UNIVERZA V LJUBLJANI
EKONOMSKA FAKULTETA

MAGISTRSKO DELO

ANALIZA USKLAJENOSTI SISTEMOV ZA UPRAVLJANJE PODATKOVNIH BAZ S TEHNOLOGIJAMI ELEKTRONSKEGA POSLOVANJA

Ljubljana, december 2005 Tadej Poljšak

IZJAVA

Študent Tadej Poljšak izjavljam, da sem avtor tega magistrskega dela, ki sem ga napisal pod mentorstvom prof. dr. Jurija Jakliča in skladno s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah dovolim objavo magistrskega dela na fakultetnih spletnih straneh.

V Ljubljani, 27. november 2005 Podpis: _____

1. UVOD 1

2. SPREMEMBE, KI JIH XML PRINAŠA V ELEKTRONSKO POSLOVANJE 4

2.1. NOVE OBLIKE ELEKTRONSKEGA POSLOVANJA 4
2.2. ELEKTRONSKA IZMENJAVA PODATKOV 5
2.3. TEHNOLOGIJA XML 6
2.3.1 Osnove 6
2.3.2 Načini opisa podatkov 7
2.3.2.1 DTD datoteka 7
2.3.2.2 XML shema 8
2.3.2.3 Primerjava 8
2.3.3 Načini pregledovanja 8
2.3.3.1 SAX razčlenjevalnik 9
2.3.3.2 DOM razčlenjevalnik 10
2.3.4 Pretvorbe XML 12
2.4. SPLETNE STORITVE 14
2.4.1 Motivacija 15
2.4.2 Arhitektura 15
2.4.3 Standardi 17
2.4.3.1 Ovojnica 17
2.4.3.2 Opis 18
2.4.3.3 Imenik 19

3. RAZLIKE MED XML TEHNOLOGIJO IN RELACIJSKIMI PODATKOVNIMI BAZAMI 19

3.1. PODATKOVNI MODEL 21
3.1.1 Hierarhični model 21
3.1.2 Relacijski model 23
3.1.3 Polstrukturiran model 24
3.1.4 XML kot model 26
3.2. INTEGRACIJA XML DOKUMENTOV V PODATKOVNIH BAZAH 28
3.2.1 Usmerjenost XML dokumentov 28
3.2.1.1 Podatkovno usmerjeni XML dokumenti 29
3.2.1.2 Dokumentno usmerjeni XML dokumenti 29
3.2.2 Podpora tehnologiji XML v podatkovnih bazah 30
3.2.2.1 Podatkovne baze z delno podporo tehnologiji XML 30
3.2.2.2 Podatkovne baze s popolno podporo tehnologiji XML 32
3.2.3 Shranjevanje XML dokumentov v podatkovnih bazah 35
3.2.3.1 Dekompozicija XML dokumentov 36
3.2.3.2 Shranjevanje izvirnih XML dokumentov 40
3.3. POIZVEDBE 43
3.3.1 Poizvedovalni jezik XQuery 45
3.3.2 Izrazi za pot 45
3.3.3 Konstruktorji elementov 47
3.3.4 FLWR izrazi 47
3.3.5 Operatorji in funkcije 48
3.3.6 Pogojni izrazi 49

4. PRIMERJALNA ANALIZA SUBP ... 49
4.1. ODLOČITVENI MODEL ... 52
4.1.1 Določitev kriterijev ... 52
4.1.1.1 Upravljanje podatkov ... 53
4.1.1.2 Uparitev podatkov ... 55
4.1.1.3 Kontrola podatkov ... 57
4.1.1.4 Poizvedbe ... 58
4.1.2 Funkcija koristnosti ... 59
4.2. OPIS VARIANT ... 60
4.2.1 DB2 V8 ... 60
4.2.1.1 Upravljanje podatkov ... 61
4.2.1.2 Uparitev podatkov ... 64
4.2.1.3 Kontrola podatkov ... 67
4.2.1.4 Poizvedbe ... 68
4.2.2 Oracle 10g ... 68
4.2.2.1 Upravljanje podatkov ... 69
4.2.2.2 Uparitev podatkov ... 70
4.2.2.3 Kontrola podatkov ... 71
4.2.2.4 Poizvedbe ... 72
4.2.3 SQL Server 2005 ... 72
4.2.3.1 Upravljanje podatkov ... 72
4.2.3.2 Uparitev podatkov ... 74
4.2.3.3 Kontrola podatkov ... 75
4.2.3.4 Poizvedbe ... 75
4.3. REZULTATI VREDNOTENJA VARIANT ... 76
4.3.1 Upravljanje podatkov ... 77
4.3.2 Uparitev podatkov ... 77
4.3.3 Kontrola podatkov ... 79
4.3.4 Poizvedbe ... 79
5. PRIMER IMPLEMENTACIJE UPORABE XML V SUBP ... 80
5.1. ANALIZA PRIMERA V PRAKSI ... 81
5.1.1 Uvod ... 81
5.1.2 Zbiranje in analiza zahtev ... 81
5.2. PODATKOVNI NAČRT ... 82
5.2.1 Konceptualni model ... 82
5.2.2 Logični načrt ... 83
5.3. VPELJAVA FIZIČNEGA NAČRTA IN XML OBJEKTOV ... 84
5.3.1 Izdelava in priprava podatkovne baze ... 84
5.3.2 Izdelava XML sheme ... 84
5.3.3 Izdelava meta-sheme ... 85
5.3.4 Priprava podatkovne baze ... 87
5.3.5 Zapis XML v podatkovno bazo ... 87
6. UGOTOVITVE ... 89
6.1. RAZLIKE MED PARADIGMAMA ... 90
6.2. PRIMERJALNA ANALIZA SUBP ... 90
6.3. PRIMER IMPLEMENTACIJE UPORABE XML V SUBP ... 91
7. ZAKLJUČEK ... 93

8. **LITERATURA IN VIRI 95**

1. UVOD

Informacijska tehnologija postaja ključni vir poslovanja. Učinkovite, informacijsko podprte oblike poslovanja označuje sodoben izraz elektronsko poslovanje. Prehod organizacij na elektronsko poslovanje izpostavlja potrebo po skladnem in sočasnem ter povezanem obravnavanju poslovnih procesov ter uporabo možnosti, ki jih nudi informacijska tehnologija pri prenovi poslovanja (Hsieh, 2003, str. 68; Jerman Blažič, 2001, str. 7; Turban, 2004b, str. 20).

Klasična ekonomija doživlja konec dvajsetega in začetek enaindvajsetega stoletja korenite spremembe na poti v digitalno ekonomijo. Ta transformacija je povezana z informatizacijo, selitvijo poslovanja v večji ali manjši meri na splet, prilagoditvijo poslovanja novim poslovnim modelom, itd… Splet postaja ne zgolj eden od kanalov, temveč prevladujoča pot za distribucijo dobrin, storitev, znanja in dela, preko katerega podjetje komunicira s svojimi poslovnimi partnerji (Turban, 2004a, str. 3).

Digitalna ekonomija prinaša nove poslovne modele. Večina od njih je povezana s spletom. V vlogi posrednika in skupnega imenovalca najdemo jezik XML. Le-ta nastopa tudi kot povezovalni člen obeh že omenjenih paradigem. Zasnovan je bil kot jezik za opisovanje podatkov za razliko od jezika HTML, katerega namen je prikaz in oblikovanje podatkov (W3C/XML, 2005), prerasel pa je najbolj razširjen standard za izmenjavo podatkov na spletu. Pri tem je pomembno poudariti, da XML ni zamenjava za HTML, saj je bil razvit v popolnoma druge namene.

Podatkovne baze srečujemo dandanes praktično na vsakem koraku. So logična posledica modeliranja resničnega okolja (Mohorič, 1997, str. 5). S pomočjo abstrakcij želimo z modeliranjem okolja zaznati glavne zakonitosti, njihove karakteristike ter medsebojne odvisnosti. Podatke shranjujemo v podatkovne baze z namenom pridobivanja informacij. Prav informacija pa postaja strateški potencial podjetja, ki se na trgu popolne konkurence srečuje s svojimi tekmeci.

Iz povedanega je razvidno, da imamo opravka z dvema kulturama (Abiteboul, 2000, str. 3):

- na eni strani zaznavamo splet kot enostaven in splošen standard za izmenjavo informacij. Splet deluje na osnovnem načelu členjenja informacij v obvladljive in poimenovane enote, ki jih lahko prenašamo po spletu. Osnovno enoto informacije na spletu na ta način predstavlja datoteka, ki se ponaša z unikatnim spletnim naslovom (URL). Uporabniki po svetu lahko dostopijo do takšne informacije, v kolikor poznajo njen naslov. Datoteke so zapisane v označevalnem jeziku (HTML), ki vsebini doda predstavitveno funkcionalnost;

- na drugi strani imamo nekaj desetletij star svet relacijskih podatkovnih baz, kjer je konceptualni model modeliranega sveta preslikan v relacijski model (Mohorič, 1997, str. 5) in vsebujejo mehanizme ter poizvedovalne jezike, s pomočjo katerih pridobivamo informacije. Vsebujejo mehanizme za podporo množici uporabnikov ter transakcijski način dela. Poizvedbe se v podatkovnih relacijskih bazah izvajajo s pomočjo poizvedovalnega jezika SQL.

V vsakdanjem življenju se torej srečujemo z dvema različnima paradigmama, ki sta med seboj soodvisni in povezani, pa kljub temu tako različni. S tem se pojavi potreba po premostitvi razlik, ki obstajajo med obema kulturama.

Poslovne potrebe in pritiski narekujejo podjetjem hitre in učinkovite odzive na spremembe v okolju. Časi trajanja informacijskih projektov in poslovnih procesov se zaradi lažjega pretoka informacij krajšajo. Podjetja se med seboj združujejo in kupujejo. Takšne združitve so seveda pogojene z učinkovitim povezovanjem in razvojem informacijskih sistemov. Podjetje mora biti sposobno sprejeti, obdelati ali poslati informacije v čim krajšem času. Spletne tehnologije nam narekujejo nov, hitrejši tempo.

Vse to so razlogi, ki narekujejo uporabo učinkovite informacijske podpore, brez katere podjetje danes ni sposobno preživeti na tržišču. Prav podatkovne baze pa se pri tem običajno nahajajo v samem jedru informacijskega sistema, ki zadošča informacijskim potrebam podjetja.

Izbira ustreznega sistema za upravljanje baz podatkov pa lahko kritično vpliva na dinamiko razvoja projektov elektronskega poslovanja, saj niso vsi sistemi za upravljanje baz podatkov (Bates, 2002, str. 531) enako pripravljeni na spletne tehnologije.

V novejši literaturi na tem področju zasledimo več razprav o:

- raznolikosti podatkov realnega sveta, ki povzroča težave pri implementaciji le-teh v sistemih za upravljanje baz podatkov (Abiteboul, 2000, str. 7);
- razvoju sistemov za upravljanje baz podatkov, ki vodi v smer podpore XML tehnologiji, saj večji ponudniki programske opreme vlagajo znatne napore na tem področju (Smith, 2003, str. 96);
- integraciji XML z obstoječimi sistemi in tovrstnih problemih pri razvoju informacijskih sistemov (Vakali, 2005, str. 62);
- razlikah med tehnologijo XML in sistemi za upravljanje baz podatkov (Kappel, 2004, str. 345; Trippe, 2002, str. 25; Champion, 2001);
- opravljenih določenih analizah o podpori XML v sistemih za upravljanje baz podatkov (Seng, 2003, str. 111 – 120).

Namen naloge je sinteza omenjenih spoznanj na tem področju, dopolnjenih z lastnimi ugotovitvami. Vse to me bo vodilo k odgovoru na vprašanje, ali so sistemi za upravljanje s podatkovnimi bazami pripravljeni na izzive, ki jih prinaša elektronsko poslovanje.

