto stejně lze využívat i obyčejná pole nebo seznamy typu **`java.util.List`**, které jsou zase přirozené pro objektově orientované databáze.

7 Závěr

Jak lze z nastíněného návrhu metodického postupu vidět, použití objektově orientovaných databází při tvorbě webových aplikací je velmi jednoduché a intuitivní. Hlavní výhodou použití objektové databáze při tvorbě webových aplikací je přímé použití persistentních objektů bez nutnosti jejich mapování do tabulek relačních databází. Navíc v případě použití embedded objektové databáze se objektově orientovaná databáze stává přirozenou součástí aplikace.

Naopak obrovskou nevýhodou je problematická záměna objektových databázových systému kvůli jejich minimální standardizaci na implementační úrovni a minimální podpoře ze strany aplikačního serveru.

8 Literatura

[1] Ball, J., Carson, D., Evans, I., Fordin, S., Haase, K., Jendrock, E. Java EE 5 Tutorial. Sun Microsystems, 2006, on-line: http://java.sun.com/javaee/5/docs/tutorial/doc/

[2] Caché, on-line: http://www.intersystems.com/

[3] Cattell, R.G.G., Barry, D.K. The Object Data Standard: ODMG 3.0. SanDiego: Academic Press, 2000, 280 s. ISBN 1-55860-647-4

[4] db4objects. db4o tutorial. db4objects, 2006, on-line: http://www.db4o.com

[5] Geary, D., Horstmann, C. Core JavaServer Faces. New York: Prentice Hall PTR, 2004, 658 s. ISBN 0-13-146305-5

[6] Hall, M.. Java: Servlety a stránky JSP. Praha: Neocortex spol. s.r.o., 2001, 586 s. ISBN 80-86330-06-0

[7] Kirsten, W., Ihringer, M., Kuhn, M., Rohrig, B. Caché – databáze postrelačního typu a tvorba aplikací. Brno: CP Books, 2005, 391 s. ISBN 80-251-0491-5

[8] ObjectDB. ObjectDB Developer's Guide. ObjectDB, 2004, on-line: http://www.objectdb.com/

[9] Roos, R. M. Java Data Objects. London: Addison-Wesley, 2003, 244 s. ISBN 0-321-12380-8

[10] Shannon, B. Java Platform, Enterprise Edition (Java EE) Specification, v5. Sun Microsystems, 2006. on-line: http://jcp.org/en/jsr/detail?id=244

[11] Wolf, P. *Úspěšný podnik na globálním trhu*. CS Profi – Public, 2006, 240 s. ISBN 80-969546-5-2

[12] Fiala, J., Ministr, J. Průvodce analýzou a modelováním procesů. VŠB-Technická univerzita Ostrava, 2003, 110 s. ISBN 20-248-0500-6

Metodický rámec tvorby multimediálních aplikací

Petr Rozehnal

VŠB-TUO, Ekonomická fakulta
Katedra aplikované informatiky
Sokolská třída 33, 701 21 Ostrava
petr.rozehnal@vsb.cz

Abstrakt
Příspěvek obsahuje důvody tvorby metodického rámce tvorby multimediálních aplikací určených pro výukové účely a jeho popis. Stručně jsou nastíněny odlišnosti nového přístupu a možnosti jeho praktického nasazení. Základem je popis celého vývojového procesu tvorby multimediální aplikace. Zvláštní pozornost je věnována komunikaci mezi autorem obsahu a vývojářem. Ústředním prvkem je tvorba modelu jakožto výchozího místa pro návrh budoucí aplikace.

Abstract
This paper presents reasons of creation of methodological framework of the production of multimedia applications for education and its description. Differences of new approach and possibilities of using are sketched there. The base of framework is a description of development process. A special attention is dedicated to communication between author of content and software developer. The creation of model is a central point for suggestion of future application.

Klíčová slova
Multimediální aplikace, návrh a vývoj softwaru, metodika.

Keywords
Multimedia application, Design and development of software, Methodology.

1 Úvod

E-learning a s ním spojené aktivity jsou v posledních letech tématem aktuálním a široce diskutovaným. A to z mnoha úhlů a pohledů. V tomto příspěvku se věnuji této problematice v souvislosti s tvorbou studijních, výukových materiálů. Nejen pro zmiňovaný e-learning, ale také pro obecné potřeby výuky.

V dalším se budu věnovat způsobu, jak vytvořit výukovou aplikaci obsahující multimediální prvky. Protože jsou tyto aplikace vyvíjeny již řadu let, dovolte nejprve ohlédnutí za existujícími přístupy.

2 Multimediální aplikace

Tvorba multimediálních aplikací není nic nového. Prvotní pokusy o vědecké uchopení pocházejí již z 90. let. Dříve byl v této souvislosti užíván spíše pojem hypermediální aplikace. Existuje tedy několik přístupů, ve kterých se autoři věnují tvorbě softwaru tohoto typu. Jako příklady uvedu:

- **HDM** - A Model Based Approach to Hypertext Application Design, autoři Garzotto Franca, Paolini Paolo, Schwabe Daniel [1].
- **OOHDM** – The Object - Oriented Hypermedia Design Method, autoři Schwabe Daniel, Rossi Gustavo [2].
- **RMM –** Relationship Management Methodology, Autoři Isakowitz Tomás, Stohr Edward A., Balasubramanian P. [3].

Tyto přístupy zdůrazňovaly důležitost modelování budoucího výstupu. Modelování pomocí stávajících notací však mělo své omezení. Modely tak musely být doplněny, neboť aplikace tohoto typu jsou postaveny na bohaté vnitřní (v rámci kooperaci s externími aplikacemi také vnější) struktuře a typický rysem je také nelineární průchod aplikací. Důsledkem absence možnosti zachytit navigaci v aplikacích byl vznik tzv. navigačních modelů, které měly zachytit právě problematiku provázání jednotlivých stavů aplikace. Model je tak výchozím bodem pro pozdější implementaci v konkrétní technologii.

Později, v souvislosti s rozšířením Internetu, roste potřeba tvorby webových prezentací (ať už jsou nazývány jako internetové stránky, HTML stránky, později webové aplikace či jinak). Obě větve silně konvergují, neboť typické rysy multimediálních aplikací - multimediální prvky a interaktivita jsou na Internetu naprosto běžné. Z metodického hlediska došlo k úpravě existujících přístupů a ke vzniku řady nových. Uvedu např.:

- **eW3DT –** Extended World Wide Web Design Technique. Autor Arno Scharl [4].
- **WSDM –** Web Site Design Method. Autoři O. De Troyer, C. J. Leune [5].
- **WebML –** Web Modelling Language, kolektiv autorů univerzity Politecnico di Milano, [6].

K úsilí zachytit a formalizovat postup činností při tvorbě webových aplikací přispěl v českém prostředí např. Martin Molhanec [7]. V rámci tvorby výukových materiálů stojí za zmínku Metodika tvorby platformově nezávislých eLearningových učebních podpor kolektivu Mendelovy zemědělské a lesnické univerzity v Brně [8]. Tento přístup je postaven na automatizaci při generování výstupu.

Obecně je evidentní snaha o efektivní vývoj aplikací obsahujících multimediální prvky, určených pro prostředí Internetu. Patrné je to zejména u rozsáhlejších aplikací, kde se vyskytuje složitá datová i navigační struktura. To s sebou nese potřebu zachycení obsahu, struktury a chování budoucí aplikace (tvorbu modelu) a zachycení procesního postupu vývoje. Považuji za

důležité zmínit také aspekt komunikace mezi vývojáři a zadavatelem. Ad hoc přístup k tvorbě aplikace většinou vede k problémům a nedorozuměním (jak v obsahové části, tak v části procesní).

Vrátím se k multimediálním výukovým materiálům. Uváděné metodiky by nepochybně byly aplikovatelné pro jejich vývoj. Domnívám se ale, že existují určité odlišnosti, které mohou ovlivnit proces vývoje. V rámci svého metodického rámce se snažím tyto odlišnosti akceptovat a nabídnout tak variantu k uváděným přístupům. Rozdíly spatřuji zejména v:

- V postavení klíčových osob procesu vývoje, zejména autora obsahu.
- U výukových aplikací je velká pravděpodobnost začlěnění do systému výuky a do chodu organizace jako celku.

ad 1) Autor obsahu je ústřední postavou vývoje. Autor obsahu je odborníkem na danou problematiku (v prostředí školství často také pedagog), který má vlastní vizi výsledku. Jeho znalosti a představy o obsahu a finální podobě díla jsou velmi konkrétní. Jde tedy o vyřešení komunikace mezi autorem a vývojářem, který potřebuje tuto vizi znát, tak aby ji mohl realizovat po technické stránce. Přitom je zapotřebí akceptovat, že tyto představy nevycházejí z ICT znalosti autora. Autora nezajímá datová struktura a technologické řešení. Zná pouze obsahovou a funkční stránku aplikace. Tomu je potřeba přizpůsobit prostředky modelování budoucího výstupu.

V komunikaci se proto musí vycházet z této úrovně znalostí autora. Je věcí vývojáře, aby posléze převedl tyto uživatelské požadavky do podoby potřebné pro implementaci v dané technologii.

ad 2) Výuková aplikace nebude viset ve vzduchoprázdnu. V procesu vývoje by proto měly být akceptovány širší znalosti o začlenění do chodu organizace. Proto by metodický rámec měl pokrývat celý vývoj aplikace, včetně analýzy aplikačního prostředí a organizace v instituci.

3 Metodický rámec tvorby multimediálních aplikací

Cíle, které by měl metodický rámec určený pro tvorbu výukových materiálů splňovat, jsou následující:

- Uplatnění na aplikace malého a středního rozsahu určené pro výukové účely.
- Vymezení fází postupu tvorby multimediálních aplikací tak, aby na sebe jednotlivé činnosti navazovaly v logickém sledu a byla zajištěna efektivita celého projektu. Ukončení fází představuje hlavní milníky tvorby. V rámci těchto bodů budou definovány další aktivity a konstrukty navrhovaného metodického rámce.
- Stanovení konečného výstupu musí být jasné a mělo by vycházet z představ a možností zadavatele. Nereálné cíle, které nerespektují výchozí podmínky a možnosti, vedou k neúspěchu. Je proto potřeba důkladně zvážit faktory ovlivňující průběh tvorby a konečný cíl

specifikovat tak, aby na jedné straně byly splněny představy o výsledku a na straně druhé byl cíl realizovatelný.

- Návrh prostředků pro popis modelu, jež bude reflektovat představu tvůrce obsahu v takovém stavu, aby byla jasná a srozumitelná pro tvůrce softwarové realizace - programátora (pokud se budou lišit). Zároveň by měla být součástí dokumentace programu. Model by měl také částečně zlepšit komunikaci mezi tvůrcem obsahu a programátorem. Důraz bude přitom kladen na jednoduchost popisných instrumentů a principů popisu.
- Předpokladem je interdisciplinární řešení problému, na němž se podílí řada odborníků. Týmová práce je jedním ze základních faktorů nutných pro úspěšné dokončení díla. Aby takový tým úspěšně fungoval, musí být definována pravidla komunikace, vytýčeny odpovědnost a kompetence členů týmu.

Na základě těchto cílů jsem navrhnul metodický rámec, který je chápán jako základní struktura, doplnitelná podle potřeby konkrétního projektu tvorby. Metodický rámec je tvořen čtyřmi hlavními fázemi. Jsou to:

- analýza,
- návrh,
- implementace,
- evaluace.

Jednotlivé fáze jsou dále rozpracovány na činnosti resp. konkrétní postup prací. Nutno poznamenat, že jde o rámec, nikoliv o rigorozní metodiku. Předpokladem úspěchu je schopnost přizpůsobení, proto je zcela na místě doplnění či změna.

Fáze analýzy

Stěžejní v této fázi je stanovení cílů, vymezení výchozí situace a ustanovení řešitelského týmu. Chybně stanovené cíle mohou znehodnotit výsledek nebo být příčinou celkového neúspěchu. Součástí této úvodní fáze je také stanovení kritérií pro kontrolu výstupů. Přehled fáze analýza je uveden v tabulce níže.

Fáze	Činnosti fáze	Postup jednotlivých činnosti
ANALÝZA		
	Identifikace potřeb	
		Definice současného stavu
		Definice očekávaného stavu
		Stanovení cíle
		Volba vedoucího projektu
	Vymezení výchozí situace	
		Mm aplikace

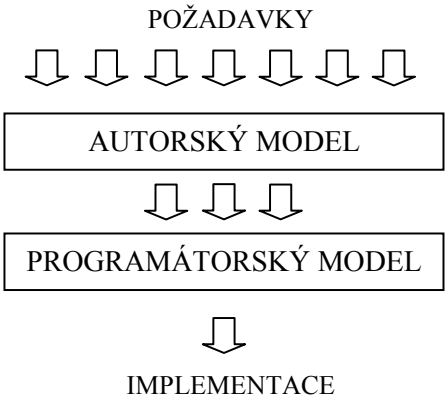
		Čas
		Lidské zdroje
		Vybavení ICT
		Finanční zabezpečení
		Analýza klientů
	Definitivní stanovení cílů	
		Stanovení variant a priorit řešení
		Výběr konečného rozhodnutí
		Finální rozpracování vybrané varianty a stanovených cílů řešení
	Stanovení kritérií pro kontrolu	
	Tvorba realizačního týmu	

Tab.1: Činnosti a kroky fáze analýzy

Fáze návrhu

Hlavním cílem fáze návrhu je sestavení modelu aplikace pro pozdější technologickou implementaci. Model bude doplněn o vypracování pravidel procesní stránky vývoje (např. komunikace členů týmu, ukládání mezivýsledků, jmenné konvence, metadata apod.) Ve fázi návrhu reaguji na nedostatky existujících přístupů (především rozhraní komunikace mezi vybranými členy týmu, viz výše) a snažím se o zavedení mezivrstvy, která usnadní komunikaci mezi členy vývojového týmu. Vycházím přitom z předpokládaných možností autora obsahu a jeho znalostech budoucí aplikace. Úspěch v praktickém nasazení bude podle mých zkušeností závislý na softwarové podpoře při vývoji autorského modelu.

Touto mezivrstvou je zavedení tzv. *autorského modelu* aplikace. Pro tento model, který bude sestavovat primárně autor obsahové části aplikace, bude užito jednoduché notace a bude tolerován vágnější popis vybraných prvků. Tyto rysy jsou podřízeny možnostem většiny autorů, kteří většinou nedisponují znalostmi modelování z oblasti ICT.



Obr.1: Souvislost autorského a programátorského modelu

Na autorském modelu pak bude postaven tzv. *programátorský model* aplikace. Ten již bude sestaven primárně vývojářem na základě autorského modelu a dalších potřebných doplnění. Použitá notace bude již záležet na vývojáři. Tabulka níže uvádí bližší přehled činností fáze návrhu.