V magistrskem delu bom analiziral in ugotovil, ali so sistemi za upravljanje baz podatkov zreli in pripravljeni na izzive, ki jih ponuja splet in elektronsko poslovanje. Ugotovil bom, kakšne so danes možnosti uporabe XML v podatkovnih bazah in kakšne možnosti prinaša prihodnost.

Cilji, ki jih bom v nalogi zasledoval:

- V uvodnih poglavij bom razčlenil spremembe, ki jih prinaša elektronsko poslovanje. Le-te se nanašajo na nove poslovne modele, trženje na spletu in elektronsko izmenjavo podatkov. V ozadju teh sprememb se skrivajo tehnologije kot so XML in spletne storitve.

- V nadaljevanju bom ugotovil razlike med tehnologijo XML in sistemi za upravljanje podatkovnih baz. Te se nanašajo na razlike med relacijskim in pol-strukturiranim modelom, kamor uvrščamo XML. Podpora XML v sistemih za upravljanje baz podatkov prinaša nove podatkovne tipe. Pojasnil in opisal bom, na kakšne načine lahko v podatkovno bazo zapišemo XML podatke. S podporo tehnologiji XML pa se razvijajo tudi poizvedovalni jeziki, s katerimi pridobivamo podatke.

- S primerjalno analizo bom primerjal glavne izdelke na tržišču podatkovnih baz glede podpore XML. Na podlagi analize bom izdelal odločitveni model.

- V zaključnem delu bom z enim od orodij pokazal implementacijo konkretne uporabe jezika XML v podatkovni bazi. Cilj tega je ovrednotiti praktično uporabnost podpore tehnologiji XML v sistemih za upravljanje baz podatkov.

Podlago za pripravo magistrskega dela bom črpal iz del domačih in tujih avtorjev. V dodatno pomoč mi bodo lastne več kot desetletne izkušnje na področju podatkovnih baz in računalniške izmenjave podatkov. Na podlagi študija literature bom analiziral spremembe, ki jih v svet podatkovnih baz prinašajo tehnologije elektronskega poslovanja. Tema, katere se lotevam, je relativno nova, saj prva vprašanja o povezavi in vlogi jezika XML v podatkovnih bazah najdemo šele v zadnjih letih. Ravno zato pričakujem, da bom dodatna in najnovejša spoznanja našel v bogatih bazah člankov revij in publikacij na spletu. Na osnovi strokovnih člankov in literature bom v nadaljevanju primerjal razlike med podatkovnimi bazami in tehnologijami elektronskega poslovanja.

S primerjalno analizo bom primerjal glavne izdelke na tržišču podatkovnih baz glede podpore XML. Na podlagi analize bom določil kriterije in odločitveni model.

Za izdelavo večkriterijskega odločitvenega modela in vrednotenje variant bom uporabil lupino ekspertnega sistema DEX-i in na ta način ugotovil praktično uporabnost podpore tehnologiji XML v sistemih za upravljanje baz podatkov.

Sledi študija primera iz prakse, kjer bom analiziral možnosti elektronskega prijavljanja v zdravstveno zavarovanje z uporabo tehnologije XML in možnosti zapisa podatkov v podatkovno bazo. Na osnovi izbranega orodja bom prikazal možnost implementacije obravnavanega primera.

2. SPREMEMBE, KI JIH XML PRINAŠA V ELEKTRONSKO POSLOVANJE

Globalizacija, združevanje podjetij, podiranje mej med državami, seljenje proizvodnje na cenejša tržišča delovne sile so le nekatere od sprememb, s katerimi se podjetja danes srečujejo. Na drugi strani opažamo nesluten razmah elektronskega poslovanja. Pri obeh pojavih gre pravzaprav za pojav simbioze, soodvisnosti. To pomeni, da so na eni strani naštete spremembe glavni krivec za pojav elektronskega poslovanja, na drugi strani je elektronsko poslovanje glavno gonilo za nastanek teh dejavnikov in sprememb poslovanja, brez katerih bodo podjetja le stežka obstala na tržišču.

2.1. NOVE OBLIKE ELEKTRONSKEGA POSLOVANJA

Tradicionalna (t.i. »brick-and-mortar«) podjetja se soočajo z naraščajočimi pritiski na tržišču. Ti pritiski silijo podjetja v spremembe načina poslovanja. Te spremembe pa se kažejo v uvajanju različnih nivojev elektronskega poslovanja, ki za seboj potegnejo nižanje stroškov, krajšanje poslovnih ciklov, …

Posamezne oblike elektronskega poslovanja se ločijo po naravi opravljenih transakcij ter odnosih med udeleženci (Turban, 2004a, str. 7):

- B2B (angl. business-to-business): vsi udeleženci v poslovnem modelu so pravne osebe. Večina opravljenih transakcij v elektronski trgovini je tega tipa (85%).

- B2C (angl. business-to-customer): gre za odnos, kjer podjetje prodaja dobrine ali storitve končnemu potrošniku.

- C2B (angl. customer-to-business): v tem primeru gre za odnos, kjer posamezniki uporabljajo splet za prodajo dobrin ali storitev podjetjem.

- C2C (angl. customer-to-customer): v tem modelu sodelujeta dva posameznika, ki si izmenjujeta dobrine ali storitve.

- Mobilna trgovina: aktivnosti se opravijo brezžično, s pomočjo prenosnih naprav, kot so prenosni računalniki ali telefoni ter dlančniki.

- B2E (angl. business-to-employees): v tej kategoriji podjetje elektronsko komunicira s svojimi zaposlenimi v obliki informacij, storitev ali izdelkov.

- Učenje na daljavo: v tem primeru se učenje izvaja preko spleta. Podjetja uporabljajo to obliko za izobraževanje zaposlenih. Prav tako večina univerz, ki ponujajo takšno obliko študija.

- e-država: država lahko na ta način posreduje izdelke, storitve ali informacije podjetjem (G2B) ali državljanom (G2C).

2.2. ELEKTRONSKA IZMENJAVA PODATKOV

Elektronska izmenjava podatkov je izmenjava podatkov med poslovnimi subjekti z namenom zmanjšanja stroškov poslovanja (Hsieh Chang-Tseh, 2003, str. 68). Koristi elektronske izmenjave podatkov izhajajo predvsem iz:

1. Zmanjšanja stroškov dela.
   Elektronska izmenjava podatkov odpravlja ročno pošiljanje dokumentov in vnos podatkov v informacijski sistem.

2. Vrednosti informacije.
   Znano je, da vrednost informacije pada s časovno oddaljenostjo od njenega nastanka. Elektronska izmenjava močno pospeši pretok informacij.

3. Boljša kakovost informacij.
   Brez ročnega vnosa podatkov tudi ni napak zaradi ročnega vnosa. Napačni podatki v informacijskem sistemu običajno pomenijo dodatne stroške.

4. Standardizacija.
   Standardizacija postopkov in protokolov pripomore k boljši komunikaciji med podjetji in pospešuje integracijo poslovnih procesov.

Elektronska izmenjava podatkov prinaša velike spremembe pri komunikaciji med podjetji. Ob tem je potrebno upoštevati, da uvajanje elektronske izmenjave podatkov v podjetju pogosto za seboj potegne tudi korenite organizacijske spremembe.

2.3. TEHNOLOGIJA XML

2.3.1 OSNOVE

XML je bil zasnovan kot jezik za opisovanje podatkov za razliko od HTML, katerega namen je prikaz in oblikovanje podatkov (W3C/XML, 2005). Pri tem je pomembno poudariti, da XML ni zamenjava za HTML, saj je bil razvit v popolnoma druge namene.

Kratica XML pomeni Extensible Markup Language. XML je označevalni jezik, sestavljen iz t.i. označb (angl. tags). Vsebina označb ni predpisana, kar pomeni, da je jezik poljubno razširljiv. Slika 1 prikazuje primer XML podatkov, ki vsebujejo informacije o naslovu osebe ali organizacije.

Slika 1: Primer podatkov v XML obliki

```
<Naslov>
     <Ulica>Trubarjeva 34</Ulica>
     <SifraPoste>1000</SifraPoste>
     <KrajPoste>Ljubljana</KrajPoste>
     <SifraObcine>117</SifraObcine>
     <ImeObcine>Ptuj</ImeObcine>
     <SifraDrzave>205</SifraDrzave>
     <ImeDrzave>Slovenija</ImeDrzave>
</Naslov>
```

V preteklosti sta na področju izmenjave podatkov kraljevali predvsem dve obliki zapisa podatkov:

- splošno uveljavljen ASCII zapis. Slabost tovrstnega zapisa podatkov je nestandardnost zapisa, saj splošno uveljavljeni standardi na tem področju ne obstajajo in gre običajno za izmenjavo podatkov na bilateralni ravni. Iz oblike zapisa ni mogoče neposredno razbrati strukture podatkov. Pogosto so podatki zapisani v binarni obliki in zatorej človeku neberljivi.

- EDIFACT oblika, ki je zahtevna za obdelavo in zahteva licenčno programsko opremo. Ta način izmenjave se v Sloveniji ni nikoli širše uveljavil. Edini tovrstni načini izmenjave podatkov so se obdržali v primerih, ko ima ena od udeleženih strani ustrezno (gospodarsko) moč in lahko na ta način uveljavi svojo voljo.

Prav v tem pa pridejo do izraza prednosti XML. XML je neodvisen od programske in strojne platforme. Isto datoteko lahko uporabimo za izmenjavo podatkov na operacijskem sistemu Windows XP, HP Unix-u, ali na IBM centralnem računalniku z z/OS.

XML uporablja za opis podatkov dve vrsti datotek: DTD datoteke ali XML sheme. Oba načina bom podrobneje predstavil v nadaljevanju. Ker je iz dokumenta razviden opis podatkov pravimo, da je XML samoopisljiv.

XML datoteka je zgolj dokument s podatki, ki sam po sebi ni uporaben. Za obdelavo teh podatkov je še vedno potrebno napisati kos programske kode. XML dokumente obdelujemo z razčlenjevalniki (angl. parser). V praksi poznamo dva tipa razčlenjevalnikov: SAX[1] in DOM[2]. V nadaljevanju bom obravnaval oba načina natančneje.

2.3.2 NAČINI OPISA PODATKOV

Osnovna značilnost XML jezika, ki izhaja že iz njegovega imena, je razširljivost. Ta lastnost nam mnogokrat pride prav, saj nam omogoča svobodo pri oblikovanju standardov za izmenjavo podatkov. Na drugi strani pa potrebujemo mehanizme, ki zagotavljajo ustrezno strukturo datoteke. Tipičen primer, ko se zahteva vnaprej predpisana struktura datoteke so spletne storitve. V ta namen so se najprej razvile t.i. DTD (angl. data type definition) datoteke z omejeno funkcionalnostjo. Prav zaradi omejene funkcionalnosti danes uporaba DTD datotek zamira. Večinoma jih je nadomestil nov standard, imenovan XML shema.

Prav tako kot DTD dokument lahko tudi XML shemo oblikujemo ročno, še bolje pa je, če pri tem uporabimo temu namenjeno orodje (XMY Spy, Visual XML, …).

2.3.2.1 DTD DATOTEKA

Opis in zaporedje podatkov XML datoteke zapišemo v posebnem dokumentu imenovanem DTD. DTD predstavlja starejši in enostavnejši način opisa podatkov. Dokument je veljaven, kadar ustreza pravilom, določenim v opisu tipa dokumenta (Jurič, 2001, str. 10).

<u>V DTD zapišemo naslednje omejitve XML dokumenta:</u>

1. Označbe, ki se lahko ali morajo pojaviti,
2. kako pogosto se označbe pojavljajo,

[1] Več o SAX razčlenjevalniku v poglavju 2.3.3.1.
[2] Več o DOM razčlenjevalniku v poglavju 2.3.3.2.

3. gnezdenje označb in
4. katere atribute lahko uporabimo in njihove privzete vrednosti.

Slabosti DTD (Harold, 2004, str.29):

1. DTD sintaksa ni v XML zapisu,
2. števila ponovitev posameznega XML elementa/atributa ni mogoče določiti,
3. določenih omejitev ni mogoče zapisati (dolžina podatka, predpisana oblika podatka).

DTD dokument lahko oblikujemo ročno, še bolje pa je, če pri tem uporabimo temu namenjeno orodje (XMY Spy, Visual XML, …).

DTD datoteke so široko uveljavljene in podprte v programski opremi. Vendar pa so potrebe uporabnikom kmalu presegle možnosti, ki jih DTD ponuja. Vse to je vodilo k razvoju zmogljivejšega mehanizma za kontrolo XML podatkov - XML sheme (Vlist, 2002, str.4)

2.3.2.2 XML SHEMA

XML shema je novejši način opisa podatkov in temu primerno za razliko od DTD zmogljivejši način opisa podatkov. Sheme so zapisane v XML jeziku.

Sheme omogočajo številne že vgrajene osnovne podatkovne tipe, lahko pa definiramo svoje lastne podatkovne tipe. Pri tem lahko vnaprej predpišemo obliko zapisa podatka (angl. pattern), dovoljene vrednosti (angl. restrictions), ipd…

Za XML sheme je značilna delna objektna orientiranost, saj lahko podatkovne tipe izpeljujemo.

2.3.2.3 PRIMERJAVA

Iz prejšnjih dveh poglavij lahko naredim primerjavo med obema načinoma kontrole XML dokumentov. Odločitev, katero od obeh alternativ uporabimo za kontrolo XML podatkov, je odvisna od tehničnih pogojev in zahtev. Tabela 1 obravnava možne dejavnike, ki vplivajo na to odločitev.

2.3.3 NAČINI PREGLEDOVANJA

Z načinom pregledovanja opredelim način dostopa in obdelave podatkov, zapisanih v XML obliki. V praksi sta se uveljavila dva tipa razčlenjevalnikov: SAX in DOM, ki ju podrobneje opisujem v nadaljevanju.

Tabela 1: Primerjava značilnosti DTD datoteke in XML sheme

Značilnost	DTD datoteka	XML shema
Oblika zapisa	DTD zapis.	Standardni XML zapis. To lahko predstavlja omejitev, saj mora razčlenjevalnik najprej določiti imensko področje. Nato sledi kontrola veljavnosti XML sheme.
Besednjaki	-	Na spletni strani organizacije OASIS (OASIS, 2005) zasledimo večinoma besednjake, napisane za XML shemo.
Izpeljava tipov	-	Za XML sheme je značilna objektna naravnanost, saj omogočajo izpeljavo novih tipov objektov.
Ponovna uporabnost	-	Izpeljava novih tipov objektov omogoča ponovno uporabnost le-teh.
Funkcionalnost	Z vidika strukture DTD omogoča enake funkcionalnosti kot XML shema. V enostavnih primerih to povsem zadošča.	XML shema ponuja množico funkcionalnosti. Nekatere od njih v praksi niso splošno uporabne. Nove tehnologije bodo verjetno izkoristile večji delež možnosti, ki jih XML shema ponuja.
Razčlenjevalniki	Večina razčlenjevalnikov podpira DTD datoteke.	Določeni razčlenjevalniki ne podpirajo XML sheme. Določeni starejši sistemi morda ne podpirajo novejše tehnologije.
Imenska področja	-	XML shema podpira uporabo imenskih področij.
Omejitve in kontrole	V DTD datoteki so podatki predstavljeni zgolj kot niz znakov.	XML shema omogoča določitev vrednosti elementa (številka, datum). Poleg tega je mogoče natančneje določiti odnos med elementi (število pojavnosti določenega elementa).

Vir: Jurič, 2001, str. 15

2.3.3.1 SAX RAZČLENJEVALNIK

SAX (angl. Simple API for XML) je programski vmesnik (angl. application programming interface - API) za pregledovanje XML dokumenta. Temelji na dogodkih, ki se pojavljajo pri pregledovanju XML dokumenta. Programska oprema je na voljo v mnogih programskih jezikih: Java (Xerces), C++, Visual Basic, … Prednosti in slabosti SAX pregledovalnika prikazujeta tabela 2 in tabela 3.

Tabela 2: Prednosti SAX razčlenjevalnika

Prednost	Razlaga
Manjša poraba pomnilnika	SAX se je uveljavil kot standard pri obdelavi XML dokumentov. Za razliko od DOM modela SAX razčlenjevalnik obdeluje XML datoteko postopoma, kar pomeni, da se v pomnilnik naloži le manjši del vsebine datoteke. Zaradi tega je primeren tudi za obdelavo večjih količin podatkov ali velikega števila dokumentov, predvsem če ni potrebno spreminjati vsebine dokumenta. Ta ugotovitev je izredno pomembna pri spletnih aplikacijah, kjer je odjemalec lahko praktično kjerkoli in so pomembni odzivni časi aplikacije.
Delni pregled dokumenta	SAX razčlenjevalnik dopušča prekinitev obdelave dokumenta kadarkoli, zato je uporaben za iskanje posameznih podatkov, ki se nahajajo v dokumentu. Ko podatek najdemo, enostavno prekinemo nadaljnje iskanje. Prav tako je to uporabno pri ugotavljanju napak. Izvajanje obdelave lahko na ta način prekinemo ob prvi napaki.
Pridobivanje delov dokumenta	SAX razčlenjevalnik uporabimo, kadar želimo iz dokumenta izluščiti posamezne informacije. V ta namen namreč ni potrebno v pomnilnik naložiti celotnega dokumenta. Na ta način pripomoremo k odzivnosti sistema, saj znatno zmanjšamo porabo sistemskih virov.
Pretvorbe strukture	SAX razčlenjevalnik je uporaben pri določenih pretvorbah XML dokumenta, zlasti ko nas zanimajo podatki na višjih nivojih drevesa. V tem primeru vsi dodatni podatki iz DOM drevesa predstavljajo nepotrebno obremenitev sistema.

Vir: Rambhia, 2005; Jurič, 2001, str. 30

2.3.3.2 DOM RAZČLENJEVALNIK

DOM (angl. Document Object Model) za razliko od SAX razčlenjevalnika preoblikuje XML dokument v drevesno strukturo v pomnilniku. Zaradi tega je bolj potraten glede virov strojne opreme. Do posameznih podatkov (elementov) XML dokumenta nato dostopamo preko objektnega modela.

Programska oprema za obdelavo XML dokumentov z uporabo DOM razčlenjevalnika je bolj kompleksna kot pri uporabi SAX razčlenjevalnika.

Kot že omenjeno, pa lahko DOM razčlenjevalnik uporabimo za kontrolo celotnega dokumenta, branje posameznih elementov pa nadaljujemo s SAX razčlenjevalnikom.

Prednosti in slabosti DOM pregledovalnika prikazujeta tabela 4 in tabela 5.

Tabela 3: Slabosti SAX razčlenjevalnika

Slabost	Razlaga
Odkrivanje napak	V kolikor želimo odkriti vse napake v XML dokumentu, moramo pregledati celoten dokument, saj se preverjanje ustreznosti XML dokumenta izvaja sprotno, z branjem posameznih elementov. V določenih primerih lahko to predstavlja slabost.
Potrditev sprememb	V SAX razčlenjevalniku se preverjanje ustreznosti XML dokumenta izvaja sprotno, z branjem posameznih elementov. Če prepisujemo XML dokument v podatkovno bazo in naletimo na sredini dokumenta na napako, moramo pospraviti za seboj vse opravljene postopke (angl. rollback). Temu se lahko izognemo, če za samo kontrolo celotnega dokumenta uporabimo DOM razčlenjevalnik, ki ob branju dokumenta takoj vrne celotno množico napak.

Vir: Rambhia, 2005; Jurič, 2001, str. 30

Tabela 4: Prednosti DOM razčlenjevalnika

Prednost	Razlaga
XSL pretvorbe	DOM razčlenjevalnik deluje bolje pri XSL[3] (angl. Extensible Stylesheet Language) pretvorbah (XSLT - XSL Transformations), kjer XML dokument preoblikujemo na osnovi XSLT šablone. V primeru, ko želimo izdelati več različnih pogledov na iste podatke, uporabimo v ta namen več XSL šablon. Prav tako je potrebno formirati več DOM dreves. V prvem shranimo izvirni XML dokument, v ostalih pa preoblikovano vsebino.
Xpath poizvedbe	Pri uporabi zahtevnih Xpath[4] poizvedb, rezultat katerih so kompleksni XML elementi, je potrebno uporabiti DOM razčlenjevalnik, saj se drevesna struktura ohrani tudi v rezultatu poizvedbe.
Spremembe vsebine XML dokumenta	DOM razčlenjevalnik je namenjen izdelavi ali popravljanju, kot tudi branju XML dokumenta v pomnilnik. Na drugi strani je SAX razčlenjevalnik namenjen predvsem učinkovitemu branju XML dokumentov.

Vir: Rambhia, 2005; Jurič, 2001, str. 30

[3] Več o XSL pretvorbah v poglavju 2.3.4.

[4] Več o XPath jeziku za naslavljanje XML dokumentov v poglavju 3.3.2.

Tabela 5: Slabosti DOM razčlenjevalnika

Slabost	Razlaga
Velika poraba pomnilnika	DOM razčlenjevalnik preoblikuje XML dokument v drevesno strukturo v pomnilniku. Poraba pomnilnika pa je znatno večja od dejanske velikosti dokumenta. Primer: dokument velikosti 100 kilobyte-ov (KB) lahko porabi do 1 megabyte (MB) pomnilnika v primeru uporabe DOM razčlenjevalnika. V primeru uporabe SAX razčlenjevalnika je poraba pomnilnika znatno manjša.

Vir: Rambhia, 2005; Jurič, 2001, str. 30

2.3.4 PRETVORBE XML

XML podatki so sicer končnemu uporabniku berljivi, vendar nepregledni, saj so namenjeni predvsem računalniški obdelavi. V kolikor pa iste podatke zapišemo in predstavimo v uporabniku prijaznejši obliki, dodamo podatkom novo uporabno vrednost.

Zmožnost preoblikovanja XML podatkov v druge oblike dodatno prispeva k uporabnosti XML tehnologije. V ta namen je bila razvita tehnologija XSL, ki se je nato razvila v XSLT.

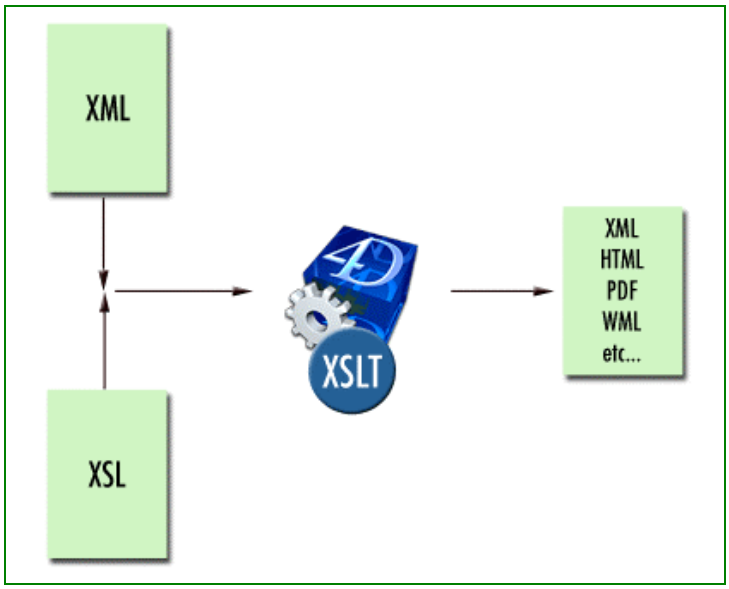
XSLT je na XML zapisu temelječ jezik, namenjen preoblikovanju in pretvorbi XML dokumentov. Slika 2 ponazarja postopek preoblikovanja, pri čemer izvirni XML dokument ostane nespremenjen. Rezultat pretvorbe je nov XML dokument, ki nastane na osnovi izvirnika in navodil za preoblikovanje. Nov dokument je v tem primeru lahko XML dokument s spremenjeno strukturo ali obliko, lahko je vsebina spletne strani v HTML zapisu, PDF datoteka ali običajna besedilna datoteka.