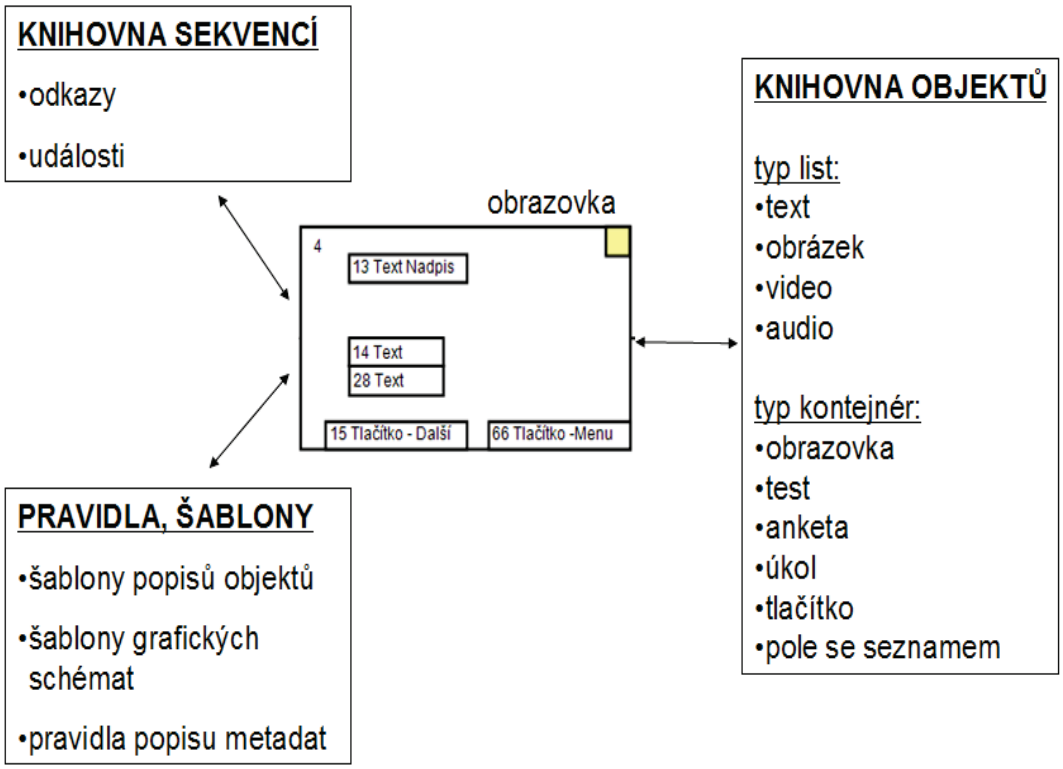
Fáze	**Činnosti fáze**	**Postup jednotlivých činnosti**
NÁVRH		
	Analýza relevantních podkladů	
	Vypracování pravidel tvorby	
	Tvorba autorského modelu aplikace	
		Definice knihovny objektů
		Tvorba diagramů objektů
		Tvorba knihovny sekvencí
	Volba technologie pro realizaci mm aplikace	
	Tvorba šablon	
	Tvorba programátorského modelu aplikace	

Tab.2: Činnosti a kroky fáze návrhu

Autorský model je sestaven ze tří částí: *knihovny objektů*, *diagramu objektů* a *knihovny sekvencí*. Tyto doplňují *šablony*, a *pravidla* uplatňovaná v rámci vývoje. Na obrázcích níže je uvedeno schéma autorského modelu. Popis aplikace stojí na identifikování jednotlivých objektů (komponent), ze kterých je aplikace tvořena. Jejich uložištěm je *knihovna objektů*. Tyto komponenty jsou vyčleněny jako samostatné objekty, které mají své vlastnosti. Tyto vlastnosti tvoří atributy jimiž jsou popsány. Komponenta je jednoznačně identifikovatelná. Podrobnost popisu komponent, stejně jako míra abstrakce při jejich popisu je potřeba stanovit s ohledem na potřebu konkrétní aplikace. V širší návaznosti na softwarem podpořený vývoj je možné komponenty spravovat v rámci repository. Práce s komponentami (tedy reálně uchpitelnými objekty) lépe odpovídá schopnostem a potřebám autorů obsahu. Jde o první definování budoucí aplikace z pohledu obsahového a funkčního. Nejde o definování datové struktury a konkrétní, technologické řešení.

Ústřední komponentou je *obrazovka*, která představuje konkrétní, statický stav aplikace. Mezi další komponenty pak mohou patřit *texty*, *obrázky*, *audio*, *animace* atd. Zařadit však můžeme také komponenty složitější, které samy nesou určitou funkčnost. Tyto mohou být již připraveny k začlenění do aplikace nebo budou definovány pouze abstraktně s předpokladem realizace v případě konkrétního nasazení (na podobných principech je založen např. produkt Authorware,

který užívá připravené Knowledge Object). Mezi tyto komponenty mohou patřit *testy*, *ankety* atd.



Obr.2: Schéma autorského modelu 1

Dále je popsána struktura aplikace, tedy vzájemné propojení mezi objekty. V modelu jde o *diagram objektů*. Je rozpracováno složení komponent do vyšších celků, kde na nejvyšší úrovni stojí aplikace jako celek. Hierarchie bude rozvíjena podle jednoduchých principů:

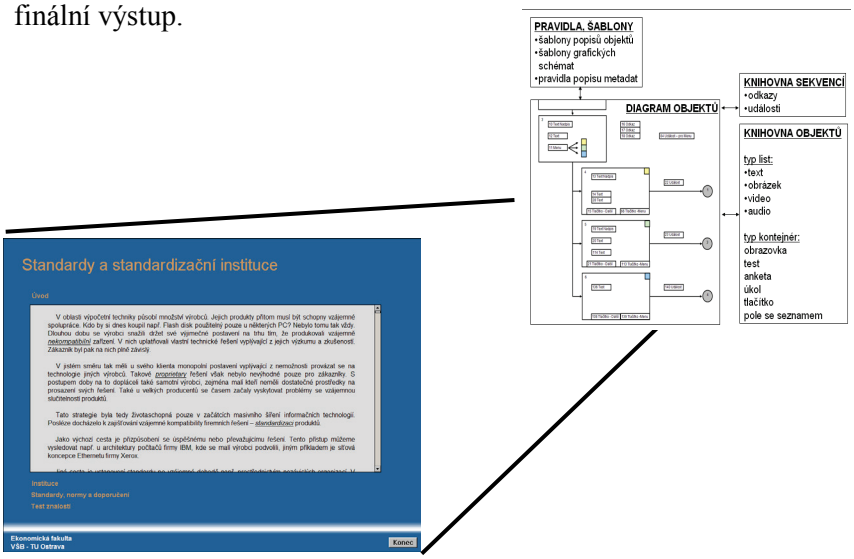
- sekvenčně – v případech, kdy jednotlivé obrazovky budou následovat jedna za druhou v přímé posloupnosti
- pomocí větvení – tam, kde dochází k vytvoření nové sekvence. Jde tedy o vytvoření další úrovně hierarchie.

Chování aplikace popisuje *knihovna sekvencí*, kde jsou uloženy jednotlivé změny ve stavu aplikace. Jde tedy o dynamický popis chování aplikace. Zatímco doposud šlo při modelování aplikace o princip rozdělení do jednotlivých časových úseků, které jsou popsány samostatnými obrazovkami, nyní dochází k dalšímu upřesnění možných změn stavů a průchodů aplikací. Jedná se o:

- reakci na chování uživatele (přechody mezi obrazovkami, linky, spuštění událostí apod.)
- automatické změny stavu v průběhu chodu aplikace.
- Uvedené události vyvolají v rámci aplikace dvě základní reakce:
- přechod mezi částmi aplikace, resp. externím cílem
- změnu stavu aplikace.

Z pohledu autora obsahu je nutné popsat chování popisem počátečního a cílového stavu aplikace. Je zřejmé, že v případě komplikovaného chování nebo členění aplikace je tento způsob popisu nedostatečný. Stále platí, že jde o úvodní popis aplikace v podobě autorského modelu. Snahou je zachování maximální jednoduchosti při definování modelu.

Model doplňují *šablony a pravidla*. Šablony určují prezentační vrstvu aplikace, pravidla definují procesní prvky potřebné v rámci vývoje. Na obrázku níže je znázorněna také vazba na finální výstup.



Obr.3: Schéma autorského modelu 2

Užitá notace je velmi jednoduchá. Dle potřeby může být doplněna či pozměněna. Zde by mělo jít především o dohodu mezi mezi členy vývojového týmu. Důraz je kladen na jednoduchost a použitelnost.

Fáze implementace

Fáze implementace je především zaměřena na programátorskou část prací. Na základě modelu, je aplikace realizována ve vhodně zvolené technologii. Dále jsou prováděny práce na multimediálních součástech aplikace tak, aby mohly být použity v paralelně prováděných programátorských pracích. V rámci metodického rámce není tato fáze primárně řešena, neboť

její realizace může být závislá na zvolené technologii, počtu programátorů, jejich zkušenostech apod.

Fáze	Činnosti fáze	Postup jednotlivých činnosti
IMPLEMENTACE		
	Tvorba prototypu (volitelně)	
	Tvorba mm součástí	
	Programovací část	

Tab.3: Činnosti a kroky fáze implementace

Fáze evaluace

V této fázi dochází k finalizaci výstupu. Jde o testování, hodnocení, konečnou kontrolu a následné úpravy díla dle připomínek recenzentů a výsledků testů nové aplikace. Dochází k celkovému ukončení všech aktivit na projektu.

Výsledná aplikace je předána zadavateli včetně veškeré dokumentace. Níže jsou uvedeny činnosti fáze.

Fáze	Činnosti fáze	Postup jednotlivých činnosti
EVALUACE		
	Testování mm aplikace, hodnocení a recenze díla	
	Finální úprava a tvorba dokumentace	
	Ukončovací práce na procesu vývoje	

Tab.4: Činnosti a kroky fáze evaluace

4 Závěr

Uvedený metodický rámec si klade za cíl reagovat na potřebu tvorby elektronickcýh výukových materiálů s multimediálními prvky. Pokud pomineme automatizaci procesu vývoje takových výstupů [např. 8], stojí autoři mj. před problémy jak organizovat proces vývoje a jak vyřešit komunikaci v rámci vývojového týmu. Na oba problémy se v tomto metodickém rámci snažím odpovědět.

Návrhem odpovídající posloupnosti činností a kroků, včetně složení vývojového týmu reaguji na procesní stránku vývoje. Důraz je přitom kladen na komplexní pokrytí celého procesu vývoje [9, 10]. Zároveň však zdůrazňuji možnost úpravy tohoto návrhu s ohledem na skutečné potřeby

konkrétní implementace. Eliminací chaotického postupu při tvorbě je možné zmenšit riziko neúspěchu projektu. Na druhou stranu to nesmí znamenat přílišné svázání členů vývojového týmu nesmyslnou byrokracií.

V oblasti komunikace mezi členy týmu vidím řešení doplněním o autorský model aplikace. Měl by představovat efektivní nástroj pro formalizaci uživatelských požadavků do podoby, která je primárním a dobře strukturovaným zdrojem informací pro vývojáře aplikace. Metodický rámec předpokládá důkladný popis všech definovaných objektů a jejich zařazení do modelu. Pokud je tato činnost realizována bez podpory CASE nástroje, jedná se u větších aplikací o značně náročnou práci. A to jak z hlediska časového, tak z hlediska technického. Plné nasazení pro tvorbu autorského modelu je proto zamýšleno v souvislosti s využitím nástroje typu CASE (v tuto chvíli chybí). V tom případě by byla usnadněna nejen tvorba autorského modelu, ale i následné manipulace s modelem. V neposlední řadě také dodatečné úpravy ze strany autora např. v nových verzích.

Následné sestavení programátorského modelu je již plně v kompetenci analytiků a návrhářů, kteří pak mohou uplatnit klasické nástroje pro popis modelu aplikace. V oblasti komunikace s autorem obsahu se mohou opřít o již zpracovaný autorský model a také o dodatečné informace autora, který však byl již donucen k důkladnému promyšlení budoucího výstupu.

Tvorba autorského modelu tak představuje vložení mezivrstvy, jejímž úkolem je zmírnit rozdíl v přístupu autora obsahové části aplikace (většinou však laika v oblasti ICT) a realizátorem implementace.

5 Literatura

[1] Garzotto, F.; Paolini, P.; Schwabe, D. HDM – A Model-Based Approach to Hypertext Application Design. *ACM Transactions on Information Systems* [online]. 1993, vol. 11, no. 1, January 1993. Dostupný na WWW: <http://doi.acm.org/10.1145/151480.151483> ISSN: 1046-8188.

[2] Schwabe, D.; Rossi, G. An Object Oriented Approach to Web-Based Application Design. *Theory and Practice of Object Systems,* 1998, 4(4), New York: Wiley and Sons. Dostupný na WWW: <http://www-di.inf.puc-rio.br/schwabe//papers/TAPOSRevised.pdf> ISSN 1074-3224.

[3] Isakowitz, T.; Stohr, E. A.; Balasubramanian, P. Rmm: A Methodology for Structured Hypermedia Design. *Communications of the ACM, 1995,* vol. 38 , no. 8, August 1995, p. 34 – 44. Dostupný na WWW: <http://doi.acm.org/10.1145/208344.208346> ISSN:0001-0782.

[4] Scharl, A. A Conceptual, User-Centric Approach to Modeling Web Information Systems. In *AusWeb99*, *the Fifth Australian World Wide Web Conference,*1999. Lismore, Australia. Dostupný na WWW: <http://ausweb.scu.edu.au/aw99/papers/scharl/>.

[5] *WSDM* [online]. Oficiální webové stránky projektu WSDM. Dostupný na WWW: http://wsdm.vub.ac.be/Research/overview.php

[6] *WebML Papers* [online]. Oficiální webové stránky projektu WebML. Dostupný na WWW: <http://www.webml.org/webml/page93.do?UserCtxParam=0&GroupCtxParam=0&ctx1=EN >.

[7] molhanec, M. *Tvorba webových sídel jako inženýrský úkol*. In Tvorba softwaru 2001, Ostrava, 2001.

[8] *Metodika tvorby platformově nezávislých eLearningových učebních podpor* [online]. Brno: Mendelova zemědělská a lesnická univerzita. Dostupný na WWW: <http://mesua.mendelu.cz/metodika/Metodika.pdf>.

[9] Wolf, P. *Úspěšný podnik na globálním trhu*. CS Profi – Public, 2006, 240 s. ISBN 80-969546-5-2

[10] Fiala, J., Ministr, J. Průvodce analýzou a modelováním procesů. VŠB-Technická univerzita Ostrava, 2003, 110 s. ISBN 20-248-0500-6

Standardizace modelu pro environmentální data, informace a znalosti

Karel Kisza

Masarykova univerzita, Institut biostatistiky a analýz,
Divize environmentální informatiky a modelování
Jana Uhra 10, 602 00 Brno
kisza@iba.muni.cz

Abstrakt
Příspěvek popisuje hlavní důvody, které poukazují na nutnost standardizace modelů pro environmentální data, informace a znalosti. Dále jsou nastíněny základní principy a struktura takového modelu a technologie, které jsou vhodné pro jeho implementaci.

Abstract
This paper presents the main reasons which show the necessity of standardization of models for environmental data, information and knowledge. Then follow the basic structure and main principles of such model and technologies suitable for its implementation. .

Klíčová slova
Data, Informace, Znalosti, Standardizace modelu.

Keywords
Data, Information, Knowledge, Model standardization.

1 Úvod

Pokud bychom chtěli vysvětlit, co je to informatika a čím se zabývá, můžeme v literatuře najít velké množství definic. Oxford English Dictionary z roku 1976 definuje informatiku jako „vědeckou disciplinu, která zkoumá strukturu a vlastnosti vědecké informace“ (Collen in Hogarth, 1997), Random Hause Webster´s Dictionary jako „studium informačních procesů“. Podle Všeobecné encyklopedie Diderot z roku 1999 je informatika „teorie informace - obor zabývající se zákonitostmi vzniku, sběru, přenosu, třídění, zpracování a užití informací“. Dalo by se tedy obecně říct, že informatika je věda, jejímž hlavním předmětem jsou data, informace a pravidla, kterými se řídí vztahy mezi nimi. Ale zde narážíme na další problém – co jsou data a co jsou informace.

V současné době existuje mnoho definic pojmů „informace“ a „data“. V praxi se pak často můžeme setkat s tím, že jsou tyto pojmy různě kombinovány, slučovány nebo také zaměňovány. Základním mezinárodním standardem definujících základní pojmy pro informační komunikační

technologie (ICT - Information communication technologies) je norma ISO/IEC 2382-1:1993, která byla v roce 1998 vydána Českým normalizačním ústavem jako česká norma ČSN ISO/IEC 2382-1:1998.

Pojem „**data**“ je zde definován následovně:

- opakovaně interpretovatelná formalizovaná podoba informace vhodná pro komunikaci, vyhodnocování nebo zpracování.

a pojem „**informace**“ jako:

- poznatek (znalost) týkající se jakýchkoliv objektů, např. faktů, událostí, věcí, procesů, myšlenek nebo pojmů, které mají v daném kontextu specifický význam

Dalším pojmem velmi často používaným v souvislosti s daty a informacemi, který bývá také definován různě „znalost“. Bývá v běžném hovoru často používán jako synonymum pro pojem informace. Proto je vhodné uvést také jeho definici:

„**Znalost**“ je schopnost člověka nebo jakéhokoli jiného inteligentního systému uchovávat, komunikovat a zpracovávat informace do systematicky a hierarchicky uspořádaných znalostních struktur. Znalost je charakterizována schopností abstrakce a generalizace dat a informací. Znalost je rovněž to, co jednotlivec vlastní (ví) po osvojení dat a informací a po jejich začlenění do souvislostí, tj. znalost je získána zobecněním nalezené informace [7].