XSL sestavljajo trije deli:

- XSLT - jezik za pretvorbo XML dokumentov,
- XPath - jezik za naslavljanje XML dokumentov in
- XSL-FO – jezik za oblikovanje XML dokumentov

Večina spletnih brskalnikov že podpira pretvorbo XML dokumentov na osnovi XSLT (W3Schools/XSLT Browsers, 2005). To pomeni, da lahko težko berljiv XML dokument v spletnem brskalniku z uporabo XSL šablone preoblikujemo v običajno berljiv obrazec. Slika 3 prikazuje XML podatke o knjigah, ki bi jih radi predstavili na drugačen način (preoblikovali).

Slika 2: XSL pretvorba



Vir: AJAR, 2005

Slika 4 vsebuje navodila za preoblikovanje XML datoteke, t.i. XSL šablono. Z XSL šablono določimo način preoblikovanja XML dokumenta. Pri tem za vsak XML element/atribut določimo način pretvorbe. V prikazanem primeru XML podatke zapišemo v obliki tabele, zapisane v HTML jeziku in s tem podatkom dodamo še predstavitveno dimenzijo.

Slika 3: XML dokument z označeno XSL datoteko

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="knjige.xsl"?>
<Knjigarna>
  <Knjiga>
    <Naslov>Ime roze</Naslov>
    <Avtor>Umberto Eco</Avtor>
    <Zalozba>Vale Novak</Zalozba>
    <Cena>10500</Cena>
    <LetoIzdaje>1985</LetoIzdaje>
    <Strani>552</Strani>
  </Knjiga>

  <Knjiga>
    <Naslov>Pod svobodnim soncem</Naslov>
    <Avtor>Fran Saleski Finzgar</Avtor>
    <Zalozba>Mladinska knjiga</Zalozba>
    <Cena>1890</Cena>
    <LetoIzdaje>1985</LetoIzdaje>
    <Strani>570</Strani>
  </Knjiga>
</Knjigarna>
```

Slika 4: XSL datoteka

```
<?xml version="1.0" encoding="ISO-8859-1"?><xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"><xsl:template match="/">
  <html>
  <body>
    <h2>Ugodna ponudba knjig</h2>
    <table border="1">
    <tr bgcolor="#9acd32">
      <th align="left">Naslov</th>
      <th align="left">Avtor</th>
      <th align="left">Zalozba</th>
      <th align="left">Cena</th>
      <th align="left">Strani</th>
    </tr>
    <xsl:for-each select="Knjigarna/Knjiga">
    <tr>
      <td><xsl:value-of select="Naslov"/></td>
      <td><xsl:value-of select="Avtor"/></td>
      <td><xsl:value-of select="Zalozba"/></td>
      <td><xsl:value-of select="Cena"/></td>
      <td><xsl:value-of select="Strani"/></td>
    </tr>
    </xsl:for-each>
    </table>
  </body>
  </html>
</xsl:template></xsl:stylesheet>
```

Pregledovalnik nato opravi pretvorbo v ustrezno obliko. Pregledovalnik je v tem primeru lahko spletni brskalnik, dlančnik, mobilni telefon ali podobna naprava. V našem primeru je vsebina XML datoteke preoblikovana v HTML podatke, kar prikazuje slika 5.

Slika 5: V HTML zapis preoblikovana XML datoteka

Ugodna ponudba knjig

Naslov	Avtor	Zalozba	Cena	Strani
Ime roze	Umberto Eco	Vale Novak	10500	552
Pod svobodnim soncem	Fran Saleski Finzgar	Mladinska knjiga	1890	570

2.4. SPLETNE STORITVE

Spletne storitve (angl. web services) so samostojne, modularne programske rešitve, ki jih lahko poiščemo, objavimo in uporabimo na spletu (Gottschalk, 2005, str. 2).

Arhitektura spletnih storitev se je razvila iz objektno usmerjene analize in razvoja programskih rešitev. Spletne rešitve so tako podedovale temeljne značilnosti objektno usmerjenih sistemov, kot so ograjevanje (angl. encapsulation), posredovanje sporočil (angl. message passing), dinamično povezovanje (angl. dynamic binding) in opis storitve (angl. service description). Osrednja značilnost spletnih storitev, kot izhaja že iz imena, je storitvena usmerjenost, ki je na spletu na voljo drugim storitvam preko programskega vmesnika.

Spletne storitve lahko dinamično povezujemo v namenske aplikacije, ki so glede na naravo spletnih storitev neodvisne od programskega jezika, platforme ali komunikacijskega protokola.

2.4.1 MOTIVACIJA

Za tradicionalne monolitne informacijske sisteme je značilna močna navzkrižna povezanost med komponentami sistema in zaradi tega visoka občutljivost na spremembe. Sprememba obnašanja enega od podsistemov v takšnem primeru zaradi medsebojnega vpliva lahko povzroči izpad celotnega informacijskega sistema. Vzdrževanje takšnega okolja terja veliko virov, tako znanja kot ljudi. V takšnem primeru se podjetje težko odzove na spremembe v konkurenčnem okolju.

Sodobne informacijske rešitve torej zahtevajo enostavno prilagodljivost na spremembe. Spletne storitve v tem smislu prinašajo za podjetje številne prednosti. Glavne od njih vsebuje tabela 6.

2.4.2 ARHITEKTURA

V arhitekturi spletnih storitev nastopajo tri ključni akterji (slika 6):

1. **Ponudnik spletne storitve**
   Ponudnik spletne storitve objavi svojo storitev bodisi z namenom zaslužka ali sodelovanja s partnerji. Svojo storitev opiše na standardiziran način in jo objavi v temu namenjenem imeniku.

2. **Uporabnik spletne storitve**
   Uporabnik spletne storitve želi uporabiti določeno storitev. Prvi korak je iskanje ustrezne storitve, ki zadošča poslovnim potrebam. V kolikor najdena storitev ustreza, se uporabnik poveže s ponudnikom. Oba, ponudnik in uporabnik spletne storitve, morata uporabiti enak protokol za dostop do spletne storitve.

3. **Imenik storitev**
   Imenik storitev ponuja posredništvo in hrambo, kjer so objavljeni opisi spletnih storitev. Na tem mestu se srečujeta ponudba in povpraševanje po spletnih storitvah. Ponudniki na tem mestu objavijo svoje storitve, uporabniki po njih povprašujejo.

   Imenik mora zagotoviti ustrezno klasifikacijo spletnih storitev, glede na njihov opis in funkcionalnost.

Tabela 6: Lastnosti spletnih storitev

Prednost	Opis
Tehnološka neodvisnost	Spletne storitve so neodvisne od razvojnega okolja (NET, CORBA, J2EE), kar omogoča njihovo uporabo na različnih infrastrukturah. Ta lastnost omogoča hitro povezavo z obstoječimi programskimi rešitvami.
Hitra integracija	Povezovanje na spletno storitev poteka v času klica storitve. Potrebna je le prilagoditev na vmesnik spletne storitve, opisan v WSDL.
Prilagodljivost	Zamenjava spletne storitve z enako funkcionalnostjo je enostavna, saj poteka povezovanje v času klica storitve. Potrebna je le prilagoditev na nov vmesnik.
Nadgradljivost	Kompleksnost in obnašanje spletne storitve je uporabniku skrito, zato je funkcionalnost storitve nadgradljiva z novimi lastnostmi brez dodatnih posegov.
Enostavnost uporabe	Vse spletne storitve se obnašajo kot storitve. Med seboj se razlikujejo le po dostopu do storitve in načinu uporabe. Ograjevanje nam prihrani vso kompleksnost in logiko storitve, saj se storitev obnaša kot črna skrinjica. Storitev zahteva le določene vhodne podatke. Uporabnik storitve je zgolj njen uporabnik in se ne ubada z njenim delovanjem in kompleksnostjo.
Ponovna uporabnost	Razvijalcem je na voljo širok izbor spletnih storitev, ki lahko na ta način modularno razvijajo poslovne aplikacije.
Zaščita investicije	Storitvena naravnanost spletnih storitev omogoča enostavno zamenljivost komponent brez negativnega vpliva na stranke in poslovni proces.
Globalizacija	Dostop do spletne storitve in imenika je mogoč kjerkoli.

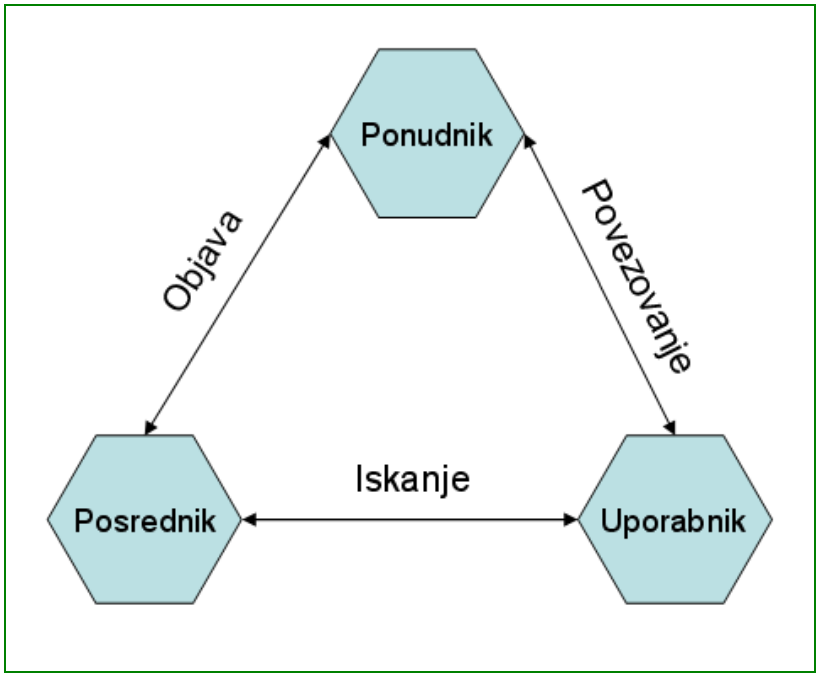
Vir: Gottschalk, 2005, str. 4

Pred uporabo spletne storitve je potrebno opraviti naslednje postopke:

- spletno storitev je potrebno izdelati in določiti njene vmesnike ter način uporabe;
- spletno storitev je potrebno objaviti v ustreznem imeniku, kjer jo potencialni uporabniki lahko poiščejo;
- spletno storitev naj bo mogoče preklicati, ko ni več potrebna ali razpoložljiva;
- arhitektura spletnih storitev zahteva tri osnovne operacije: objavo, iskanje in povezovanje (slika 6). Ponudnik storitve objavi svojo storitev v imeniku

(posredniku). Uporabnik storitve poišče zahtevano storitev v imeniku. Posrednik se na osnovi informacij iz imenika poveže s ponudnikom.

Slika 6: Arhitektura spletnih storitev



Vir: Gottschalk, 2005, str. 3

2.4.3 STANDARDI

Uspešnost tehnologij je običajno povezana s standardi, saj le-ti zagotavljajo enotno uporabo tehnologije. To je še posebej pomembno pri elektronskem poslovanju, kjer v poslovnem procesu sodeluje veliko število partnerjev.

2.4.3.1 OVOJNICA

SOAP (angl. Simple Object Access Protocol) določa model, s katerim elektronska sporočila, zapisana v XML obliki, ovijemo z ovojnico. SOAP model izmenjave sporočil lahko uporabimo za poljubno elektronsko izmenjavo podatkov. Tabela 7 našteva XML elemente, ki sestavljajo SOAP ovojnico.

SOAP je nadgradljiv komunikacijski protokol za posredovanje sporočil med spletnimi aplikacijami, neodvisen od operacijskega sistema, programskega jezika in različnih tehnologij.

Tabela 7: Elementi SOAP ovojnice

Element	Opis
<Envelope>	Obvezen korenski element, ki določa XML dokument kot SOAP sporočilo.
<Header>	Neobvezen element, ki opredeljuje posebnosti aplikacije, kot so avtentikacija ali plačilo. Če element obstaja, se mora pojaviti kot prvi podrejeni element elementa »Envelope«.
<Body>	Obvezen element, ki predstavlja telo ovojnica in vsebuje klic spletne storitve in ustrezen odgovor.
<Fault>	Neobvezen element, ki vsebuje informacije o napakah, ki so se zgodile pri obdelavi sporočila.

Vir: W3Schools/SOAP tutorial, 2005

2.4.3.2 OPIS

WSDL (angl. Web Services Description Language) opredelimo kot standardni jezik za opis spletnih storitev, zapisan v XML obliki. Namenjen je opisu storitev, vmesnikov in protokolov za dostop do storitev. WSDL je neodvisen od tehnološkega okolja, kar izhaja iz njegove XML usmerjenosti. Tabela 8 podrobneje opisuje elemente WSDL dokumenta.

Tabela 8: Elementi WSDL dokumenta

Element	Opis
<portType>	Element določa spletno storitev, aktivnosti, ki se zgodijo in sporočila, ki se prenašajo med sistemi. Podatek je primerljiv s programsko knjižnico v običajnih programskih jezikih.
<message>	Element opredeljuje sporočila, ki jih uporablja spletna storitev. Vsako sporočilo lahko sestavlja več delov. Vsak del lahko primerjamo s parametri funkcije pri običajnem programiranju.
<types>	Podatek opredeljuje podatkovne tipe, ki jih uporablja spletna storitev. Zaradi boljše prenosljivosti se za definicijo tipov uporablja skladnja XML sheme.
<binding>	Podatek opredeljuje komunikacijski protokol, ki ga uporablja spletna storitev. Prav tako določa obliko sporočil.

Vir: W3Schools/WSDL tutorial, 2005

V primeru, ki ga prikazuje slika 7, predstavlja podatek »Zaloge« knjižnico, ki se uporablja v povezavi z zalogami, »getZaloga« pa ime spletne storitve. Ta vsebuje vhodno sporočilo »getZalogaZahtevek« in izhodno sporočilo »getZalogaOdgovor«.

Element »message« določa podatkovne tipe v sporočilu. V prikazanem primeru ima funkcija »getZalogaZahtevek« parameter »Sifra«, ki je niz znakov, funkcija »getZalogaOdgovor« pa vrača količino izdelka na zalogi.

Slika 7: Primer dela WSDL dokumenta