Jedním z cílů sedmého rámcového programu rozvoje vědy a výzkumu EU (FP7) je vytvoření „Jednotného evropského informačního prostoru pro životní prostředí“ (SEISE – Single European Information Space for Environment), ve kterém instituce pracující s environmentálními informacemi (definováno dále), poskytovatelé služeb a občané mohou spolupracovat a využívat všechny dostupné informace, které potřebují bez jakýchkoliv technických omezení.

Naplnění tohoto cíle FP7 zahrnuje jak management dat a informací, tak i znalostí.

Vytvoření kompletního konečného modelu celého SEISE, který by pokrýval veškerá environmentální data, informace a znalosti, je v podstatě nemožné. Neustále narůstají objemy existujících dat, informací a znalostí. Pro reprezentaci znalostí SEISE je nejprve nutné standardizovat obecný model environmentálních dat, informací a znalostí.

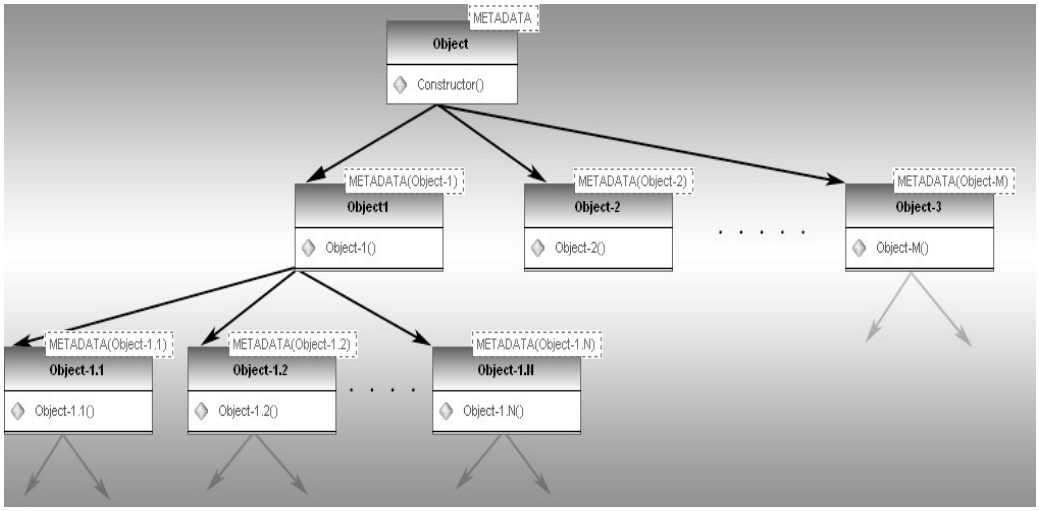
2 Idea modelu pro standardizaci modelu pro environmentální data, informace a znalosti [1]

Konceptuální model znalosti v SEISE může být znázorněn jako síť „objektů“. Všechny tyto objekty patří do některé ze čtyř bázových množin:

- *Atributy*
- *Objekty*
- *Metody*

- *Informace*

Každá bázová množina je reprezentována stromovou strukturou, viz následující obrázek:



Každý prvek bázové množiny je pouze třída - abstraktní typ definující množinu objektu stejných vlastností a teprve jednotlivá instance těchto tříd obsahují reálná data.

SEISE je v podstatě množina prvků bázových množin a množina relací (vztahů) mezi jejich prvky. Každá (instance) objektu obsahuje kromě vlastní hodnoty (hodnot) také metadatový popis, ze kterého lze odvodit například důvěryhodnost dat.

Každý prvek má v prostoru svůj daný účel při vzniku požadované informace (což představuje vlastní znalost).

- Atributy jsou základní stavební jednotky Objektů
- Objekty jsou předávány jako vstupní parametry metodám
- Výsledky metody (metod) jsou prezentovány (seskládány) jako informace

3 Formalizace modelu

Model

SEISE definujeme jako pětici: S = [A, O, M, I, R], kde A, O, M, I jsou bázové množiny - stromy a R je relace (množina relací) popisující vztahy mezi souvisejícími prvky bázových množin (relace dědičnost, rodič, potomek, předchůdce, následník).

- Strom = souvislý acyklický orientovaný graf – kořenový strom
- Vrchol v je označen jako **kořen** tohoto kořenového stromu. Kořen v je vrchol, do kterého nevchází žádná hrana, pouze z něho vychází alespoň jedna hrana.

- Kořen je prvek, který označuje základní (nejobecnější) datový typ, od něhož jsou odvozeny všechny další prvky ve stromu – relace **dědičnosti**
- Pro každý vrchol w označíme vrchol p ležící na hraně (p,w) vstupující do vrcholu w jako **rodiče** vrcholu w (**každý vrchol různý od kořene má jediného rodiče**). Podobně k vrcholu w označíme q **potomka** vrcholu w, pokud leží na hraně (w,q). Každý vrchol kromě koncových vrcholů má alespoň jednoho potomka (může mít i více potomků).
- Platí: jestliže p je předchůdce q a zároveň q je předchůdce r, pak platí p je předchůdce r .
- Strom se základním typem kořene T je buď prázdná struktura nebo prvek typu T, na který je připojen konečný počet disjunktních stromových struktur se základním typem T (označujeme je jako **podstromy**).

Propojení bázových množin

- pro každý prvek $i \in I$ existuje množina M' a relace r tak, platí: $M' \subseteq M$ a $r(i, M')$ je platná;
- pro každý prvek $m \in M$ existuje množina O' a relace r tak, platí: $O' \subseteq O$ a $r(m, O')$ je platná;
- pro každý prvek $o \in O$ existuje množina A' a relace r tak, platí: $A' \subseteq A$ a $r(o, A')$ je platná.

Reprezentace informace (znalost)

- Informace = souvislý acyklický orientovaný graf – **čtyřpatrový** kořenový strom
- Vrchol v je označen jako **kořen** tohoto kořenového stromu. Kořen v je vrchol, do kterého nevchází žádná hrana, pouze z něho vychází alespoň jedna hrana. Kořen **patří do množiny I**.
- Pro účely tohoto stromu už nebude použita relace dědičnosti, ale nová relace přiřazení (zobrazeni) R - "**je založen na**":
- *R: Mn -> I, nebo On -> M, nebo An -> O*
- Kořenem každé relace je právě jeden objekt z množiny I (tj. první patro stromu). Jeho následníci (tj. druhé patro stromu) je tvořeno objekty z množiny M. Následníci těchto prvků z množiny M (tj. třetí patro stromu) jsou prvky z množiny O. Následníci těchto prvků z množiny O (tj. třetí patro stromu) jsou prvky z množiny A.
- Pro každý vrchol w označíme vrchol p ležící na hraně (p,w) vstupující do vrcholu w jako **předchůdce** vrcholu w. Podobně k vrcholu w označíme q **následníka** vrcholu w, pokud leží na hraně (w,q). Každý vrchol kromě koncových vrcholů má alespoň jednoho následníka (může mít i více následníků).
- Platí:
    a. Jestliže *w* patří do *A*, pak všichni jeho následníci patří do *O*
    b. Jestliže *w* patří do *O*, pak všichni jeho následníci patří do *M*
    c. Jestliže *w* patří do *M*, pak všichni jeho následníci patří do *I*

d. Jestliže *w* je následníkem *u*, pak pro žádného potomka u není *w* jeho následníkem (tj. *w* není následníkem žádného prvku patřícího do podstromu, jehož kořenem je *u*)

4 Idea implementace

Sémantické webové služby

Prvky množin jsou "objekty" na různém stupni implementace. Implementované prvky jsou stejné (konstantní) pro všechny uživatele (matematické definice, základní data, …). Neimplementované části jsou reprezentovány jako interfaces, které je nutné implementovat podle konkrétních možností uživatele (implementuje sám uživatel – poskytovatel dat – sémantické webové služby), bez nutnosti znalosti celé "domain knowledge".

Jak implementované "objekty", tak interfacy reprezentují "domain knowledge". Jsou propojeny pomocí ontologie, která představuje "operational knowledge".

Informace = strom, který obsahuje implementované části a neimplementované části. Neimplementované části jsou definovány jako obecné interfacy se vstupními a výstupními omezeními. Každý takový prvek může být implementován jako služba. Typ této služby se odvíjí od typu prvku (bázové množiny):

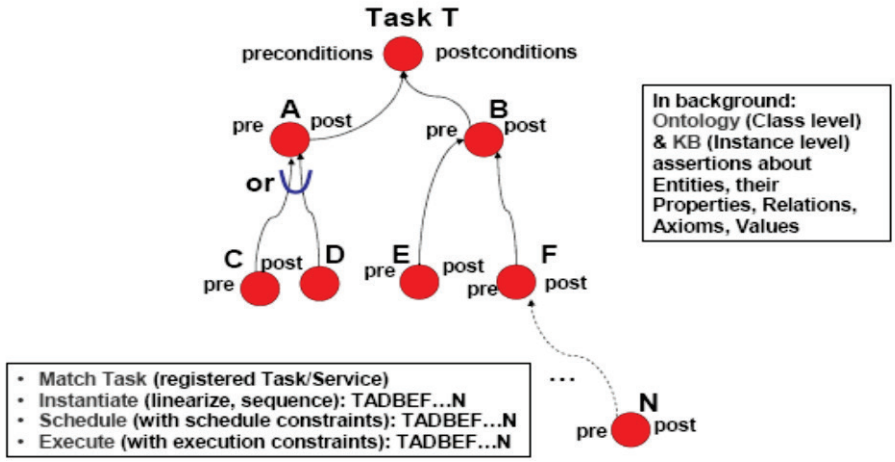
- *M* – metody – služba umožňující získání informace z agregovaných dat (prvků množiny *O*)
- *O* – Objekty – služba umožňující přístup k datům
- *I* – Information – služba umožňující agregaci základních informací – výstupů metod – pro získání informace ve formě požadované uživatelem.

OWL-S

Každá informace může být prezentována jako proces T, která má určitý plán, který musí být proveden pro dosažení požadovaného výsledku (informace). T je komplexní proces, který může obsahovat podprocesy, které musí být naplánovány a provedeny (prvky množin M, O, A). Každý z těchto podprocesů má svá vstupní a výstupní omezení (vlastnosti).

OWL-S se zejména zaměřuje na pohled reaktivní plánování sémantiky webových služeb, což znamená, že služba a její komplexní kompozice jsou obecně pojaty jako tří fázová operace:

- planning,
- scheduling,
- execution.



Tento model reprezentace informace je analogický pro model informace definovaný v EIS

5 Závěr

Základem pro vybudování SEISE je funkční interakce mezi rozličnými zdroji digitálních dat, informací a znalostí, tj. zajištění tzv. sémantické interoperability. Sémantická interoperabilita je obsahové vyjádření struktury metadat, které dovoluje sémanticky kombinovat datové prvky z různých schémat, slovníků a jiných nástrojů a umožňuje tak vyhledávat informace napříč heterogenními distribuovanými datovými zdroji. Pomocí sémantické interoperability jsou řešeny např. případy, kdy jednotlivé zdroje používají různé termíny pro popis téhož pojmu nebo naopak používají stejné termíny pro různé pojmy.

V dnešní době jsou technologie podporující sémantickou interoperabilitu velmi populární a využívané (sémantické technologie), zejména pak ontologie, které oproti například UML (Unified Modeling Language) poskytují některé další výhody. Velmi rychle vzniká velké množství různých ontologických modelů jak pro systémy v různých vědních oborech (doménové ontologie), tak i pro systémy v jednotlivých podnicích (aplikační ontologie), … Aby byla zajištěna celková sémantická interoperabilita systémů, je nutné vyřešit problém vzájemné kompatibility těchto modelů - standardizovat model pro environmentální data, informace a znalosti.

6 Literatura

[1] Hřebíček, Jiří - Ráček, Jaroslav - Kisza, Karel - Štefaník, Matěj. Environmental Information Space II. In Proceedings of the 3rd Internationl Summer School on Computational Biology. 1. vyd. Brno : Masaryk University, 2007. od s. 158-170, 13 s. ISBN 978-80-210-4370-1.

[2] Štefaník, Matěj - Kisza, Karel. Komunikační rámec pro výměnu environmentálních informací. In 3. letní škola aplikované informatiky. Brno : Masarykova univerzita, 2006. od s. 106-115, 10 s. ISBN 80-210-4146-3.

[3] Jiří Hřebíček, Jaroslav Ráček, Systémy integrovaného managementu - Elektronický učební text předmětu PA088.

[4] Ressler M. a kol: Informační věda a knihovnictví: Výkladový slovník české terminologie z oblasti informační vědy a knihovnictví. Vydavatelství VŠCHT Praha. 2006.

[5] Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: Proceedings of the 2nd International Conference on Formal Ontology in Information Systems, Chris Welty and Barry Smith, eds, Ogunquit, Maine, October 17-19, 2001

[6] P., Andersen, J., Schärfe, H.: What Has Happened to Ontology. Conceptual Structures: Common Semantics for Sharing Knowledge 2005, 425-438.

[7] Jiří Hřebíček, Jan Žižka, Vědecké výpočty v biologii a biomedicíně, Masarykova univerzita, 2007

Service Oriented Approach for Accessible Publishing of Environmental Information in Web Environment

Miroslav Kubásek

Institute of Biostatistics, Masaryk University,
Brno, Kamenice 128/3,
625 00 Brno, Czech Republic,
kubasek@iba.muni.cz

Abstract

With the increasing growth in popularity of Service-Oriented Architectures based on Web Services we discuss in this article the possibility to use this approach in the area of publishing environmental information by web-portal. System for Publishing of Environmental Information (SPEI) must reflect the fact that many kind of actors are involved in environmental activities. The actors can be international institutes, public administration, businesses, academicians, citizens, governmental, and nongovernmental organizations (often without any deeper information and communication technology knowledge and also with different technology used). Under these conditions any developed information system has to support communication and collaboration with existing information systems and other software systems. It must be opened for new actors and the use of new software systems. All these requirements can be realized if SPEI is service oriented and have specific service-oriented architecture. We will discuss several crucial points of service-oriented architecture, its different application in developing environmental information systems. Next we will present the Environmental Web-portal (EnviWeb) as a unique complex and open SPEI web-portal on the Czech Internet (www.enviweb.cz). In this article we will also present the functionality background of this portal based on web services.

Abstrakt

S rostoucí růst popularity architektury orientované na služby založené na webových službách budeme diskutovat v tomto článku možnost používat tento přístup v oblasti zveřejňování informací o životním prostředí podle internetového portálu. Systém pro zveřejňování informací o životním prostředí (SPEI) musí odrážet skutečnost, že mnoho druhů aktérů jsou zapojeny do činností v oblasti životního prostředí. Tyto subjekty mohou být mezinárodní instituce, veřejná správa, podniky, akademici, občané, vládní a nevládní organizace (často bez hlubší informační a komunikační technologie znalosti a také s různými použité technologie). Za těchto podmínek se jakékoliv vyvinutý informační systém pro podporu komunikace a spolupráce se stávajícími informačními systémy a jinými softwarovými systémy. Musí být otevřen pro nové účastníky a využívání nových softwarových systémů. Všechny tyto požadavky mohou být realizovány v případě, SPEI je orientované na služby a mají konkrétní architektury orientované na služby.

Budeme diskutovat o několik klíčových bodů servisně orientované architektury, jeho různé aplikace ve vývoji informačních systémů v oblasti životního prostředí. Dále budeme prezentovat na životní webový portál (EnviWeb) jako unikátní komplexní a otevřený SPEI internetového portálu na českém internetu (www.enviweb.cz). V tomto článku se budeme také prezentovat funkčnost pozadí tohoto portálu na bázi webových služeb.

Keywords
Environmental information, Knowledge, SOA, SPEI.

Klíčová slova
Environmentální informace, Znalosti, SOA, SPEI.

1 Introduction

Environmental information and their management (environmental data collection and presentation, legislative, etc.) involve a broad group of actors of very different types. Actors can be international institutes, public administration bodies, businesses, academicians, citizens (often without any deeper information and communication technology knowledge) and nongovernmental organizations. The actors must communicate, collaborate and provide their data to all interested (sometimes also authorized) parties. The actors can require responses from other actors, can take part in various (business) processes based on workflow management tools or specifications based on networks of activities. Almost any SPEI must be able to support communication with existing information systems and other software systems. These SPEI must be opened for public and new actors[9]. It should further support environmental management (e.g. decision making with respect to environment protections).

The actors can be subjects of very different properties. They can be based on e-Government in Environment or they have no explicit organizational structure. The actors are as a rule autonomous. Many actors and other interested subjects have their own information communication or control systems. They use specific data structures able to support various administration functions, not only the environmental ones. The SPEI must be able to work independently.

We show that all these requirements can be realized if used systems are service oriented and have Service Oriented Architecture (SOA). SPEI has SOA if it has structure formed by a virtual network of asynchronously communicating software components and encapsulated real world bodies called (software) services. The services behave like the real world services in mass service systems. SOA simplifies the implementation of user-oriented interfaces of services.

2 SPEI and Service Oriented Architecture

[9] Directive 2003/4/EC on public access to information on the environment

System for publishing of environmental information must be able to communicate with almost independent or autonomous large software components. The corresponding software systems must usually have a structure peer-to-peer network of large components (i.e. the system has a coarse granularity and from the point of view of system architecture all components are equal) communicating asynchronously via messages of appropriate structure or via data-stores. The messages are transported by a middleware (CACM 1995, Král/Žemlička 2002b).

With the introduction of the World Wide Web, a new era for software architectures is in progress. The Internet initially began as a way to publish scientific papers and evolved into dynamic HTML (Hypertext Markup Language) and eventually to XML (Extensible Markup Language), which is a meta-language and a fundamental, standard, and enabling technology for data exchange between different platforms.

XML describes the data in an application-independent manner, and Web Services use this technology to enable the sharing of distributed processes between heterogeneous computing environments. Today's application architecture involves creating independent services accessible through the firewall and support one discrete function. Clients interact with services, which are assembled to build the application infrastructure.

Service Oriented Architecture (SOA) is a particular type of software architecture which has distinguished features and characteristics. The concept of SOA emerged in the early 1980s and become a significant architectural style especially after invention of Web Services.

2.1 Definition of Service Oriented Architecture

Service Oriented Architecture (SOA) is an architectural style which utilizes methods and technologies that provides for enterprises to dynamically connect and communicate software applications between different business partners and platforms by offering generic and reliable services that can be used as application building blocks. In this way it is possible to develop richer and more advanced applications and information systems.

Although SOA is not a new concept, especially after the invention of Web Services, the new developments in this area bring about a new way of constructing software application architectures, a new approach to rebuild available software infrastructures and possibility of communicating with other enterprises according to the available services.

However, building SOA is still challenging for the following reasons:

- The way SOA approaches to software resources is different from traditional architectures,
- SOA needs a level of architectural regulation,
- SOA implementation needs an environment capable of being accessed by different enterprises,

- Definition and composition of services into new ones in a secured and managed environment is another aspect that needs a particular concentration.

SOA is an architectural style that defines an interaction model between three main functional units, in which the consumer of the service interacts with the service provider to find out a service that matches its requirements through searching registry. A meta-model describing this interaction is shown in Fig.1.

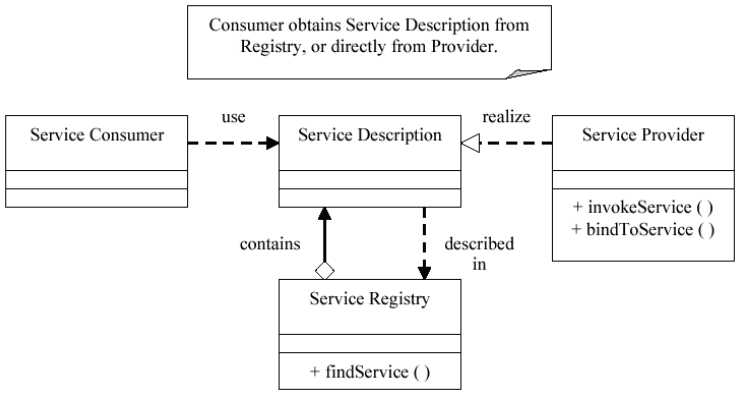


Fig. 1. Service Oriented Architecture Conceptual Model

SOA contains 6 entities in its conceptual model, described as follows (McGovern, J. et all 2003)

Service Consumer: It is the entity in SOA that looks for a service to execute a required function. The consumer can be an application, another service, or some other type of software module that needs the service. The location of the service is discovered ether by looking up the registry, or if it is known, the consumer may directly interact with the service provider.

- *Service Provider*: It is the network addressable entity that accepts and executes requests from consumers. It provides the definite service description and the implementation of the service. The service provider can be a component, or other type of software system that fulfills the service consumer's requirements.
- *Service Registry*: It is a directory, which can be accessible through network and contains available services. Its main function is to store and publish service descriptions from providers and deliver these descriptions to interested service consumers.
- *Service Contract*: A service contract is the description that clarifies the way of interaction between the service consumer and provider. It contains information about

request-response message format, the conditions in which the service should be executed, and quality aspects of the service.

- *Service Proxy*: It assists the interaction between service provider and service consumer by providing an API written in the local language of the consumer. The service provider and a convenience entity supply it for the consumer. As well as it can enhance performance and provides caching facilities. Service proxy is an optional entity of SOA.
- *Service Lease*: It specifies the amount of time that a service contract is valid. It is manager by registry and determines the executive well-defined timeframes of binding to the services. Usage of service lease supports loose coupling between service provider and consumer and maintenance of state information for the service.

SOA reflects specific principles and characteristics that need to be applied when building Service Oriented Application Infrastructures, which are described as follows:

- *Services are discoverable and dynamically bound*: Services need to be discoverable dynamically at run-time. The consumer of the service finds the required service through searching of registry and gets all the information necessary to execute the service. There is no compile-time dependency to bind to the service, apart from the service contract that the registry provides.
- *Services are self contained and modular*: A service supports unified and functional interfaces aggregated to perform specific and concrete business logic. These interfaces are related to each other in the context of a module and contain sufficient information to be authentic without any dependency to other software modules or applications.
- *Services are interoperable*: Services express the ability to communicate with each other without any platform and language dependencies. Because each software module might have proprietary and tightly coupled structures, service based architecture utilizes interoperable technologies that support the protocols and data formats of the service's current and potential consumers.
- *Services are loosely coupled*: Coupling describes the level of dependencies between software modules. Loosely coupled modules are flexible and have well-defined dependencies, on the contrary, tight-coupled software systems are difficult to configure because of unknown requirements of modules within the software structure. Service oriented architecture stress the development of loosely coupled services as the software construction unit. Loosely coupled is achieved by usage of service registries. The service requires no other system specific information to be executed autonomously. However, tight coupling cannot be avoided at the interface definition level or binding to a specific protocol.

- *Services have a network-addressable interface*: A service should publish its interface on the network to conform service oriented architecture design principles, so that the consumer is able to invoke the service in a distributed manner over the network. In this way it is probable to use the service at any time with different consumers. A service can also be accessed through local interface without usage of network. However, one of the main aims in building SOA is to allow the consuming of services in a location independent manner.
- *Services have coarse-grained interfaces*: The concept of granularity is related with the way of implementation of interfaces within software system. If the interface supports all the functions necessary for complete business logic, it is a coarse-grained interface. In contrast, if the interface implements only a part of certain functionality, it is considered fine-grained (Fig. 2). Granularity can be applied to the entire service implementation, and also to the individual methods of the interface execution. A service oriented software system supports coarse-grained interface design by nature with different granularity levels. The objects, which build the service, may still be fine-grained, however these objects are kept inside the physical structure of the service.

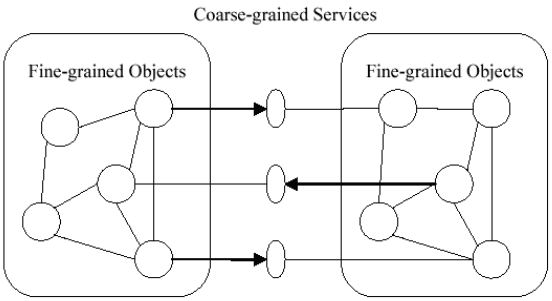


Fig. 2. Granularity

- *Services are location transparent*: SOA promotes location transparency, which means the consumer of the service does not have pre-knowledge about the position of the service until to execute it at run-time through registry. The only dependency between consumer and provider is the service contract, which can shift from one location to another one without affecting consumer.
- *Services can be composed into new applications*: Another key characteristic of SOA is to enable building new applications composed from existing services. Composition is an effective design which mainly focuses on service modularity and reuse of service functionality without having pre-knowledge of which applications will use the service in the future.

- *SOA supports self-healing*: Self-healing is described as the ability of a system to recover itself in case an error occurs during its execution. Service oriented systems should give more importance to support self-healing than other architecture styles as the services are combined to execute a business function at run-time. The consumer should have the ability to find different services on the network that supports the same functionality for the aim of a proper and regular execution of the software system.

2.2. SOA Layered Architecture

Currently the most frequently used application development model is based on three-tier architectural structure (Fig. 3), which supports an additional layer between client and data storage tiers. The additional layer, called as business logic layer, provides code isolation from client and sharing of the application logic between various client implementations. It is a competent approach to software development for flexible managing of data and usage of system resources.

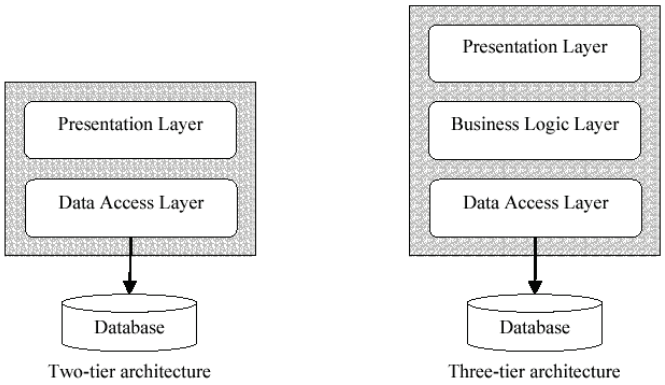


Fig. 3: Two-tier and Three-tier Architectural Models

SOA is based on n-tier application development in which services are layered on top of components that are responsible for providing certain functionalities and maintaining quality of service requirements for services (Arsanjani 2005), as shown in Fig. 4.

Each layer in SOA has specific architectural characteristics, as described below:

- **Operational Systems Layer**: This layer contains existing applications including CRM and ERP packaged applications, object-oriented systems, and business intelligence applications. These applications provide the background for

services and each of them has its own proprietary structures, databases and other system resource access.

- **Enterprise Components Layer**: These are specialized components to provide certain functions and requirements for services. They are the business assets for service implementations, and other system necessities such as management, availability and load balancing of services.
- **Services Layer**: This layer contains the actual services which can be discoverable and invoked by other applications to provide a specific business function for enterprises. Services realize and deploy enterprise component interfaces as service descriptions and are published on the network.
- **Business Process Composition Layer**: The services can be composed into a single application through service orchestration or choreography, which supports specific use cases and business processes.
- **Presentation Layer**: It provides user interfaces for services and composite applications. Although presentation layer is not a straight concern for SOA, because of the objective of using services as user interface bring about the development of standards such as portlets and Web Services for Remote Portlet (WSRP) specifications.

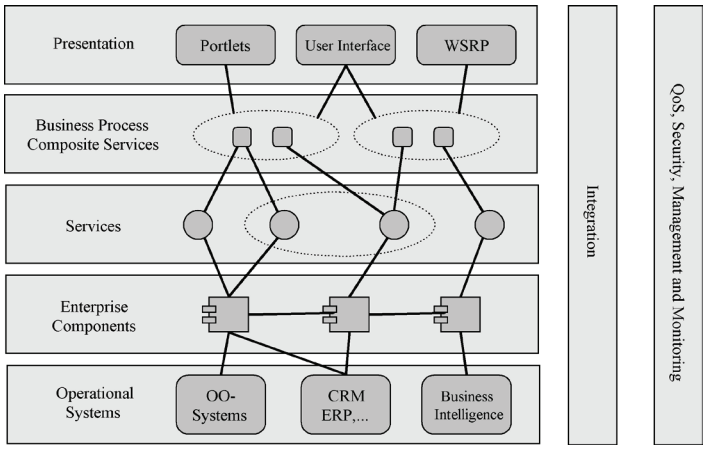


Fig. 4: The Layers of Service Oriented Architecture

Other concern for SOA is the integration of services and composite applications within the enterprise by supporting the features such as reliability, proper routing and coordination of services, and managing other technical details including protocol and integrating party agreements. QoS (quality of service) requirements, security,

management and monitoring of services are also other requirements that need to be clarified when designing service based application architectures.

2.3 Service Oriented Software Systems

Service-oriented software systems (SOSS) usually have the structure of virtual peer-to-peer networks of autonomous software components (usually complete applications). SOA is advantageous for business information systems and decentralized organizations. The services are permanently active processes and the systems behave like real world mass service systems. SOA provides no inherent explicit tools for the definition and control of business processes. SOA enables new ways of implementation of (environmental) business processes in SOSS effectively using the ICT enabled by service orientation. Such an implementation reduces the use of centralized services, if any. The basic idea of the implementation is the generation and use of newly generated services having specific roles and structure.

3 Environmental service-based web portal

The EnviWeb web portal is primarily intended for experts, professional public, firms, government and citizens. The main purpose of the EnviWeb is to provide public environmental information and to enable free access to environmental information. For foreign users, who do not understand the Czech language, there is English version of the EnviWeb. There is also a possibility of accessing the EnviWeb by hand-held vendors (e.g. PalmPC).

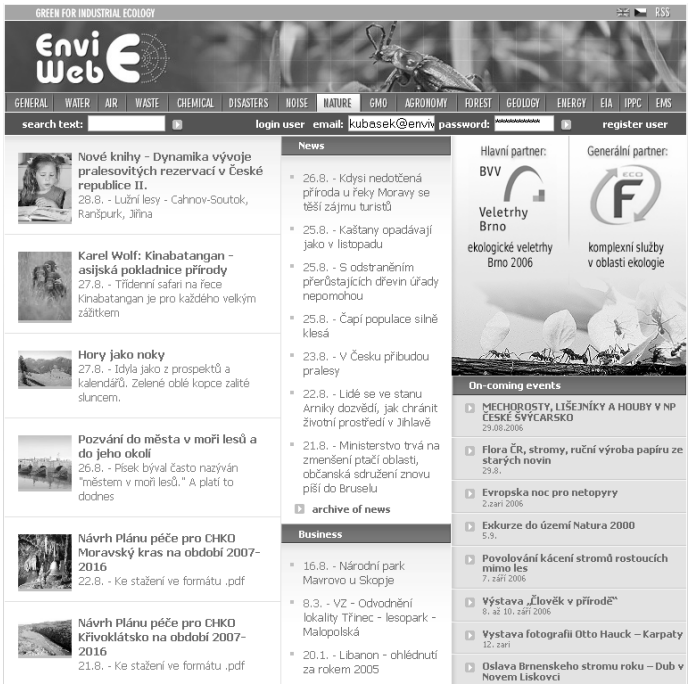


Fig. 5: Web portal EnviWeb

The EnviWeb portal contains a large amount of varied environmental information, which are collected and verified by the group of environmental experts. Information are divided into several portal sections to cover particular elements of environment. The EnviWeb portal is composed of the following sections (see Figure 5): Water (water management and its aspects), Air (air protection problems, monitoring of pollution sources and their reduction), Waste (waste management, demeaning pro-duction of waste and waste recycling), Chemicals (environmental aspects of chemi-cal treatment and their influence on the living environment), Disasters (prevention of serious industrial accidents, reactions to accidents and removing their conse-quences), Noise (noise problems, noise measurement, noise reduction, modeling and abnormal noise prevention), Nature (problems concerning nature protection), Soil (soil conservation and ecological agronomy), Forest (forest management and forest protection), Geology (environmental aspects of geologic prospecting, extraction and rehabilitation), EIA (Environmental Impact Assessment), EMS (Environ-mental Management Systems and Program EMAS of EU), IPPC (Integrated Pre-vention and Pollution Control), Misc. (this section contains articles, links, firms etc., which can't be placed into the specific sections or are universal in scope). The EnviWeb is like a crossroads to environment

sources in the Czech Republic. For this reason a huge catalogue of links to other special web sites is maintained.