```
<message name="getZalogaZahtevek">
   <part name="Sifra" type="xs:string"/>
</message>

<message name="getZalogaOdgovor">
   <part name="Kolicina" type="xs:string"/>
</message>

<portType name="Zaloge">
     <operation name="getZaloga">
          <input message="getZalogaZahtevek"/>
          <output message="getZalogaOdgovor"/>
     </operation>
</portType>
```

2.4.3.3 IMENIK

UDDI (angl. Universal Description, Discovery and Integration) predstavlja ogrodje za opis ter iskanje spletnih storitev. Predstavlja imenik spletnih storitev in si ga lahko predstavljamo kot globalne rumene strani.

UDDI omogoča objavo spletnih storitev, zapisano v WSDL standardu, preko katerih se podjetje pojavi na tržišču kot ponudnik spletnih storitev.

Uporabniki ali iskalci določenih spletnih storitev lahko v tem imeniku izvajajo poizvedbe, zapisane po SOAP specifikaciji in na ta način odkrijejo ali izberejo ustreznega ponudnika storitve.

Pred nastankom UDDI ni bilo enotnega standarda, preko katerega bi podjetja lahko ponudila svoje storitve ali izdelke. Prednosti, ki jih prinaša UDDI:

- iskanje ustrezne storitve med množico ponudnikov na spletu,
- trženje lastnih storitev in izdelkov,
- odkrivanje novih poslovnih partnerjev in
- enoten način komunikacije med partnerji.

3. RAZLIKE MED XML TEHNOLOGIJO IN RELACIJSKIMI PODATKOVNIMI BAZAMI

Že v uvodu sem predstavil osnovne razlike med svetovoma relacijskih podatkovnih baz in XML tehnologije. To poglavje obravnava te razlike podrobneje. V poglavju podrobneje obravnavam razlike med podatkovnimi modeli, načine integracije XML tehnologije v relacijske podatkovne baze ter razlike med poizvedovalnimi jeziki.

Tabela 9 opisuje osnovne razlike med funkcionalnostmi, ki jih ponujata obe tehnologiji.

Tabela 9: Primerjava funkcionalnosti

Funkcionalnost	Relacijske podatkovne baze	XML
Podpora transakcijam	Sistemi za upravljanje baz podatkov podpirajo transakcije in možnost preklica sprememb (angl. rollback support). Določeni sistemi za upravljanje baz podatkov imajo podporo za dvofazno potrjevanje (angl. two-phase commit) tudi preko več podatkovnih virov.	Tehnologija XML sama po sebi ne ponuja transakcijskega načina dela. Novejši XML sistemi za upravljanje baz podatkov že ponujajo transakcijski način dela.
Večuporabniško delo, zaklepanje zapisov	Za zaščito integritete podatkov sistemi za upravljanje baz podatkov ponujajo mehanizme zaklepanja podatkov. Pri tem gre lahko za zaklepanje na nivoju strani (angl. page-level) ali zapisov (angl. row-level).	Dostop do XML dokumentov poteka preko datotečnega sistema, ki ne ponuja tovrstnih funkcionalnosti kot podatkovne baze.
Prenosljivost	Relacijski sistemi za upravljanje baz podatkov so tržni izdelki različnih proizvajalcev, ki shranjujejo podatke na različne načine. Podatke med posameznimi izdelki lahko prenašamo s pomočjo temu namenjenih orodij.	XML specifikacija temelji na odprtih industrijskih standardih in je zato prenosljiva med posameznimi platformami.
Podatkovna shema	Podatkovna shema in njene funkcionalnosti so obvezen del sistema za upravljanje baz podatkov.	Podatkovna shema je neobvezen del XML dokumenta. Za uporabo podatkovne sheme obstaja več načinov uporabe (DTD datoteka, XML shema).
Ponovna uporaba sheme	Podatkovna shema je pripeta k sistemu za upravljanje baz podatkov. Ponovna uporaba (izvoz) podatkovne sheme je pogojen z uporabo orodij za modeliranje.	Podatkovna shema se nahaja v ločenem dokumentu, neodvisno od podatkov, pri čemer lahko množica različnih dokumentov uporablja isto podatkovno shemo.
Gnezdenje podatkov	Relacijski sistemi za upravljanje baz podatkov na splošno ne podpirajo gnezdenja podatkov v stolpcih tabel.	XML elementi lahko vsebujejo gnezdene (podrejene) elemente.

Vir: Erl, 2004, str. 241

3.1. PODATKOVNI MODEL

V praksi so se na področju zajema podatkov oblikovali trije prevladujoči tipi sistemov za upravljanje z bazami podatkov:

- hierarhični,
- relacijski in
- objektno orientirani.

Poleg njih srečamo še večdimenzionalni podatkovni model na področju analitičnega odločanja.

Hierarhične baze se pojavijo okrog leta 1960, v času obdelave podatkov na centralnem računalniku. V sedemdesetih letih preteklega stoletja se kot rezultat raziskovalnega dela E. F. Codd in drugih pojavijo relacijske baze podatkov. Desetletje kasneje, z razmahom objektno orientiranih programskih jezikov, doživijo svoj razcvet še objektno orientirane baze podatkov (Connoly, 1998, str. 25, 750).

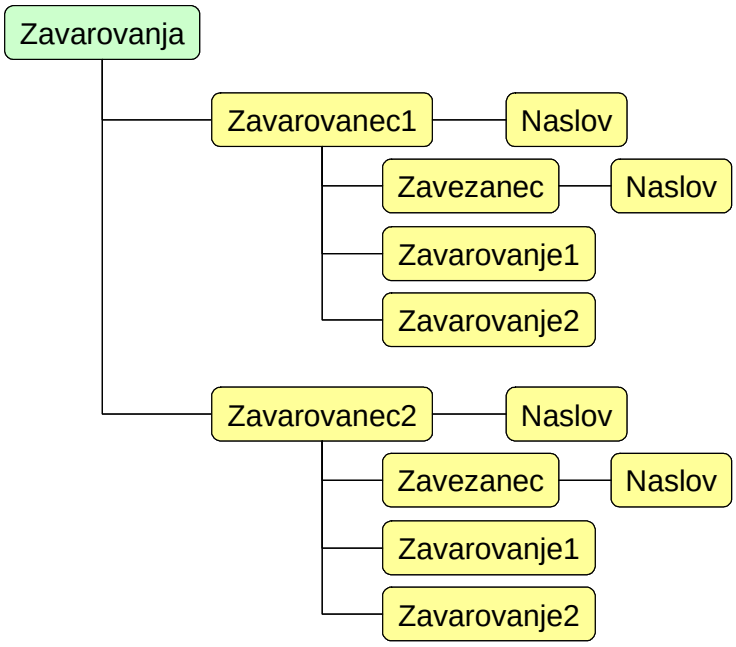
Trenutno zavzemajo na tržišču levji delež med bazami podatkov relacijske baze podatkov. Hierarhične in objektne baze zapolnjujejo zlasti tržne niše na specializiranih in ozko usmerjenih segmentih. Vendar lahko v zadnjih letih opazimo določeno konvergenco relacijskega in objektnega sveta, saj so določene relacijske baze vstopile v svet t.i. hibridnih relacijsko-objektnih podatkovnih baz in privzele določene objektno usmerjene lastnosti in funkcionalnosti. Meje med relacijskim in objektno orientiranim svetom torej počasi bledijo (Mertz, 2001, str. 1).

3.1.1 HIERARHIČNI MODEL

Hierarhično bazo podatkov tvori strogo določeno drevo podatkov, začenši s korenskim vozliščem (angl. root node), sestavljeno iz vozlišč (angl. nodes). Vsako vozlišče lahko poleg podatkov vsebuje tudi podrejena vozlišča v določeni sestavi in zaporedju. Pri tem lahko število podrejenih vozlišč zavzame poljubno vrednost (Mittra, 1991, str. 167). Vsako vozlišče, razen korenskega, je povezano z natanko enim nadrejenim vozliščem in tvori t.i. razmerje starš/otrok (angl. parent/child). Teoretični hierarhični podatkovni model, kjer so predstavljeni podatki o zavarovanjih, predstavlja slika 8.

V hierarhičnih bazah podatkov je pristop do podatkov natančno določen z drevesno strukturo. Tako branje kot spreminjanje podatkov sta visoko optimirana.

Slika 8: Teoretični hierarhični podatkovni model



Do določenega podatka v hierarhični strukturi dostopimo tako, da navedemo njegovo pot preko vozlišč. Naslov zavarovanca lahko tako ugotovimo na naslednji način s pseudo poizvedovalnim jezikom :