The portal EnviWeb was initiated and developed as a complex environmental web-portal of the Czech Internet. It is dedicated to provide comprehensive and up-to-date environmental information and news to public. After two years it turned into a daily environmental information source for hundreds of visitors who are professionals in the environment, enterprise environ-mentalists, managers of companies providing services in the branch and producing or selling products and technology for environmental protection and labor safety. The EnviWeb helps students of high school or university who are interested in the environmental protection, informs representatives and members of professional unions, employees of public administration, organizers and participants in special events, editors and readers of professional publications. It acts as an information broker for environmental information in the Czech Republic. EnviWeb is approximately daily visited by 1500 users and they retrieve daily about 9000 pages.

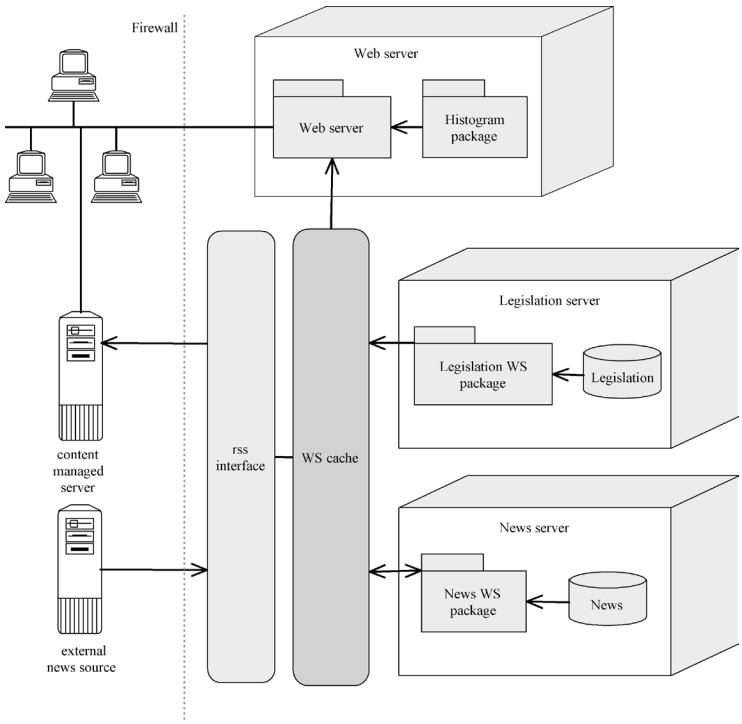


Fig. 6: Schema of EnviWeb

The portal is divided to several independent servers. They are web server, server with legislation and news server (see Figure 6). All servers run operation system Linux and as data management they use MySQL7 database engine. All servers comunicate through SOAP messaging. In the middle of this communication is semantic-based Web Services cache to improve responses from Web Services. Web Services are implemented in PHP by NuSOAP library and on the side of web server are used standard web server technologies as PHP, XHTML, CSS and JavaScript.

4 References

[1] Andrews, T. (et al) (2003) Specification: Business Process Execution Language for Web Services Version 1.1. http://www-106.ibm.com/developerworks/library/ws-bpel/.

[2] Arsanjani, A. (2005) How to Identify, Specify and Realize Services for your SOA (Part II and III), IBM, 2005,

[3] CACM (1995) The Promise and the Cost of Object Technology. A five years Forecast. Communication of ACM, Nr. 38.

[4] Král, J., Žemlička, M. (2002a) Autonomous components. In: Hamza, M. H. (Ed.) Applied Informatics, ACTA Press, Anaheim. pp. 125-130.

[5] Král, J., Žemlička, M. (2002b) Necessity, Challenges, and Promisses of Peer-to-Peer Architecture of Information Systems. In: Harindranath, G. (et al) (Eds.): New perspectives on Information Systems Development: Theory, Methods and Practice, pp. 125-134. Kluwer Academic, New York.

[6] Král, J., Žemlička, M. (2003) Software Confederations – An Architecture for Global Systems and Global Management. In: Kamel, S. (Ed.) Managing Globally with Information Technology. Idea Group Publishing, Hershey.

[7] McGovern, J. et all. (2003) , Java Web Services Architecture, Morgan Kaufmann Publishers

[8] W3 Consortium (1999b) XSL transformations (XSLT). http://www.w3c.org/TR/xslt.

[9] W3 Consortium (2001a) Web service definition language. http://www.w3.org/TR/wsdl.

[10] W3 Consortium (2001b) Semantic web. http://www.w3.org/2001/sw/.

[11] W3 Consortium (2003) Simple object access protocol. http://www.w3.org/TR/SOAP.

Hromadné zpracování neurčitých dat

Michal Hejč

Masarykova univerzita, Fakulta informatiky
Botanická 68a, 602 00 Brno
3148@mail.muni.cz

Abstrakt
Kvalita dat (a neurčitosti v datech) je oblastí, která v současné době výrazně vystupuje do popředí prakticky ve všech oblastech lidské činnosti, kde dochází k hromadnému zpracování dat. Současně ale chybí její pevnější teoretické zázemí. Výzkum probíhá roztříštěně, přičemž nejvíce úsilí je vyvinuto zejména v praktických oborech a aplikovaných vědních oborech. Dosažené výsledky si často vzájemně konkurují, terminologie je nekompatibilní. Příspěvek mapuje situaci, zabývá se dále motivačním příkladem a vytyčuje další směr výzkumu.

Abstract
In the last time, data quality (and the data uncertainty) is becoming much more important in almost all branches of human activity where mass data treatment is involved. At the same time there is the lack of theoretical background for the problem. Research is fragmented. The best advances can be observed in the field of applied science and in the practice. The results of research are not complementary, often they compete. The paper maps the situation, presents motivating example and states the future research direction.

Klíčová slova
Data, informace, neurčitost, Internet.

Keywords
Data, Information, Uncertainties, Internet.

1 Úvod

Typické úkoly v informatice vypadají tak, že máme nějaký vstup a požadujeme určitý výstup. Na vstupu je podmnožina množiny objektů, se kterými informatika dokáže pracovat. Tato množina objektů obsahuje data, informace, formalizovaná zadání, atd. (podrobnější definici důležitých pojmů uvedeme později). Konkrétní typy vstupních objektů obsažené ve vstupní množině závisí na příslušném informatickém odvětví. Například běžný webový formulář zpracovává informace, operační systém má na vstupu data, matematický software může mít na vstupu informace, data nebo formalizované zadání, atd. Výstupem jsou opět zmíněné objekty.

Způsob řešení je v informatice většinou koncipován tak, aby mohl proběhnout pokud možno bez lidského zásahu. Informatika se tak pokouší o určitou náhradu lidí. Ultimativním úkolem informatiky je pak vytvoření umělé inteligence převyšující ve všech směrech a oborech inteligenci lidskou.

Tato záměrná paralela informatiky s principy, na kterých funguje lidské myšlení, je v některých důsledcích pro informatiku paradoxní. Lidské myšlení není dokonalé. Pracuje s odhady, neurčitými formulacemi, předpokládá akceptování řešení, které nemusí být úplně správné, atd.

Zde se nám tedy přirozeně odštěpují dva proudy informatiky: jeden počítá s exaktní přesností a nemá žádnou z nevýhod lidského myšlení (říkejme mu „ideální systémy“), druhý je připraven akceptovat určitou možnost omylů, vlastní lidem, a díky tomu získává oproti prvnímu proudu některé výhody a nevýhody (říkejme mu „kompromisní systémy“).

Jednou z výhod kompromisních systémů je možnost práce s neurčitostmi ve vstupních (resp. výstupních) objektech. Podrobnější definice příslušných pojmů jsou uvedeny v dalším textu. Pokud bychom chtěli využívat pouze „spolehlivé“ vstupy a výstupy (to znamená, že věrně odráží skutečnost, kterou mají popisovat), nikam bychom se nedostali – takové prakticky neexistují. Vždy je tady určitá možnost chyby, kdy mohou přejít do stavu, který realitě neodpovídá a tím se stávají takzvaně neurčitými. Může se jednat například o chybu technickou (ke které dojde při elektronickém přenosu dat nebo na měřícím zařízení), o chybu způsobenou lidským faktorem (špatné zadání čísla na klávesnici) nebo o nějakou jinou další chybu, možností je mnoho (viz další text).

Ideální systémy mají nastaven práh, za kterým považují vstupní (resp. výstupní) objekty za spolehlivé, přestože tento předpoklad ve většině případů explicitně nikde neuvádí (například účetní software nepočítá s tím, že by účetní špatně zadala cenu z přijaté faktury). Kompromisní systémy mohou oproti tomu využít i objekty, které spolehlivé zcela zřejmě nejsou.

Tato výhoda se nemusí zdát nijak závratná, protože vstupní a výstupní objekty je ve většině případů možno považovat za spolehlivé a je možno využívat na jejich zpracování ideální systémy. To je dáno ale tím, že nespolehlivé vstupní a výstupní objekty se zatím nevyužívají v plné míře. Tato disertační práce by měla vytvořením vhodných teoretických nástrojů a postupů přispět k většímu využití a rozšíření kompromisních systémů, využívajících nespolehlivé vstupy a výstupy, zatížené neurčitostmi.

2 Shrnutí problematiky a utřídění dosavadních znalostí

Problematika neurčitostí v datech a informacích je velmi různorodá a zabývá se jí příliš mnoho vědních a praktických odvětví, než aby bylo možno předpokládat konzistentní přístup k chápání základních pojmů napříč všemi obory.

Situaci neulehčuje ani český jazyk a obecné trendy, které se v běžné lidské mluvě projevují. Pokud se některé termíny dostatečně dlouho a frekventovaně nesprávně využívají (například díky politikům, veřejným sdělovacím médiím, atd.), stane se jejich nesprávné využití

normou a tím se posouvá jejich původní význam. Tento problém je zřejmý také v případě častého sloučení významu pojmů data a informace.

Roztříštěnost problematiky je v některých případech doplněna zatajováním postupů, kterými jsou dosahovány výsledky. To se týká zejména praktických oborů, kde jsou postupy nezřídka předmětem obchodního tajemství.

Utřídění zjištěných dosavadních znalostí a nalezení jednotící linie pro všechny vědní a praktické obory, zabývající se neurčitostmi v datech a informacích, pak bude závěrem celého shrnutí problematiky.

Analýzy problémů se dříve často prováděly nad malým množstvím dat, s velmi omezenými znalostmi, intuitivně, na základě zkušeností nebo jiným ne zcela spolehlivým a současně automatizovatelným způsobem.

V současné době došlo v souvislosti s rozvojem ICT k posunu v této oblasti, zejména z hlediska objemu zpracování dat. Bohužel vzrostla možnost chyby v datech. Zpočátku nebyl problém tak významný, protože samotný přechod na využití ICT byl dostatečným motorem pro zlepšování kvality výsledků analýz. Vývoj se ubíral zejména směrem ke zvyšování zpracovávaných objemů a druhů dat. Nyní se ale znovu dostává do popředí problém chyb v datech. Kvalitu analýz již není stávajícími prostředky jak zlepšovat, výrazně větší množství dat už k dispozici nebude a i kdyby bylo, vzorky jsou již natolik reprezentativní, že nárůst kvantity není v mnoha případech nutný.

Dostáváme se tedy k současné situaci, která je motivací pro tuto práci. Je k dispozici velké množství zdrojů dat a informací. Jsou to různé komerční i vládní databáze, encyklopedie, soubory dat a v neposlední řadě obrovský prostor Internetu. Rozvíjí se služby, které práci s informacemi na Internetu organizují (Google).

Stále ale zůstává problém chybovosti veškerých těchto zdrojů. V jednotlivých případech je zkušený analytik schopen jasně rozhodnout, jak s daným údajem nebo informací naložit – tedy jestli mu věřit, nevěřit, věřit částečně, atd. Jak se ale zachovat v případě hromadného strojového zpracování? To je otázka, na kterou bych se chtěl zaměřit při mapování současného stavu řešení problematiky.

Problematika hromadného zpracování neurčitých dat je zatím v rámci vědy a praxe řešena velmi nejednotně. Praxe je zřejmě mnohem dál než věda, protože poptávka po řešení problémů zatím převyšuje možnosti teoretické vědy. Tím vznikají různá prakticky více či méně použitelná řešení, která ovšem často postrádají návaznost na jiný vědecký výzkum. Bude nutno vymezit pozici této problematiky v rámci výzkumu.

Samostatnou kapitolu si zasluhuje podrobný rozbor pojmů, využívaných v této oblasti (jako jsou neurčitost, kvalita dat, důvěryhodnost, atd.). Mnoho těchto pojmů je používáno nejednotně, nesprávně, jejich významy jsou často zaměňovány nebo posouvány. Samotné zmapování a utřídění pojmového aparátu by mělo být jedním z cílů a přínosů této práce.

Dalšími použitelnými znalostmi jsou způsoby reprezentace dat s ohledem na možnosti kvantifikace jejich kvality, míry kvality dat a možnosti zásahu do dat.

a. Základní definice a chápání pojmů

Základem veškeré následující teorie je dobře zmapovaný a případně upravený či nově navržený pojmový aparát. Zvolená oblast je velice citlivá na volbu (a interpretaci) správných odborných termínů. V praxi jsou oproti jiným vědním oblastem pojmy častěji zaměňovány, slučovány nebo jinak špatně využívány a bylo by velkým paradoxem, aby teorie na odstranění chybovosti nebyla sama prosta chyb již ve své pojmové podstatě.

V dané problematice je možno identifikovat čtyři charakteristické základní množiny pojmů, obsahující konkrétní pojmy s příbuzným významem. Množiny je možno charakterizovat jako:

vstupy a výstupy (data, údaje, čísla, informace, ontologie, modely, atd.),

neurčitosti v datech (nepřesnosti, záměrné lži, chybějící údaje, atd.),

kvalita dat (důvěryhodnost, přesnost, reprezentativnost, atd.),

akce nad daty (zásahy, validace, nahrazení, zneplatnění, verifikace, atd.).

Další pojmy (zejména z oblasti měr a algoritmů) jsou již obecně známy a nejsou pro počáteční pochopení problematiky natolik podstatné. Proto budou podrobněji uvedeny až v rámci disertační práce.

i. Vstupy a výstupy

Vstupy pro analýzu a výstupy jsou objekty, nad kterými se bude provádět hodnocení kvality. V našem případě se v textu kromě následujících definic bude jednat většinou o data, ale je možno text zobecnit i na ostatní objekty (informace, znalosti, atd.). Zde je nutno mít na paměti jaké definice jsou pro nás směrodatné. V souvislosti s tím bude nutno ujasnit vztah mezi daty a informacemi a znalostí, chápání pojmů metadata, ontologie, údaj, záznam, atd.

Podrobnější definice těchto pojmů budou dále zkoumány, v této fázi zpracování řešení problému je k dispozici zatím jen obecný předpoklad, jak bude situace pravděpodobně vypadat. Pokud bychom již teď důsledně mapovali definice těchto pojmů, kapacitně bychom překročili rozsah tezí disertační práce, protože co pojem, to minimálně jedna až dvě stránky textu. Pojmy nezávisle existují v tak rozdílných vědních a praktických oborech, jako jsou filozofie, matematika, ekonomika, právo, atd. Podobné zmapování pojmů bude součástí jednoho ze dvou hlavních cílů disertační práce.

Data (množné číslo) – znovu použitelné reprezentace informace formalizovaným způsobem vhodné ke komunikaci, interpretaci a zpracování. Data budou chápána jako soubor jednotlivých údajů (údaj – jednotné číslo) nebo záznamů. Příkladem budiž část databáze hlášení o produkci a nakládání s odpadem jako data a jedna položka (políčko) databáze jako údaj. Databáze je chápána jako formální celek, obsahující všechna spolu související data.

Kódování dat a fyzické uložení v paměti počítače nejsou směrodatné (kromě případu, kdy bychom se snažili vyjádřit možnost chyby, vzniklé díky nedokonalosti fyzického záznamového mechanizmu), proto se jím nebudeme nadále zabývat.

Ontologie bude chápána jako popis významu dat (vazba na jiná data) a metadata vzniknou sloučením ontologie a dat. Informace bude konkrétní závěr (výrok), který jsme schopni z dat (metadat) nějakým způsobem vyvodit. Analýzu dat nejčastěji provádíme právě za účelem vyvození konkrétní informace. Požadovaná informace může být v takovém případě jedním z měřítek kvality dat.

Znalost bude chápána vždy pouze ve vztahu k nějakému objektu – vstupu nebo výstupu nebo jejich množinám (data, ontologie, informace, modely, znalosti). Touto znalostí o daném objektu pak budou veškeré vstupy nebo výstupy, které můžeme v souvislosti s ním využít a také vyjádření vzájemných vztahů.

Důležitým pojmem je i model – jedná se o funkci, která na základě alespoň jednoho vstupu vytvoří hodnotu, která simuluje hodnotu alespoň jednoho vstupního nebo výstupního údaje.

ii. Neurčitosti v datech

Na základě předchozích definic je již zřejmé, co je myšleno pod pojmem data. V těchto datech je pak možno identifikovat určité nedostatky - neurčitosti. Tyto nedostatky je nutno pro naše účely dobře rozlišit – kategorizovat. Určité rozlišení je v této oblasti výzkumu definováno (například normativně), objevují se pojmy jako entropie, neuspořádanost, nekonzistence, nedůvěryhodnost, atd. a jim vždy odpovídající pojmy z hlediska kvality dat – konzistence, důvěryhodnost, atd.).

Dalším úkolem je tedy toto rozlišení kompletně zmapovat, kriticky zhodnotit a přidat (nadefinovat) veškeré chybějící kategorie neurčitostí, případně ubrat nadbytečné.

Kromě toho je zřejmé, že mezi některými vlastnosti je za určitých předpokladů možno očekávat závislost. Jedná se například o důvěryhodnost zdroje dat. Pokud data pochází z maximálně důvěryhodného zdroje, je zřejmé, že i jejich ostatní vlastnosti budou na výši (v opačném případě zdroj není maximálně důvěryhodný). Mapování závislostí mezi vlastnostmi bude jedním z přínosů práce, vytyčeném v prvním cíli.

Pokud neurčitost v datech překročí určitou mez, za kterou již data nepovažujeme podle nějakých předem daných měřítek za akceptovatelná, je možné hovořit o chybných datech. Jednotné číslo je možno u tohoto pojmu definovat jako chybný údaj.

iii. Kvalita dat

Tento pojem přímo souvisí s kategorií chyb v datech. Určitou kategorii kvality dat je možno odvodit od míry, v jaké je daná kategorie chyb v datech zastoupena. Dále je možno kvalitu odvodit od účelu dat (tedy jako kvalitu informace, která vznikne odvozením z dat). Zde se ovšem jedná o rozdílný pojem – kvalita informace. Zmapování vztahu takto chápané kvality dat a informace bude důležitým úkolem dalšího výzkumu. Vztah je dosti složitý, přičemž z velké části bude záviset na kategorii kvality dat a na typu informace.

Jednotné číslo je možno u kvality dat definovat jako kvalita údaje. Kvalita informace je chápána přímo jako jednotné číslo, pokud bychom měli hodnotit kvalitu více informací, hodnotili bychom již například kvalitu studie. Takto vysokou úrovní hodnocení kvality se tedy nebudeme dále zabývat.

Pokud kvalitu dat zatím neznáme, můžeme jednotlivým údajům před započetím dalších akcí nastavit výchozí hodnotu, kterou definujeme jako implicitní kvalitu.

iv. Akce nad daty

Stanovení kvality je operace nad daty, kdy údajům s neznámou kvalitou je do jejich příslušných podpůrných struktur (atributů) přiřazena kvalita na základě zvolené míry.

Aby bylo možno zlepšit kvalitu dat, je nutné v datech provést nějaké změny nebo je nutné kvalitu ověřit a tím ji zvýšit. Přímo s jednotlivými údaji je tedy možno provádět tři základní akce: nahrazení, vyřazení a doplnění. Ověření (není řečeno jakým způsobem, ale může to být například nějaká ruční operace nad daty) pak při nezměněných údajích může zvýšit (ale i potvrdit nebo snížit) jejich kvalitu.

Nahrazení, zneplatnění a doplnění údajů se provede v případě, že provedená akce slibuje zvýšení jejich kvality nebo kvality informace. Ověření by bylo vhodné provádět v ideálním případě vždy, ale protože tato operace je většinou časově a zdrojově náročná, je nutno pro údaje stanovit kritéria, za kterých se ověření bude vyžadovat, a tím snížit množství provedených ověření na únosnou mez.

b. Způsoby reprezentace dat s ohledem na možnosti kvantifikace jejich kvality

Vstupní data jsou reprezentována ve svém vlastním formátu. Tento formát sám o sobě zřídkakdy umožňuje ohodnocení kvality jednotlivých údajů, proto je nutné formát rozšířit o některé další atributy. Nové atributy by měly být schopny pojmout hodnocení všech kategorií chyb.

Další otázkou je, jakou formou se stanovení kvality bude provádět. Nejjednodušším způsobem je ohodnocení každého údaje hodnotou true nebo false. Takový způsob může mít v určitých případech své opodstatnění, ale současně se ztrácí případný potenciál chybného údaje. Lze předpokládat, že údaj je sice chybný, ale přesto je blízký správné hodnotě (pokud bude vyhodnocován například údaj z nepřesného monitorovacího zařízení, víme, že je chybný, ale je možné jej brát jako použitelný v určité toleranci).

Nabízí se tedy další známé možnosti, jak data reprezentovat, přičemž všechny jsou silnější než použití hodnot true, false (které je pouze jejich speciálním případem):

- pomocí intervalu, ve kterém je údaj platný,
- pomocí fuzzy množin,
- pomocí pravděpodobnosti, uváděné u každého údaje,

- pomocí kombinace výše zmíněných přístupů.

Intervalová aritmetika se používá k zachycení chyb, které vznikají buď vzhledem k nepřesnostem v měření nebo vzhledem k existenci několika alternativních postupů k odhadu parametrů [6]. Nevýhodou je, že pokud jsou známy pravděpodobnostní distribuce vstupů, aplikací intervalové aritmetiky přicházíme o tyto informace.

Teorie fuzzy množin je metodou, která pokrývá analýzu nepřesností takových systémů, kde nepřesnosti vznikají spíše díky jisté vágnosti, nepřesnosti a nejasnosti než díky nahodilosti [8]. Teorie je vhodnější pro kvalitativní rozhodování a klasifikaci údajů do fuzzy množin než pro kvantitativní předpoklady. Takto pojatý formální popis není vhodný pro zachycení chyb při nelineárních strukturálních analýzách.

Pravděpodobnostní analýza se používá zejména v případech, kdy je známá pravděpodobnostní distribuce analyzované veličiny [2]. Neurčitost je zde charakterizována jako pravděpodobnosti asociované s událostmi.

Ve chvíli kdy data na vstupu takto reprezentována nejsou (v některých případech mohou být, ale ve většině případů budou data na vstupu bez tohoto rozšíření) bude nutno doplnit hodnoty chybějících atributů. Samotné doplnění těchto hodnot je již určitým zásahem do kvality dat, kdy z implicitní kvality dat (například nulové) se po aplikaci nějaké míry stane kvalita dat.

c. Míry kvality dat

Na základě vhodné reprezentace dat je možno přistoupit k měření některé kategorie jejich kvality aplikací příslušné míry. Míry jsou na reprezentaci přímo závislé, ale jejich závislost by se měla projevovat spíše na typu příslušné chyby. Proto bude nutno dobře zmapovat vztahy mezi reprezentací dat, typem chyby a použitou mírou.

Samotný postup mapování měr na jednu obecnou míru by měl být jedním z hlavních přínosů dalšího výzkumu. Současně se předpokládá vytvoření metodiky použitelné obecně a ilustrované v některých speciálních případech (v případových studiích).

d. Možnosti zásahu do dat

Jak již bylo výše zmíněno, nad daty je možno provádět dva druhy operací: buď se provádí přímo zásah do údaje (doplnění, nahrazení, vyřazení) nebo se zásah provede do podpůrné struktury, dané reprezentací dat (ohodnocení kvality).

Další výzkum by měl pouze metodicky sjednotit tyto akce. Nejedná se o cíl práce. Tyto možnosti budou využity pouze pro účely testování navržené míry kvality dat.

e. Pozice ve výzkumu

Informatika se podobně jako matematika stává spíše vědou, využívanou jako nástroj pro ostatní aplikované vědy nebo přímo pro praxi. Informatický výzkum tak často probíhá zcela mimo mateřský obor, na rozdíl od matematiky se ale jeho výsledky mnohem hůře vrací zpět pro další obecné využití. To je dáno i tím, že informatika existuje oproti matematice mnohem kratší dobu

a množství nových výsledků z oblasti aplikačních věd je při jejich současném velkém rozvoji jen velmi pomalu vstřebáváno.

I z tohoto důvodu se tato práce pokusí o malé přispění ke zvratu v nepříznivém vývoji a návrat části iniciativy na stranu teoretičtěji zaměřeného mateřského oboru. Roztříštěnost výzkumu na podobných věcech v mnoha aplikačních oblastech je nevýhodná z hlediska hospodárného využití zdrojů.

Nyní je nutno zmapovat vědní obory a praktická odvětví, která se zabývají neurčitostmi v datech, případně hromadným zpracováním dat. Jejich kategorizaci můžeme provést třemi způsoby: podle původu dat, podle oboru nebo podle činnosti, která je s daty prováděna.

i. Původ dat

Tato kategorizace je provedena s ohledem na typ neurčitostí, které se mohou převážně objevit – jedná se o data z oblasti fyzikálně-chemického měření, z oblasti společenské a data se zásadním vlivem lidského faktoru při jejich digitalizaci.

Data z oblasti fyzikálně-chemického měření (například monitoring teploty, tlaku a vlhkosti vzduchu) jsou z hlediska neurčitostí a kvality dat pro zadání méně zajímavá, navíc v této oblasti je teorie poměrně dobře zvládnuta.

Data ze společenské oblasti (např. průzkumy veřejného mínění) nebo data se zásadním vlivem lidského faktoru při digitalizaci (například údaje v technickém průkazu dovezeného ojetého vozidla) nebo i obecně data z Internetu a data mining což je pro nás nejzajímavější kategorie, jsou zatíženy zejména neurčitostmi, vzniklými díky vlivu lidského faktoru.

ii. Obor

Nejjednodušší rozdělení do kategorií je podle oborů vědních nebo praktických činností. Takové rozdělení je nejpřirozeněji k dispozici, ale pro účely práce má bohužel nejmenší význam, protože některé na první pohled nesouvisející obory mohou řešit prakticky stejný problém.

Jako příklady je možno uvést environmentální informatiku, astronomii, lékařství, historii, evidenci vozidel, atd. Právě podle této kategorizace budou zmíněny některé výsledky z několika různých oborů a odvětví, přičemž bude zmíněn typický původ dat a výsledky budou porovnány s možnostmi disertační práce.

Zvláštní pozornost bude zejména v disertační práci věnována podoborům informatiky, jako jsou databázové systémy.

iii. Činnost

Posledním typem kategorizace je rozdělení podle činností, které jsou poté s daty prováděny. Takové rozdělení úzce souvisí s kvalitou informací a v přehledu k jednotlivým oborům bude uvedeno pouze okrajově.

Jako příklady je možno uvést evidenci, monitoring, statistiku, encyklopedické zpracování, e-learning, atd. Některé tyto činnosti se víceméně kryjí s obory podle předchozí kategorizace, některé však ne.

iv. Přehled výsledků podle jednotlivých oborů

Samotné zmapování komplikované situace bude samostatným přínosem a ne jen postupem, jak dosáhnout vyššího cíle. Přehled je vzhledem k omezenému rozsahu zúžen zejména na výzkumné oblasti a pracoviště, na které je autorovi k dispozici dobrý kontakt. Kromě toho jsou zmíněny i některé běžně známé praktické výsledky a základní legislativní a normativní předpoklady.

Legislativní a normativní požadavky (požadavky řady norem ISO/IEC 9126 a připravované ISO/IEC 250xx pro jakost systémů a softwaru, dále Zákon č. 365/2000 Sb., o informačních systémech veřejné správy a jeho navazující právní předpisy) nebo jiné normativní požadavky [5].

Informatická praxe – důležité výsledky se začaly objevovat v informatické praxi, bohužel se velmi málo z nich zařazuje do kontextu teoretické vědy. Často se stává, že výsledky nejsou vůbec zveřejňovány, jsou předmětem obchodního tajemství. Jmenujme například systém hodnocení relevance odkazů v Internetových vyhledávačích, Internetové encyklopedie a antispamový software.

Matematika a Fyzika – nejzákladnější teoretické obory přírodních věd studují neurčitost jako přírodní jev a snaží se o její přesný popis. Dosažené výsledky jsou vhodné spíše pro ideální systémy nebo se jim svojí filozofií blíží, proto jsou pravděpodobně hůře využitelné pro kompromisní systémy. Jejich výhoda spočívá v propracovanosti a dobré uchopitelnosti teorií [4]. Pokud se přesto v průběhu výzkumu ukáže vhodné tyto dosažené výsledky využít, bude to velký přínos.

Obory informatiky – různá odvětví informatiky mohou také navrhnout zajímavé výsledky, které by ve výzkumu mohly být zohledněny (zpracování přirozeného jazyka, znalostní management, atd.).

Environmentální informatika a jiná praktická odvětví informatiky – bylo dosaženo výrazných výsledků, zejména z hlediska tvorby modelů [1], [3]. Environmentální data jsou velmi často zatížena neurčitostmi [7]. Rozdíl oproti běžným datům je dán tím, že environmentální data se většinou vztahují k nějakému času a složce životního prostředí, kterou popisují a zejména tyto dvě základní vlastnosti jsou častým zdrojem neurčitostí v environmentálních datech.

Další možné vědní nebo praktické obory – prakticky všechny vědní a praktické obory, které zpracovávají větší množství dat (resp. informací) se musí u těchto vstupních objektů setkávat s neurčitostmi. Několik příkladů: astronomie (zpracování obrazových informací s neurčitostmi), ekonomika (ekonomické modely chování trhu zatížené neurčitostmi), sociologie (průzkumy veřejného mínění), historie (zpracování historických dat, od slova datum, zatížených neurčitostmi), atd.

3 Motivační příklad

Ideální systémy jsou rozšířené a známé, kdežto kompromisní systémy jsou na začátcích vývoje. Často se dokonce ideální systémy využívají na vstupy, o kterých se obecně ví nebo předpokládá, že nejsou spolehlivé. Pokud by se ale ukázalo, že využití výhod kompromisních systémů nabízí vytvoření v informatice zatím nedosažené funkčnosti, situace by se pro kompromisní systémy podstatně zlepšila.

Ukažme si situaci na motivačním příkladu – zjištění telefonního čísla mobilního telefonu na určitého konkrétního člověka:

Ideální systém prohledá vlastní databázi (které stoprocentně důvěřuje) a číslo nalezne nebo nenalezne. Internet (zvláště stránky jako jsou různá volně přístupná fóra) není tímto systémem považován za důvěryhodný zdroj a je ignorován.

Člověk (po případném neúspěšném prohledání vlastního adresáře kontaktů) využije například vyhledávací systém Google, kam jednoduše zadá jméno člověka a na nalezených stránkách hledá požadované telefonní číslo. První číslo, které se mu zdá být pravděpodobné (objeví například Googlem archivovaný záznam z diskuse, kdy dotyčný člověk na sebe někomu dává kontakt), vyzkouší. Pokud se dovolá, řešení bylo nalezeno, pokud ne, omluví se a hledá dál.

Kompromisní systém by mohl člověku tato čísla sám na Internetu takto vyhledat a předložit k posouzení například s ohodnocením pravděpodobnosti úspěchu.