```
Zavarovanja/Zavarovanec1/Naslov?
```

Za razliko od relacijskih, hierarhične podatkovne baze ne ponujajo možnosti pristopa do podatkov preko proceduralnih poizvedovalnih jezikov, kot je SQL, kar predstavlja slabost. Pristop do podatkov zahteva poznavanje programskih jezikov (Connoly, 1998, str. 53; Mittra, 1991, str. 167).

Poznavalci tehnologije XML lahko prepoznajo določene značilnosti naslovnega jezika XPath (XML Path Language). Model je primerljiv tudi z dokumentnim modelom DOM (Document object model), ki ga obravnavam v nadaljevanju.

Tipičen predstavnik hierarhičnih podatkovnih baz je (bil) IMS, podjetja IBM.

Glavna slabost, ki jo lahko očitamo hierarhičnim podatkovnim bazam, je pomanjkanje omejitev na podatkih, kot je npr. referenčna integriteta, ki je podprta le do določene mere.

3.1.2 RELACIJSKI MODEL

Relacijski model tvori množica tabel, sestavljena iz stolpcev (polj). Vsaka tabela vsebuje zapise, ki tvorijo vsebino tabele. Običajno tabela vsebuje enolični identifikator, t.i. glavni ključ tabele, po katerem so sortirani zapisi v tabeli. Že omenjene podatke o zavarovanjih in zavarovancih, zapisane v relacijskem podatkovnem modelu, prikazuje slika 9.

Lastnosti, značilne za tabele v relacijskem modelu:

1. vrednosti so atomarne (v vsakem stolpcu je le en podatek),
2. vsaka vrstica je enolična,
3. vrednosti v stolpcih so istovrstne,
4. zaporedje vrstice v tabelah ni pomembno in
5. vsak stolpec ima enolično ime.

Slika 9: Teoretični relacijski podatkovni model

Številka osebe	Priimek	Ime	EMŠO
001	Novak	Janez	1910960500223
002	Miklavc	Mojca	2403965505200

Številka subjekta	Ulica	Pošta	Kraj
001	Trubarjeva 20	1000	Ljubljana
002	Prešernova 3	4000	Kranj
101	Dunajska 50	1000	Ljubljana
102	Morska 20	5000	Koper

Številka zavezanca	Naziv zavezanca	Davčna številka
101	Petrol d.d.	22222222
102	Droga d.o.o.	33333333

Številka zavarovanja	Številka osebe	Številka zavezanca	Vrsta zavarovanja	Začetek zavarovanja
1	001	101	01	01.01.2005
2	001	101	02	01.07.2005
3	002	102	03	01.02.2005

Poleg glavnega ključa tabele vsebujejo tabele še tuje ključe, ki odgovarjajo ključem ostalih tabel. Polje ŠTEVILKA ZAVEZANCA (slika 9) v tabeli ZAVAROVANJA predstavlja tuji ključ, ki je istočasno ključ tabele ZAVEZANCI.

Relacijski model ponuja mehanizme, ki odražajo določena poslovna pravila. Eden od takšnih mehanizmov je referenčna integriteta. Z referenčno integriteto zagotavljamo in vzdržujemo odnose med tabelami. Primer referenčne integritete je pravilo, da v polje ŠTEVILKA ZAVEZANCA v tabeli ZAVAROVANJA ne moremo vnesti šifre, ki ne obstaja v tabeli ZAVEZANCI.

Načrtovanje tabel v relacijskem modelu utegne biti kompleksno opravilo, katerega glavni del predstavlja ustrezna normalizacija tabel. Postopek normalizacije zajema več korakov od prve do pete normalne forme, z njim pa skušamo odpraviti nepotrebne podvojene podatke.

Relacijski model je v primerjavi s hierarhičnim modelom znatno bolj zapleten. Na drugi strani pa je neprimerno zmogljivejši glede pristopa do podatkov. S strukturiranimi poizvedovalnimi jeziki, kot je SQL lahko pridemo do večine informacij.

S poizvedbo, ki jo prikazuje slika 10, lahko zelo enostavno poiščemo vse zavarovance, ki prebivajo v Ljubljani.

Slika 10: Primer poizvedbe v SQL poizvedovalnem jeziku

```
SELECT o.Priimek, o.Ime
FROM   Osebe o, Naslovi n
WHERE  o.Številka_osebe = n.Številka_subjekta
AND    n.Pošta = 1000
```

3.1.3 POLSTRUKTURIRAN MODEL

Razvoj spleta je vodil k izdelavi nove generacije aplikacij, za katere relacijski podatkovni model ni bil več primeren. Spletni brskalnik postaja osnovni pripomoček pri iskanju podatkov. Podatki so zapisani v različnih oblikah (HTML, XML, ASCII, relacijski). V ta namen je bil razvit polstrukturiran podatkovni model, ki bolje ustreza današnjim potrebam.

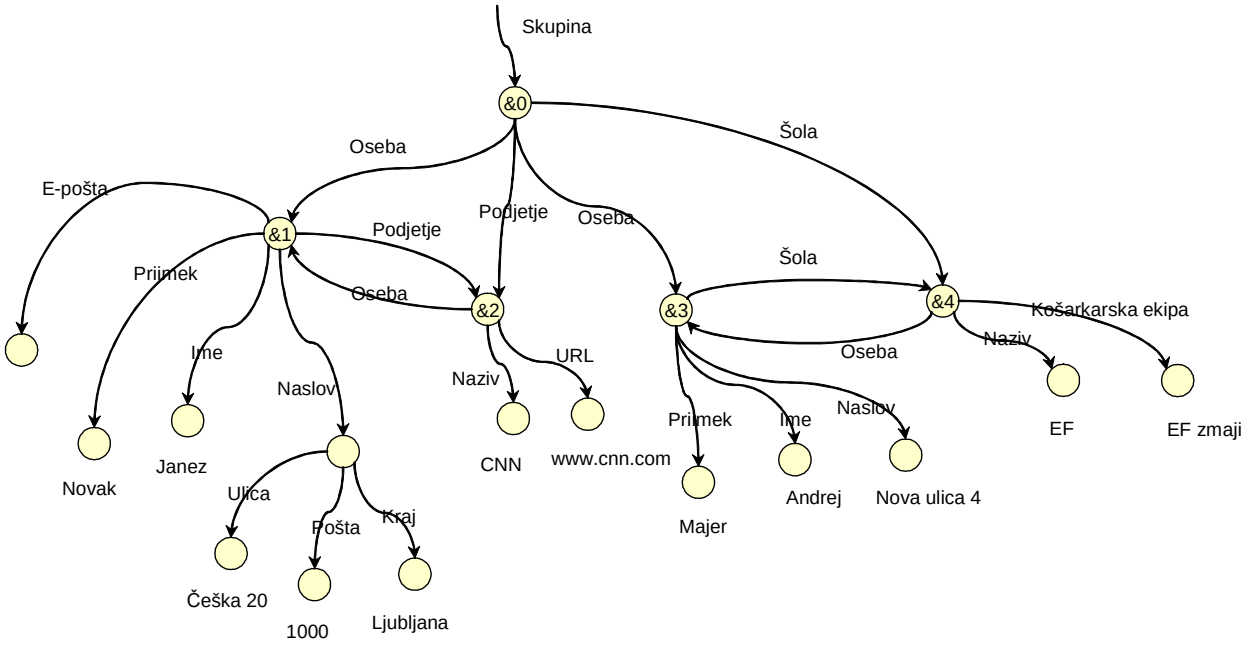
Za polstrukturirane podatke velja (Nativi, str. 8):

- na splošno so gnezdeni, nepravilne strukture in raznolike vsebine,
- struktura se razvija ter
- vsebujejo povezave na podatke, shranjene na različnih virih.

Izraz polstrukturiran podatkovni model večinoma uporabljamo, kadar podatki niso organizirani po vnaprej določeni shemi ali kadar je struktura podatkov nepopolna.

V določenih primerih pa je struktura podatkov določena po shemi, ki je ločena od podatkov. V takšnem primeru shema predpisuje obliko in hierarhijo podatkov (Barbosa, str. 2).

Slika 11: Teoretični polstrukturiran podatkovni model



Polstrukturiran podatkovni model običajno predstavljamo v obliki grafa. Presečišča oz. vozlišča v grafu predstavljajo objekte ali vrednosti (slika 11). Povezave med vozlišči nam povedo imena atributov ter lahko povezujejo objekte med seboj ali objekte z njihovimi lastnostmi.

Polstrukturirani podatki v zadnjem času pridobivajo na veljavi. Razlogov za to je več. Za podatke na spletu velja, da so na splošno slabo strukturirani in bogate vsebine. Na drugi strani prodira elektronsko poslovanje v vse pore tako potrošnikov, kot podjetij, kar narekuje uporabo lahko prilagodljivega standarda za izmenjavo podatkov. Še tretji razlog pa je povezovanje obstoječe programske opreme znotraj podjetja ali med podjetji. Na tem področju podjetja vlagajo krepka finančna sredstva, projekti integracije pa spadajo med tehnično zahtevnejše projekte.

Omenim naj še Object exchange model (OEM), ki je tipičen predstavnik polstrukturiranih podatkovnih modelov.

OEM objekte lahko razdelimo v dve skupini, glede na njihove vrednosti (Tae-Sun, 2003, str. 226 - 228):

- preprosti (atomarni) objekti: vrednost preprostih objektov vsebuje le posamezno vrednost, kot npr.: niz, številko, sliko, ipd…,
- kompleksni, sestavljeni objekti (slika 12).

Slika 12: Sestavljen objekt po OEM notaciji (Goldman, 2005)

```
<DatumRojstva{
     <Dan 16>
     <Mesec "Januar">
     <Leto 2000>}>
```

Slika 12 prikazuje sestavljen objekt »DatumRojstva«, ki ga sestavljajo trije preprosti (atomarni) objekti.

3.1.4 XML KOT MODEL

Če primerjamo omenjene podatkovne modele in XML lahko ugotovimo, da je XML nekakšen hibrid med podatkovnimi modeli. Po svoji strukturi je najbolj podoben polstrukturiranemu modelu, sestavljenemu iz vozlišč in raznolikih podatkov.

Na drugi strani pa raznoliko strukturo dokumenta omejuje DTD datoteka ali XML shema, ki natančno predpisuje njeno strukturo. V takšnem primeru je lahko XML dokument tako natančno predpisan kot zapisi v hierarhični podatkovni bazi. Kot standard za izmenjavo podatkov pa je še vedno dovolj odprt in bogat, da lahko z njim predstavimo podatkovno plat objektov.

V naslednjih odstavkih povzemam vsebino tega poglavja. Zanimajo me predvsem razlike med relacijskim in polstrukturiranim podatkovnim modelom. Tabela 10 povzema primerjavo podatkovnih elementov med relacijskim in polstrukturiranim podatkovnim modelom.

XML model v primerjavi z relacijskim ne vsebuje mehanizmov, ki zagotavljajo razmerja med podatki. Razlike na področju kontrole podatkov med obema modeloma podaja tabela 11.

Iz povedanega je razvidno, da se tehnologija XML ne vključuje najbolje v svet relacijskih baz podatkov. Še najmanjšo težavo predstavlja predstavitev podatkov iz relacijske baze podatkov v XML obliki.

Osnovo relacijske sheme sestavljajo omejitve, ki določajo razmerja med tabelami. Prav ta funkcionalnost predstavlja poglavitno prednost relacijskega podatkovnega modela. Robustnost in učinkovitost relacijskih podatkovnih baz, združeni s prilagodljivostjo in odprtostjo XML tehnologije, predstavljata dobro osnovo za razvoj aplikacij elektronskega poslovanja.

Tabela 10: Primerjava podatkovnih elementov

	Relacijske baze podatkov	XML
Podatkovni model	Relacijski podatkovni model, sestavljen iz tabel, vrstic in stolpcev.	Vsebino polstrukturiranega podatkovnega modela tvori dokument drevesne hierarhične strukture, ki vsebuje elemente in atribute.
Podatkovni tipi	Podatkovni tipi so vnaprej določeni. Podpora binarnim podatkom.	XML shema ponuja večino podatkovnih tipov poznanih iz relacijskih podatkovnih baz. Omejena podpora binarnim podatkom.
Razmerje med podatki	Relacijski model temelji na povezavami med entitetami. Povezave so lahko implementirane kot razmerja med stolpci znotraj ene tabele ali med tabelami.	Razmerja med podatki so lahko neposredno podana.
Zaporedje	Zaporedje stolpcev v tabeli ni pomembno.	Zaporedje elementov in atributov je predpisano.

Vir: Erl, 2004, str. 238; Champion, 2001, str 53.

Tabela 11: Primerjava kontrol podatkov

	Relacijske baze podatkov	XML
Shema	Ohlapna struktura relacijske sheme omogoča poljubno opredelitev odnosov med entitetami in njihovega obstoja.	V kolikor za kontrolo XML podatkov uporabimo XM shemo velja, da je XML shema v tem pogledu bolj stroga, saj opredeljuje hierarhijo XML elementov/atributov.
Referenčna integriteta	Dodatna podpora je predvidena zagotavljanju omejitev pri razmerjih med entitetami. Običajne tovrstne funkcionalnosti vključujejo prenos sprememb med odvisnimi objekti, kot so stopničasto brisanje ali spreminjanje.	Polstrukturiran podatkovni model ne vsebuje mehanizmov, kot je referenčna integriteta. Določene mehanizme prinašajo XML razčlenjevalniki, s pomočjo katerih lahko preverjamo strukturo XML dokumenta.

Vir: Erl, 2004, str. 239

3.2. INTEGRACIJA XML DOKUMENTOV V PODATKOVNIH BAZAH

Z razvojem spleta in elektronskega poslovanja narašča tudi množičnost uporabe XML tehnologije. S tem pa se pojavi potreba po upravljanju takšnih podatkov. V ta namen običajno uporabimo ustrezen sistem za upravljanje baz podatkov. Le-te pa so bolje ali slabše prilagojene XML tehnologiji. Na tržišču srečamo dve vrsti sistemov za upravljanje baz podatkov, ki ju podrobneje obravnavam v tem poglavju:

- baze podatkov z delno podporo tehnologiji XML (angl. XML enabled database) in
- baze podatkov, kjer je tehnologija XML popolnoma integrirana (angl. native XML databases).

3.2.1 USMERJENOST XML DOKUMENTOV

Pri načrtovanju informacijskih rešitev je v osnovi potrebno določiti vlogo relacijske podatkovne baze pri uporabi XML tehnologije. Vloga in izbira relacijske podatkovne baze je odvisna od poslovnih potreb. Možna vprašanja, ki si jih moramo zastaviti, so sledeča:

- Ali nameravamo javno objaviti obstoječe podatke, ki se že nahajajo v podatkovnih zbirkah?
- Ali vpeljujemo model elektronskega poslovanja med podjetji (B2B), kjer je potrebno hraniti izvirne XML dokumente?
- Ali želimo v XML obliki hraniti vsebino strani našega spletnega mesta?
- Ali se XML tehnologija uporablja zgolj kot transport za prenos podatkov med virom podatkov in aplikacijo?

Na osnovi teh vprašanj lahko izluščimo dva tipa XML dokumentov:

- podatkovno usmerjene XML dokumenti (angl. data-centric documents) in
- dokumentno usmerjene XML dokumente (angl. document-centric documents).

V praksi razlika med obema oblikama XML dokumentov ni popolnoma jasna. Tako lahko npr. pretežno podatkovno usmerjen XML dokument, kot je račun, vsebuje bogat opis postavk in materiala. In nasprotno, pretežno dokumentno usmerjen XML dokument, kot je uporabniški priročnik, lahko vsebuje drobno strukturirane podatke, kot so podatki o izdaji in avtorju.

Izbira tipa XML dokumenta bistveno vpliva na način shranjevanja XML dokumentov v relacijskih podatkovnih bazah. Zato je smiselno podrobneje opredeliti lastnosti obeh tipov dokumentov (Bourret, 2004, str. 4).

3.2.1.1 PODATKOVNO USMERJENI XML DOKUMENTI

Za podatkovno usmerjene dokumente je značilno, da služijo kot prenosni medij za transport podatkov. Tovrstne dokumente odlikuje tudi dobra in stabilna strukturiranost ter dobra granularnost.

Primeri takšnih dokumentov so naročilnice, rezervacije letalskih poletov, računi, podatki za nakup vrednostnih papirjev, ipd… Vir takšnih dokumentov so lahko bodisi obstoječi podatki, ki jih nameravamo objaviti v XML obliki, bodisi XML dokumenti, ki jih želimo shraniti v podatkovno zbirko. Slika 13 prikazuje podatke o zavarovancu. Ker posamezne podatke iz dokumenta zlahka razberemo, lahko opredelimo dokumenta kot dobro strukturiran.

Slika 13: Podatkovno usmerjen XML dokument

```
<Zavarovanec>
     <Emso>2907970505416</Emso>
     <Drzavljanstvo>705</Drzavljanstvo>
     <Priimek>Novak</Priimek>
     <Ime>Klavdija</Ime>

     <StalniNaslov>
          <Ulica>Litijska 41</Ulica>
          <SifraPoste>1000</SifraPoste>
          <KrajPoste>Ljubljana</KrajPoste>
          <SifraObcine>01</SifraObcine>
          <ImeObcine>Ljubljana</ImeObcine>
          <SifraDrzave>705</SifraDrzave>
          <ImeDrzave>Slovenija</ImeDrzave>
     </StalniNaslov>
</Zavarovanec>
```

Podatkovno usmerjeni dokumenti so namenjeni avtomatski obdelavi, brez ročnih posegov uporabnika.

3.2.1.2 DOKUMENTNO USMERJENI XML DOKUMENTI

Primere dokumentno usmerjenih XML dokumentov najdemo v elektronskih sporočilih, elektronskih knjigah, oglaševanju. Za njih je značilna slabša strukturiranost in bogatejša vsebina. Dober primer dokumentno usmerjenega XML dokumenta predstavlja RSS tehnologija (angl. Really Simple Syndication, tudi Rich Site Summary), katere primer prikazuje slika 14. V tem primeru gre za objavo datotek bogate vsebine, zapisanih v XML jeziku, ki so običajno objavljene na spletnih straneh.

Slika 14: RSS datoteka kot dokumentno usmerjen XML dokument