Nyní přejdeme k dalšímu kroku a toto zatím neúplné zadání motivačního příkladu rozšíříme. Předpokládejme, že jsme agentura, kterou si najal některý z mobilních operátorů, a máme za úkol zjistit zastoupení jeho anonymních SIM karet v určité skupině obyvatel. K tomu, abychom zjistili operátora z telefonního čísla, lze s určitým zanedbáním (přenos čísel mezi operátory) využít tabulku přidělených předčíslí.

Ideální systém může podat jen velmi omezené výsledky. Databáze těchto mobilních čísel jsou vzácné a použitelná odpověď pravděpodobně nebude nalezena.

Kompromisní systém by mohl využít vyhledání telefonních čísel vytipovaných příslušníků určené skupiny obyvatel a na základě nějaké aritmetiky pro počítání s neurčitými informacemi by mohl s minimálními náklady získat řešení, které bude vypovídací schopností zcela jistě lepší než u ideálního systému a řádově srovnatelné s řešením člověka.

Člověk by pravděpodobně využil výzkum pomocí dotazníkového šetření, který by byl finančně nákladný (což by se ještě zhoršilo, pokud by součástí výzkumu bylo i telefonické ověřování čísel). Případně by bylo možno ručně provést postup kompromisního systému. Takové řešení je ale pak dražší, protože je nutno platit pracovní sílu.

Příklad je záměrně zadán s možností získání dat z Internetu. Internet se v poslední době stává zdrojem informací č.1 a hledání odpovědí i na ty nejběžnější dotazy (např. ověření pravopisu slova pomocí zadání tohoto slova přímo v Google) je velmi běžné. Přitom data a informace na Internetu jsou obecně velmi nespolehlivá, zejména díky lidskému faktoru.

Řešení je navrženo tak, aby bylo vhodné zejména pro tento typ úloh. Z motivačního příkladu je zřejmé, že vyřešení takovýchto typů úloh by bylo určitým přínosem. Současně je zřejmé, že takové úlohy tady jsou, příkladů podobných našemu motivačnímu příkladu by se jistě dala najít celá řada, zejména z oblasti hromadného zpracování osobních údajů.

4 Závěr

Kvalita dat je oblastí, která v současné době výrazně vystupuje do popředí. Současně ale chybí její pevnější teoretické zázemí. Výsledky výzkumu by měly posloužit jako pevný teoretický základ pro případné praktické využití.

Budou dále podrobněji zmapovány a utříděny prakticky použitelné definice pojmů. Dále budou zmapovány možnosti reprezentace dat a navržena souhrnná míra k měření kvality dat. Tato míra bude testována tak, aby bylo možno odhadnout její úspěšnost. Jinak než na základě určité zpětné vazby nebude možno její závěry potvrdit, případně po přehodnocení doplnit a opravit.

5 Literatura

[1] Hřebíček, J., Dušek, L., Ráček, J., Hejč, M.: Data Quality in Biodiversity Information Management. In Proceedings of the 3rd Internationl Summer School on Computational Biology. 1. vyd. Brno : Masaryk University, 2007. od s. 171-177, 7 s. ISBN 978-80-210-4370-1.

[2] Helton, J.C., and F.J., Davis: Illustration of Sampling-Based Methods for Uncertainty and Sensitivity Analysis. Risk Analysis, 22(3), 591-622 (2002)

[3] Hřebíček, J., Hejč, M., Holoubek, I.: Current Trends in Environmental Modelling with Uncertainties. In Proceedings of the iEMSs Third Biennial Meeting "Summit on Environmental Modelling&Software". Burlington : International Environmental Modelling and Software Society, 2006. s. 342-347. ISBN 1-4243-0852-6.

[4] Gruska, J.: Quantum Computing, McGraw-Hill, 439 p, June 1999, ISBN: 0 07 709503-0

[5] Guide To The Expression Of Uncertainty In Measurement, norma publikována v roce 1995 v organizaci ISO http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=45315

[6] Kearfott R.B., Kreinovich V.: Applications of Interval Computations, Kluwer, 444 pp., Boston (1996)

[7] Král, J., Žemlička,M.: Service orientation in environmental information systems. In Proc. of 19th International Conference Informatics for Environmental Protection.EnviroInfo 2005. Networking Environmental Information. Brno, Czech Republic p. 12-23 (2005)

[8] Internetový portál Uncertainty in Engineering, portál je dostupný na URL http://www.uncertainty-in-engineering.net (září 2007)

Modelovanie procesov v krízovom riadení

Tomáš Ludík, Jaroslav Ráček

Masarykova univerzita, Fakulta informatiky
Botanická 68a, 602 00 Brno
xludik2@fi.muni.cz, racek@fi.muni.cz

Abstrakt
Príspevok sa zaoberá analýzou procesov v krízovom riadení, konkrétne analýzou Bojového poriadku jednotiek požiarnej ochrany. Analýza bola spracovaná pomocou prípadov použitia a procesných máp. Ďalším krokom bola identifikácie dát a jej následné prepojenie s modelovanými procesmi pomocou matíc CRUD. Procesné mapy sú popísané v jazyku XPDL. Použitím tohto štandardu sa kladie dôraz na interoperabilitu.

Abstract
The paper is focused on process analysis in crisis management, particularly analysis of Czech Fire and Rescue Services Act. The analysis is based on use case diagrams and process maps. The next step was data identification and its connection with process maps by CRUD matrixes. Process map are described by XPLD language. Using this standard is focused on high interoperability.

Kľúčové slová
Krízový manažment; Procesná analýza; Dynamická geovizualizácia; Interoperabilita.

Keywords
Crisis Management; Process Analysis; Dynamic Geovisualization; Interoperability.

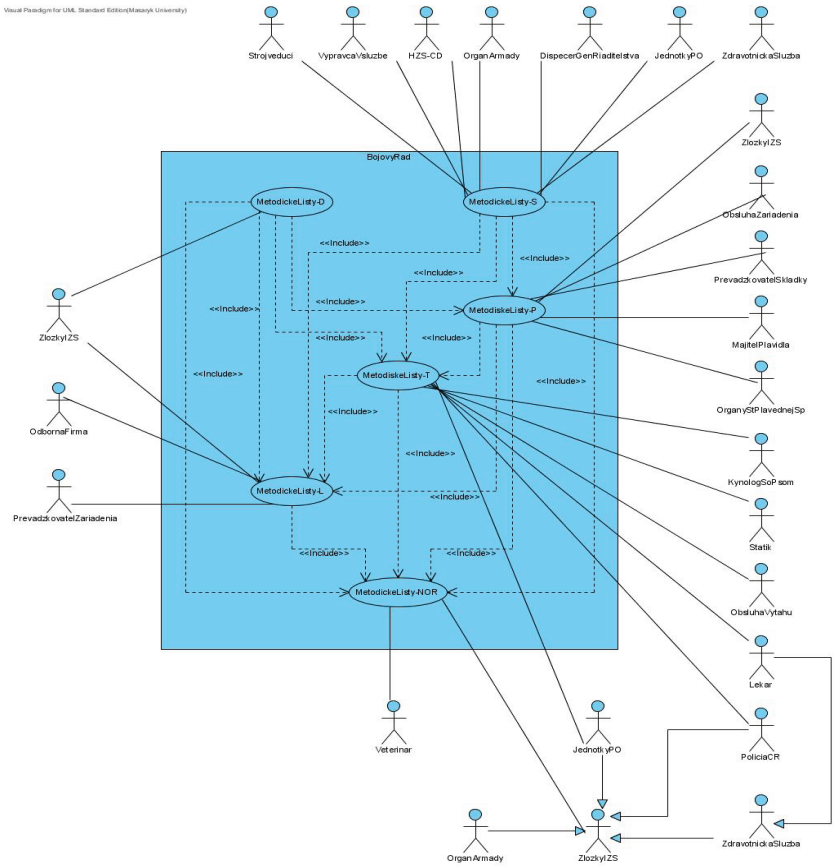
1 Úvod

Každodenne je potrebné riešiť množstvo krízových situácií. Ich riešenie nie je jednoduché. Treba si uvedomiť, že krízové situácie neohrozujú iba majetok a životné prostredie, ale aj ľudské životy. Aby mohli byť krízové situácie riešené čo najefektívnejšie, je potrené upustiť od funkčného riadenia, ktoré sa zameriava na znalosti jednotlivých ľudí. Ľudia majú v tomto prístupe rozdelené úlohy, ktoré si plnia. Na základe charakteru týchto úloh sú rozdelený do oddelených funkčných jednotiek. Lepšou alternatívou je riadenie procesné. Tento spôsob riadenia je orientovaný na výsledok práce. Práca nie je vykonávaná separátne v oddelených funkčných jednotkách, ale naopak nimi preteká. Pri tomto spôsobe riadenia dochádza k zlepšeniam hlavne formou optimalizácie a zjednodušenia celého toku práce. Riadenie procesov vedie k efektívnejšej koordinácii práce a tiež k zníženiu chybovosti. Problematika je ukázaná na Bojovom poriadku požiarnej ochrany.

2 UML – prípady použitia

Predtým ako sa zameriame na procesné mapy, je potrebné Bojový poriadok pochopiť ako celok. Jeden z dobrých globálnych pohľadov je možné vytvoriť pomocou prípadov použitia. Ide o typ diagramu v UML. Jeho hlavným účelom je vymedziť a dokumentovať súbor požiadaviek na modelovaný systém. Prvým krokom pri jeho tvorbe je vymedzenie hraníc modelovaného systému. V tomto prípade sú hranice modelovaného systému určené jednoznačne Bojovým poriadkom. Všetko ostatné je považované za okolie systému.

Ďalej je potrebné vytvoriť zoznam aktérov. Jedná sa o role, priradené osobám alebo predmetom, ktoré využívajú modelovaný systém. Kvôli vytvoreniu kompletného zoznamu aktérov, bolo potrebné prejsť všetky dokumenty Bojového poriadku. Pri tomto hľadaní bola dôležitá otázka: „Kto alebo čo systém používa, kto alebo čo s ním komunikuje?“ Počas tvorby zoznamu aktérov bolo zistené, že ich vlastnosti sa v dokumentoch opakujú alebo sú čiastočne spoločné. Aby diagram prípadov použitia nebol plný väzieb medzi prípadmi použitia a aktérmi, je potrebné použiť generalizáciu.



Obr.1: Diagram prípadov použitia

Po pochopení úlohy jednotlivých aktérov, je možné pustiť sa do vytvárania prípadov použitia. Prípad použitia je chápaný ako špecifikácia postupnosti činností, vrátane premenných postupností a chybových postupností, ktoré systém alebo podsystém môže vykonať prostredníctvom interakcie s vonkajšími (externými) aktérmi. V prípade Bojového poriadku je súbor činností obrovský, preto pri pokuse namodelovať systém prehľadnejšie, je potrebné nájsť väčšie súbory činností. Za tieto súbory činností sa budú považovať jednotlivé kapitoly metodických listov. Je možné spraviť preto, lebo v každej takejto kapitole sa nachádza zoznam metodických listov, ktoré sa zaoberajú veľmi blízkou tematikou. Výsledný diagram prípadov použitia je na obrázku 1.

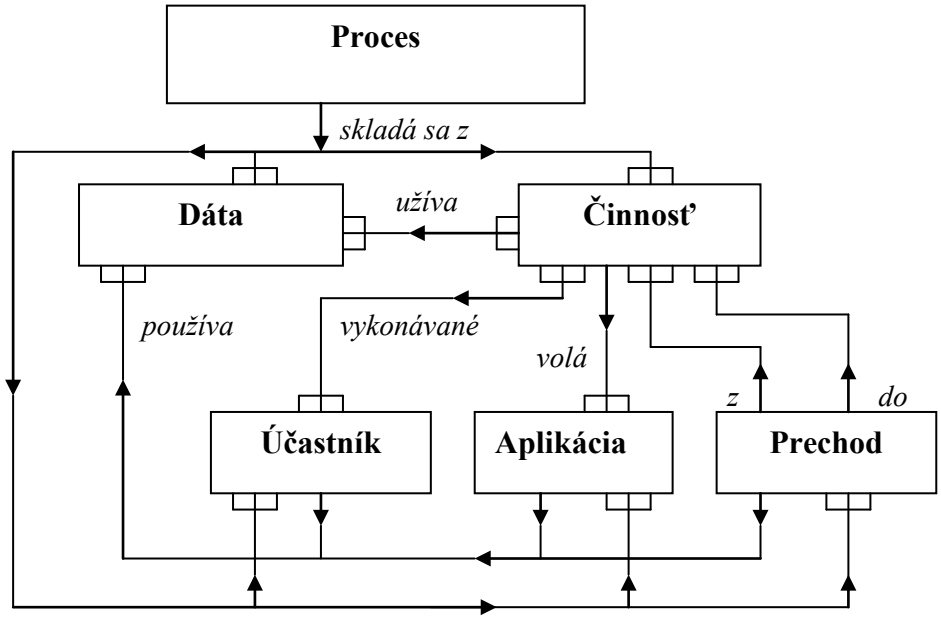
Po namodelovaní diagramov prípadov použitia, je možné začať s tvorbou špecifikácií pre jednotlivé prípady použitia. Tieto špecifikácie sú popísané pomocou procesných máp [1]. Procesné mapy boli vybrané preto, lebo sú schopné zachytiť a definovať postupnosť činností v metodických listoch.

3 Procesné mapy

Proces je po častiach usporiadaná množina činností, ktorá na základe jedného alebo viacerých vstupov tvorí opakovateľným spôsobom požadovaný výstup. Špecifikácie jednotlivých procesov [5] sa skladajú z nasledujúcich entít:

1. proces (popis celého procesu),
2. činnosť (definícia činností, z ktorých sa proces skladá),
3. prechod (definícia prechodov medzi činnosťami),
4. účastník (deklarácia účastníkov procesu),
5. aplikácia (deklarácia aplikácií používaných procesom),
6. dáta (deklarácia dát procesu).

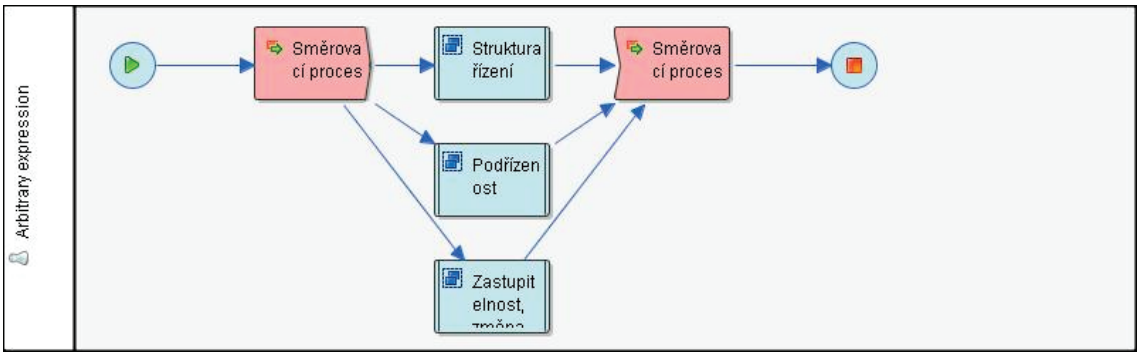
Vzájomné vzťahy entít znázorňuje tzv. Metamodel procesu na obrázku 2 [2].



Obr. 2: Metamodel procesu

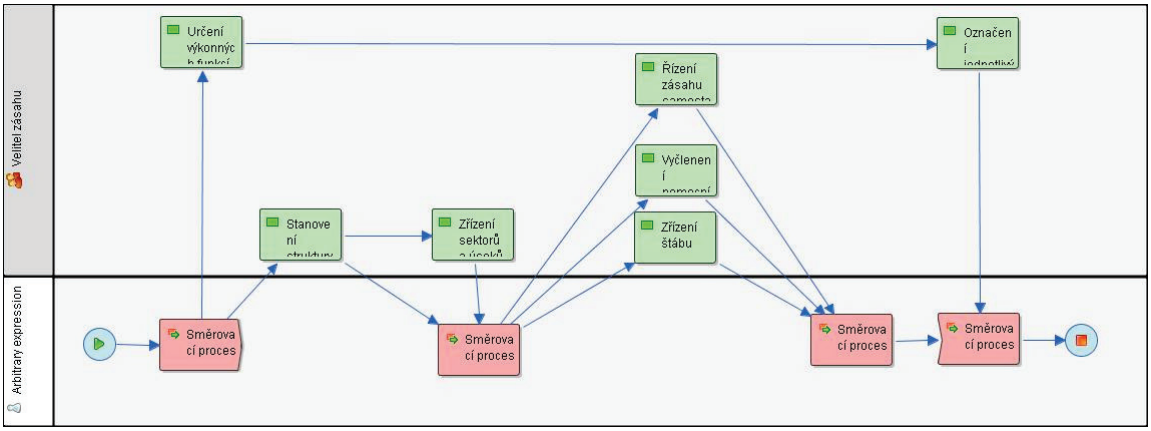
Pre ukážku procesnej mapy je spracovaný metodický list Riadenie zásahu [3]. Metodický list sa skladá s dvoch častí. Prvá časť má názov *Charakteristika* a druhá časť *Úlohy a postup činnosti*. S rovnakým rozdelením sa je možné stretnúť vo väčšine metodických listov Bojového poriadku. V časti Charakteristika sa stručne popisuje čoho sa metodický list týka, a taktiež sú tu spomenuté jeho základné pravidlá vykonávania. V časti Úlohy a postup činnosti sa nachádzajú podrobné popisy a pravidlá ako postupovať pri problematike zachytenej v danom metodickom liste.

Pri tvorbe procesných máp je potrebné sa viac zamerať na časť Úlohy a postup činnosti, kde sa dajú lepšie rozpoznať obsiahnuté procesy. V prípade metodického listu Riadenie zásahu sa táto časť delí na tri podčasti, ktorými sú *Štruktúra riadenia*, *Podriadenosť* a *Zastupiteľnosť, zmena, striedanie.* Každú túto časť je možné považovať za samostatný proces. Tieto procesy sú na sebe nezávislé, a taktiež nie je určené ich poradie. Aby sa dala táto situácia zachytiť zodpovedajúcou procesnou mapou, je potrebné použiť dva smerovacie procesy, s vetvením typu AND-Split a AND-Join, a tri bloky aktivít pre hlavné procesy. Samozrejme netreba zabudnúť na začiatok a koniec procesu. Týmto spôsobom vznikla základná procesná mapa prvého metodického listu kapitoly Ř, ktorá je znázornená na obrázku 3.



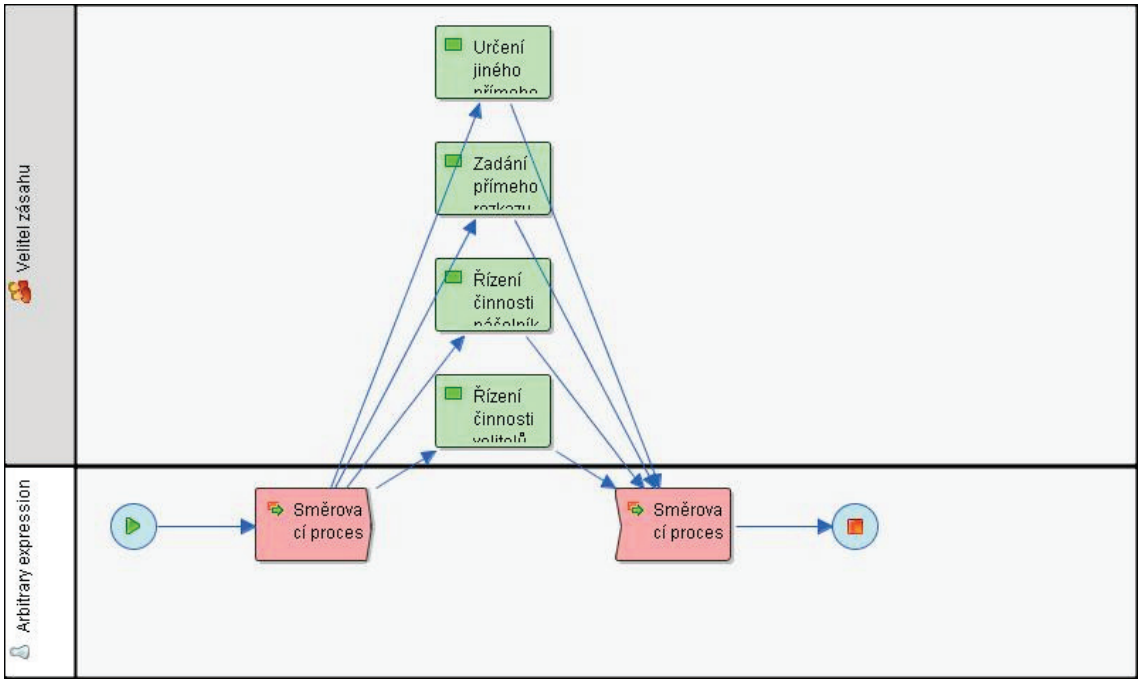
Obr. 3: Procesná mapa metodického listu č. 1 kapitoly Ř

Aby bola procesná mapa kompletná, je ešte potrebné namodelovať jednotlivé bloky aktivít, respektíve jednotlivé podprocesy hlavného procesu. Prvým blokom aktivít je *Štruktúra riadenia*. Skladá sa z dvoch skupín činností. Prvá skupina činností má za úlohu určiť výkonné funkcie ako je náčelník, člen štábu, veliteľ sektoru a úseku a ich následné označenie pomocou červenej pásky alebo vesty. Preto sa skladá z činnosti *Určenie výkonných funkcií* a jej nadväzujúcej činnosti *Označenie jednotlivých funkcií*. Druhá skupina činností analyzuje a následne ustanovuje štruktúru riadenia zásahu. Tá závisí na počte riadených jednotiek, ďalších zložkách IZS (Integrovaný záchranný systém), prípadne podľa ďalších síl a prostriedkov. Preto sa tu nachádza činnosť *Stanovenie štruktúry riadenia*, v ktorej prebieha analýza prostredia zásahu. Následne môže, ale nemusí veliteľ zásahu zriadiť sektory a úseky. Tie sa zriaďujú podľa miesta alebo taktiky zásahu. O tieto udalosti sa stará činnosť *Zriadenie sektorov a úsekov*. Na záver sa veliteľ zásahu rozhoduje akým spôsobom bude zásah riadiť. K dispozícii má tri možnosti, modelované činnosťami *Riadenie zásahu samostatne*, *Vyčlenenie pomocníkov* a *Zriadenie štábu*. Keďže je možné si vybrať len jednu možnosť, je v procesnom diagrame použité vetvenie typu XOR-Split a následné zlúčenie typu XOR-Join. Výslednú procesnú mapu tohto bloku aktivít je vidieť na obrázku 4.



Obr. 4: Procesná mapa podprocesu Štruktúra riadenia

Ďalším blokom aktivít je *Podriadenosť*. Hlavným účelom tohto bloku je určenie podriadenosti a nadriadenosťami medzi jednotlivými funkciami používanými pri zásahu. Činnosti v tomto bloku sú nasledujúce: *Určenie iného priameho nadriadeného*, *Zadanie priameho rozkazu*, *Riadenie činnosti náčelníka štábu* a *Riadenie činnosti veliteľa jednotiek*. Keďže sú rovnocenné, a nie je určené poradie ich vykonávania, tak aj ich výsledný podproces nebude veľmi zložitý. Výsledná procesná mapa tohto bloku je na obrázku 5.

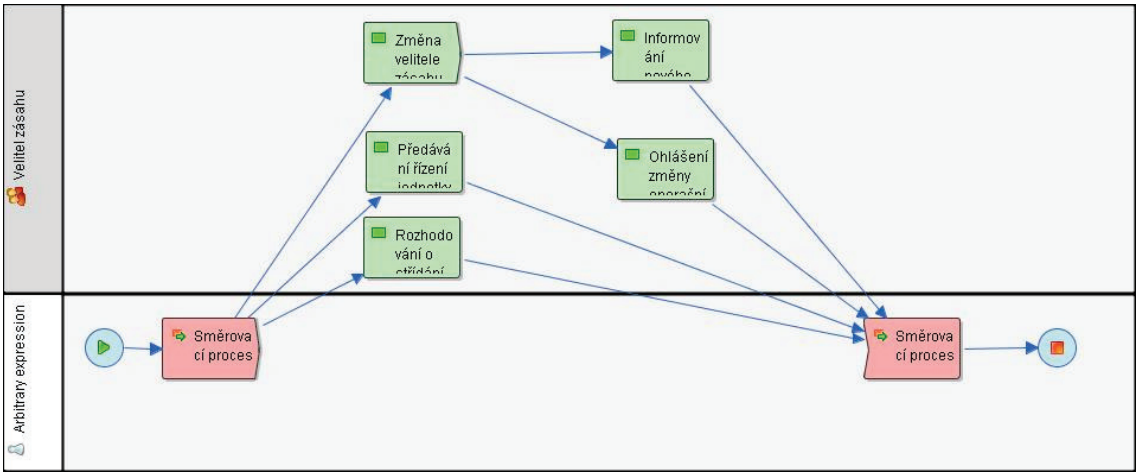


Obr. 5: Procesná mapa podprocesu Podriadenosť

Blok aktivít *Zastupiteľnosť, zmena, striedanie* je posledný blok, ktorý ostáva namodelovať. V tomto bloku sú popísané činnosti, ktoré je potrebné vykonať pri zmene veliteľa zásahu. Prvou z nich je činnosť *Zmena veliteľa zásahu*. Na túto činnosť sú naviazané ďalšie dve činnosti, ktoré je tiež potrebné vykonať. Jedná sa o činnosť *Informovanie nového veliteľa*, ktorá sa stará o to, aby nový veliteľ zásahu dostal všetky potrebné informácie o zásahu, a o činnosť *Ohlásenie zmeny operačnému stredisku*, ktorá zabezpečuje ohlásenie zmeny veliteľa zásahu na príslušné miesta.

Ďalšími činnosťami v tomto bloku sú *Predávanie riadenia jednotky*, ktorá zabezpečuje predanie riadenia po dobu neprítomnosti určenému zástupcovi, a *Rozhodovanie o striedaní hasičov*. Za túto činnosť zodpovedá veliteľ zásahu a jej cieľom je, aby zásah nebol prerušený, ale taktiež, aby nedošlo k ohrozeniu hasičov. Podobne, ako u väčšiny predchádzajúcich procesov, ani tu nie je jednoznačne určené poradie ich vykonávania. Jediné, čo jednoznačne vyplýva z metodického listu je to, že činnosti *Informovanie nového veliteľa* a *Ohlásenie zmeny operačnému stredisku* sú

vykonávané až po zmene veliteľa zásahu, teda po činnosti *Zmena veliteľa zásahu*. Výsledná procesná mapa je zobrazená na obrázku 6. Podobne sa postupovalo i pri tvorbe ostatných metodických listov Bojového poriadku jednotiek požiarnej ochrany.



Obr. 6: Procesná mapa podprocesu Zastupiteľnosť, zmena, striedanie

4 Identifikácia dát

Ďalším uhľom pohľadu na metodické listy je pohľad z hľadiska dát a dátových štruktúr. Inými slovami sa zisťovalo, aké informácie si systém potrebuje pamätať, a hľadal sa vhodný spôsob ich zastúpenia. Pretože sa metodické listy sústreďujú na popis činností a procesov, nie je vždy jednoduché dáta a dátové štruktúry nájsť. Na základe týchto skutočností [6] sa popísali dáta a dátové štruktúry v miere detailu, ktorú umožnili príslušné metodické listy. Pre ukážku je spracovaný metodický list Riadenie, ktorý sa nachádza v tabuľke č. 1.

Metodické listy kapitoly Ř	**Dáta a dátové štruktúry**
1. Riadenie zásahu	• Rozhodnutia, • Zastupiteľnosť, • Počet jednotiek, • Výkonné funkcie, • Členitosť miesta zásahu, • Podriadenosť, • Správa o zásahu.

Tab. 1: Identifikácia dát

5 Matica CRUD

Následne došlo k vyvažovaniu procesného a dátového modelu [4] pomocou matice CRUD (Create - Read - Update - Delete). Matica ukazuje aké činnosti vykonávajú jednotlivé procesy a aké dáta a dátove štruktúry k tomu potrebujú. Riadky matice reprezentujú identifikované dátové položky, stĺpce predstavujú jednotlivé činnosti procesu. V príslušných poliach matice sú potom uvedené operácie: vytvorenie (C), čítanie (R), zmena (U) a mazanie (D), ktoré proces vykonáva s príslušnými údajmi.

6 Interoperabilita

Pri vytváraní štandardov pre krízový manažment a jeho podporu formou dynamickej geovizualizácie v Českej republike za použitia moderných ICT je nevyhnuté nájsť spoločné pochopenie a zhodu aj s okolitými Európskymi krajinami. Rovnako sa očakáva, že návrh Českého národného štandardu bude dovoľovať užívateľom jednoliato spolupracovať a vymieňať si geografické dáta a informácie na lokálnej, národnej ale aj medzinárodnej úrovni. Z tohto dôvodu je pri návrhu celého systému kladený dôraz na jeho interoperabilitu. To sa prejavuje pri výbere vhodného formátu pre uloženie modelovaných procesov a aplikačnej ontológie.

Všetky modelované procesy sú vytvorené vo formáte *XPDL* (*XML Process Definition Language*). Ide o štandard Workflow Management Coalition, ktorý sa snaží vytvoriť jednotný formát pre ukladanie namodelovaných procesov. Tento štandard v súčasnosti podporuje viac ako 70 softwarových produktov. Účelom tohto štandardu je vytvoriť procesné definície v takom formáte, aby mohli byť prenášané medzi rôznymi modelovacími nástrojmi alebo automaticky spracovávané workflow manažment systémom [3].

7 Záver

Z pohľadu zvládania krízových situácií sa procesný spôsob riadenia ukazuje ako veľmi efektívny nástroj, pretože jednoznačne definuje role a zodpovednosť jednotlivých aktérov, a zároveň jednoznačne hovorí, aké úlohy a v akom poradí majú byť vykonávané, čo uľahčuje rozhodovanie v krízových situáciách. Tento príspevok taktiež popisuje základné princípy tvorby procesných máp, ich analýzy v oblasti verejnej správy, pričom sa snaží byť nezávislá na konkrétnom softvérovom nástroji a špecifickej notácii, čo dokazuje hlavne použitie štandartných modelovacích nástrojov ako sú UML diagrami a taktiež modelovanie procesných diagramov v notácii XPDL. Vytvorené procesné mapy sa dajú použiť k simulácii a optimalizácii procesov a tým k zrýchleniu riešenia krízových sittuácii. Zmapovanie procesov je tiež prvým krokom k ich automatizácii.

8 Literatúra

[1] fiala, j., ministr, j.: Průvodce analýzou a modelováním procesů. VŠB-TU, Ostrava, 2003.

[2] Hollingswort, D.: Terminology & Glossary. Workflow Management Coalition, 1999.

[3] Ludík, T.: Analýza a návrh procesov v oblasti krízového riadenia (diplomová práca). Masarykova univerzita, Brno, 2008.

[4] Ráček, J.: Strukturovaná analýza systémů. Masarykova univerzita, Brno, 2006.

[5] Ráček, J.: Workflow správních procesů v oblasti životního prostředí (disertační práce). Masarykova univerzita, Brno, 2002.

[6] Staněk, K., Friedmannová, L., Kubíček, P.: Decision Support Cartography for Emergency Management. ISPRS archives, Osnabrueck, 2007.

Príspevok bol spravcovaný ako súčásť riešenia výskumného zámeru MSM0021622418 Ministerstva školstva, mládeže a telovýchovy ČR s názvom „Dynamická geovizualizácia v krízovom manažmente“.

Sborník příspěvků
4. letní školy aplikované informatiky
Bedřichov, 24. – 26. srpna 2007

Editor
prof. RNDr. Jiří Hřebíček, CSc.

Vydala Masarykova univerzita v roce 2007
1. vydání, 2007
Tisk Vydavatelství MU, Brno – Kraví Hora

ISBN: 80-210-4146-3