```
<channel>
<title>RTV - Šport</title>
<link>http://www.rtvslo.si/rss/</link>
<description>novice...</description>
<category>Šport</category>
<item>
<title>Slovenci obstali med elito</title>
<pubDate>Wed, 11 May 2005 22:27:53 +0200</pubDate>
<description> <![CDATA[ <P><STRONG><IMG height=210 alt="Veselje slovenskih
hokejistov" src="http://www.rtvslo.si/upload/sport/hokej/haki_show.jpg"
width=280 align=right border=1 subtitle="epa|Rise bomo gledali tudi drugo
leto v Rigi.">Slovenski hokejisti so po zmagi s 6:2 nad Avstrijo obstali
med 16 najboljšimi moštvi na svetu, ki bodo drugo leto igrala v
Latviji.<BR><BR></STRONG>Risi so se po zaostanku z 0:1 prebudili v drugi
tretjini. Najprej so prvič na prvenstvu zadeli z igralcem več (Ivo Jan),
nato pa sta svoja prva gola v Avstriji dosegla še Marcel Rodman in Tomaž
Razingar. <BR></P>  ]]>
</description>
</item>
```

Vir: RTV Slovenija, 2005

Dokumenti te vrste običajno nastanejo kot posledica pretvorb iz drugih dokumentnih oblik, kot so RTF ali PDF, v XML obliko in običajno ne izvirajo iz obstoječih podatkovnih baz.

3.2.2 PODPORA TEHNOLOGIJI XML V PODATKOVNIH BAZAH

Nivo podpore tehnologiji XML v podatkovnih bazah tako deli le-te v dve skupini. V prvi skupini najdemo baze podatkov z delno podporo tehnologiji XML (angl. XML enabled database). Drugo skupino predstavljajo baze podatkov, kjer je tehnologija XML popolnoma integrirana (angl. native XML databases).

3.2.2.1 PODATKOVNE BAZE Z DELNO PODPORO TEHNOLOGIJI XML

O bazah podatkov z delno podporo tehnologiji XML govorimo takrat, ko so obstoječe funkcionalnosti baze nadgrajene z določenimi možnosti podpore tehnologiji XML. To kategorijo običajno sestavljajo relacijske baze podatkov.

Uporaba relacijskih podatkovnih baz v te namene je razumljiva, če upoštevamo količino obstoječih aplikacij in podatkov (Kappel, 2004, str. 344). Dodaten razlog je zahteva po objavi obstoječih informacij, ki so zajete v podatkovnih zbirkah. Zaradi že omenjenih razlik pa integracija XML dokumentov v baze podatkov ni enostavna.

Splošen razmah uporabe tehnologije XML v elektronskem poslovanju narekuje tudi razvoj relacijskih podatkovnih baz na tem področju. Izzivi, ki jih prinaša tehnologija XML v svet relacijskih podatkovnih baz, so predvsem (Balmin, 2005, str. 31):

- Kako pretvoriti XML dokument v relacijski model in ga shraniti v relacijsko podatkovno bazo? Postopek imenujemo tudi dekompozicija (angl. decomposition).

- Kako pretvoriti podatke iz relacijskega modela v polstrukturiran podatkovni model in sestaviti XML dokument iz relacijske podatkovne baze? V nadaljevanju za ta postopek uporabljam izraz kompozicija (angl. composition).

- Kako učinkovito pretvoriti poizvedbe po XML dokumentu v SQL stavek, na osnovi katerega nato pridobimo podatke v XML obliki?

- Kako shraniti izviren XML dokument v relacijsko podatkovno bazo in omogočiti običajno delo z dokumentom (manipulacija s podatki, iskanje)?

V tem primeru imamo običajno opravka s pretvorbo polstrukturiranega v relacijski podatkovni model. Z normalizacijo razporedimo elemente in atribute po tabelah in stolpcih, pri čemer vsak stolpec vsebuje le eno (atomarno) vrednost. Zahtevnost postopka preslikave XML dokumenta v relacijski model je odvisna od kompleksnosti XML dokumenta. Na splošno velja, da kompleksnost preslikave narašča s razvejanostjo XML dokumenta (Cincom, 2003, str. 3).

Relacijske baze podatkov z delno podporo tehnologiji XML ponujajo možnosti izdelave meta-sheme, kamor zapišemo postopek normalizacije, t.j. določimo v katere stolpce in tabele se bodo preslikale vrednosti XML elementov. Podatkovne baze običajno že ponujajo orodja in vmesnike za pretvorbo XML podatkov v relacijski model in obratno (Vakali, 2005, str. 63).

Druga možnost, ki jo te baze podatkov ponujajo, je zapis vsebine XML dokumenta v posamezen stolpec kot niz besedila (angl. large object ali LOB). Zapis XML v takšni obliki omogoča še iskanje po besedilu (angl. full-text search).

Pričakujemo lahko, da bo razvoj relacijskih podatkovnih baz peljal v smeri popolne podpore tehnologiji XML. Uporaba podatkovnih baz z delno podporo tehnologiji XML ima določene prednosti, zaradi nepopolne podpore delu z XML podatki pa tudi določene slabosti (tabela 12).

Tabela 12: Prednosti in slabosti podatkovnih baz z delno podporo tehnologiji XML

Prednost	Opis
Zrelost	Relacijski sistemi za upravljanje baz podatkov so široko uveljavljena tehnologija in ponujajo učinkovite mehanizme za upravljanje podatkov.

Slabost	Opis
Izvirniki dokumentov	V sistemih za upravljanje baz podatkov, ki ne podpirajo zapisa izvirnega XML dokumenta je mogoča le nepopolna rekonstrukcija izvirnika. V primeru, ko potrebujemo digitalno podpisan izvirnik, to predstavlja oviro.
Poizvedbe	Učinkovito poizvedovanje po hierarhično strukturiranih dokumentih in njihovi vsebini in podpora XML poizvedovalnim jezikom.
Prenosljivost podatkov	Vsak sistem za upravljanje baz podatkov zapisuje podatke na svojstven način. Podatki zato niso lahko prenosljivi.

Vir: Vakali, 2005, str. 62; Dyck, 2001, str. 1

3.2.2.2 PODATKOVNE BAZE S POPOLNO PODPORO TEHNOLOGIJI XML

Potrebo po shranjevanju XML dokumentov so sprva zaznali manjši proizvajalci programske opreme na področju podatkovnih baz. Tako so nastale podatkovne baze, kot je Tamino, Ipedo in eXcelon, namenjene izključno zapisu XML dokumentov (Dyck, 2001, str. 1).

Podatkovne baz s popolno podporo tehnologiji XML so sicer zasedle svoj delež tržišča, vendar pa ne na račun relacijskih podatkov baz, saj lastniki velikih relacijskih sistemov niso bili pripravljeni investirati finančnih sredstev v dodaten podatkovni sistem (Trippe, 2002, str. 25).

Na tržne pritiske in poslovne potrebe so se sčasoma odzvali tudi glavni igralci na polju relacijskih podatkovnih baz. Funkcionalnosti prvinskih podatkovnih baz s popolno podporo tehnologiji XML počasi prodirajo tudi med relacijske podatkovne baze.

Lastnosti, značilne za podatkovne baze s popolno podporo tehnologiji XML (Bourret, 2004, str. 29):

- Osnovna lastnost je možnost zapisa izvornega XML dokumenta kot podatkovni tip, ki ohranja vse lastnosti XML dokumenta in pridobitev istega dokumenta iz podatkovne baze (angl. round-tripping).

- Podpora poizvedovalnim jezikom: pričakovati je, da bo na tem področju prevladal poizvedovalni jezik XQuery, razvit v W3C (The World Wide Web Consortium).

- Odlična podpora shranjevanju, popravljanju in brisanju vsebine XML dokumentov; pri tem gre lahko za spremembe samo dela dokumenta.

- Transakcijski način dela, varnostne mehanizme in podporo več uporabnikom.

- Zaklepanje podatkov; običajno se zaklepanje izvaja na nivoju celotnega dokumenta, redkeje na nivoju dela dokumenta (vozlišča). Pričakujemo lahko, da bo razvoj na tem področju prinesel učinkovitejše zaklepanje delov dokumenta in s tem možnost hkratnega dela več uporabnikov na enem dokumentu.

- Programski vmesniki; večina tovrstnih baz podatkov ponuja programske vmesnike za delo z XML dokumenti. Vmesniki omogočajo funkcionalnosti, kot je povezava na podatkovno bazo, izvajanje poizvedb in delo z rezultati poizvedbe. Rezultat poizvedbe običajno prejmemo kot niz znakov v drevesni strukturi, v katerem se premikamo s pomočjo razčlenjevalnikov, kot so DOM (angl. document object model) ali SAX (angl. Simple API for XML).

- Indeksi (angl. index): vse tovrstne podatkovne baze podpirajo uporabo indeksov z namenom učinkovitejšega izvajanja poizvedb. Običajno imamo opravka s tremi tipi indeksov. *Indeksi vrednosti* (angl. value index) vsebujejo vrednosti atributov in njihova imena. *Strukturni indeksi* (angl. structural index) določajo zaporedje elementov in atributov znotraj XML dokumenta. *Indeks za iskanje po polnem besedilu* (angl. full-text index) pa predstavlja urejen seznam znakov in nizov obravnavanega besedila.

- Od relacijskih podatkovnih baz se razlikujejo v podatkovnem modelu, ki temelji na XML tehnologiji in se razlikuje od relacijskega. Temu so prilagojeni tudi iskalni mehanizmi.

Podatkovne baze s popolno podporo tehnologiji XML so glede na svoje lastnosti specifične, saj glede na svojo specialnost pokrivajo le del potreb, ki jih delo z XML podatki zahteva. Ne nudijo namreč integracije z obstoječimi podatki, zapisanimi v relacijskih bazah podatkov. Prednosti in slabosti v primerjavi s podatkovnimi bazami z delno tehnologiji XML prikazuje tabela 13.

Tabela 13: Prednosti in slabosti podatkovnih baz s popolno podporo tehnologiji XML

Prednost	Opis
Prilagodljivost (angl. scalability)	V teh podatkovnih bazah je XML osnovni interni zapis podatkov, zato bolje in lažje odgovarjajo zahtevam odjemalca oz. strežnika. Enostavno shranjevanje dokumentov, neodvisno od njihove vsebine.
Učinkovit dostop do podatkov	Zaradi osnovnega zapisa podatkov v XML obliki dodatne pretvorbe in povezave med strukturami (tabelami) niso potrebne.
Neokrnjenost dokumentov	Pretvorbe v XML zapis ali iz XML zapisa lahko povzročijo razlike med originalom in posnetkom, ki je posledica pretvorb. Enostaven pristop do izvirnih XML dokumentov.
Poizvedbe	Učinkovito poizvedovanje po hierarhično strukturiranih dokumentih in njihovi vsebini in podpora XML poizvedovalnim jezikom.

Slabost	Opis
Nezrelost	Malo število večjih implementacij. Pomanjkanje mehanizmov, ki jih ponujajo relacijske podatkovne baze.
Integracija	Težavna integracija z obstoječimi podatki, ki niso zapisani v XML obliki.
Poizvedbe	Slaba odzivnost pri podatkovno usmerjenih dokumentih, pri združevanju (agregaciji) podatkov.

Vir: Vakali, 2005, str. 62; Dyck, 2001, str. 1

Še preden pa se odločimo, katero vrsto baze podatkov bomo uporabili, pa je potrebno odgovoriti na sledeča vprašanja (Vakali, 2005, str. 62):

- Ali so XML dokumenti podatkovno ali dokumentno usmerjeni? Podatkovno usmerjeni dokumenti se bolje razumejo z relacijskimi podatkovnimi bazami z delno podporo XML, medtem ko so dokumentno usmerjeni dokumenti bolj primerni za zapis v podatkovne baze s popolno podporo XML.

- Ali je na razpolago DTD datoteka ali XML shema? Informacije o strukturi XML dokumenta, zapisane v XML shemi ali DTD datoteki, pomagajo pri pretvorbi XML

dokumenta v relacijski podatkovni model. Dokumenti, ki nimajo predpisane strukture pa so bolj primerni za zapis v podatkovne baze s popolno podporo XML.

- Katere so performančne zahteve? Tovrstne zahteve so: množične poizvedbe, pogosto popravljanje vsebine dokumentov, masovni vnos in brisanje.

- Kakšna je narava aplikacije? Sodobne informacijske tehnologije bolj ali manj temeljijo na XML tehnologiji. Spletne aplikacije, temelječe na programskem jeziku Java in sorodnih tehnologijah (spletne storitve, ASP, SOAP), ki »govorijo in razumejo« XML jezik so bolj primerne za zapis v podatkovne baze s popolno podporo XML.

3.2.3 SHRANJEVANJE XML DOKUMENTOV V PODATKOVNIH BAZAH

V praksi se pojavlja več pristopov hranjenja XML dokumentov v podatkovnih bazah. Tabela 14 povzema osnovne lastnosti posameznih pristopov.

Tabela 14: Možni pristopi k shranjevanju XML podatkov

Pristop	Oblika zapisa podatkov	Prednosti pristopa	Slabosti pristopa
Datotečni sistem	XML datoteke se nahajajo na datotečnem sistemu, v podatkovni bazi se nahaja le povezava na datoteko.	Enostavna izvedba, primerno za manjše količine podatkov.	Otežen pristop do podatkov in zapleteno popravljanje podatkov.
Relacijske podatkovne baze	Tabele.	Enostavna izvedba, robustnost, zanesljivost delovanja.	Zahteva povezavo med mnogo tabelami za pretvorbo v XML zapis.
Baze podatkov s popolno podporo tehnologiji XML	XML kot osnovna oblika zapisa.	Prilagodljivost. Učinkovit dostop do XML podatkov.	Manj zrele kot relacijske podatkovne baze.

Vir: Vakali, 2005, str. 63

V nadaljevanju bom podrobneje predstavil drugi (relacijske podatkovne baze) in tretji (baze podatkov s popolno podporo tehnologiji XML) pristop, saj hranjenje XML datotek na datotečnem sistemu ne predstavlja ustrezne rešitve v večjih informacijskih sistemih.

Prvi pristop uporablja za zapis podatkov v relacijski podatkovni model dekompozicijo XML dokumenta. Pri tem uporabimo t.i. meta-shemo (slika 17) za uparitev dokumentne sheme v relacijsko. Meta-shema določa uparitev XML elementov in atributov s tabelami in stolpci v relacijski bazi podatkov.

V primeru, ko želimo obdržati izviren in neokrnjen XML dokument, si s prvim pristopom ne moremo kaj prida pomagati, saj je rekonstrukcija izvirnega XML dokumenta neizvedljiva, ker se izgubi neokrnjenost dokumenta, ki je temeljna zahteva pri varnem elektronskem poslovanju (Cincom, 2003, str. 3). XML dokument v tem primeru predstavlja digitalno podpisan elektronski dokument, ki je enakovreden papirnatemu in mora odražati določeno stanje v danem časovnem preseku (Champion, 2001, str. 55). Takšen pristop zahteva zapis izvirnega XML dokumenta v relacijsko podatkovno bazo.

Zapis celotnega XML dokumenta v relacijsko podatkovno bazo je mogoč na več načinov. Sprva je bilo mogoče zapisati XML dokument kot dolg besedilni niz v polje tipa CLOB (angl. character large object) ali BLOB (angl. binary large object). Razvoj relacijskih baz v novejšem času pa že prinaša podatkovne tipe, kjer lahko shranimo celoten XML dokument v polje. Razlika je le v tem, da novejši podatkovni tipi prinašajo nove funkcionalnosti, kot je iskanje po XML dokumentu in spreminjanje delov dokumenta.

Izbira pristopa je odvisna od poslovnih potreb. V nadaljevanju podrobneje opisujem oba pristopa in kriterije za izbiro ustreznega pristopa.

3.2.3.1 DEKOMPOZICIJA XML DOKUMENTOV

Pri tem pristopu se polstrukturiran podatkovni model XML dokumenta pretvori v relacijski podatkovni model. Takšnemu postopku običajno pravimo dekompozicija XML dokumenta.

Za uparitev dokumentne sheme v relacijsko uporabimo t.i. meta-shemo. Meta-shema določa uparitev XML elementov in atributov s tabelami in stolpci v relacijski bazi podatkov. Dekompozicija XML dokumenta poteka na osnovi definicije relacijskih tabel ter strukture XML sheme. Lahko vodi v podatkovne nepravilnosti, ki se večinoma odrazijo kot podvajanje podatkov (redundanca).

V postopku dekompozicije običajno zanemarimo informacije v XML dokumentu, kot so opombe, navodila za obdelavo (angl. processing instructions) in zaporedje elementov oz. atributov. Navodila za obdelavo so del XML dokumenta, ki ga XML razčlenjevalnik ne kontrolira in so namenjena posredovanju informacij aplikacijam, ki berejo XML dokument (Harold, 2004, str. 377). Slika 15 prikazuje uporabo navodil za

obdelavo znotraj XML dokumenta, kjer je podan SQL stavek, kako pridobiti podatke o osebi.

Slika 15: Navodila za obdelavo v XML dokumentu

```
<Zavarovanec>
     <Priimek>Novak </Priimek>
     <Ime>Klavdija</Ime>
     <?nameprocessor SELECT * FROM osebe?> <!-– navodila za obdelavo-->
</Zavarovanec>
```

Ker razčlenjevalnik ignorira navodila za obdelavo rekonstrukcija izvirnega XML dokumenta v popolnoma enaki obliki ni več mogoča (angl. round-tripping). Ali je to sprejemljivo, pa je odvisno od poslovnih potreb.

Pri zapisu XML podatkov v relacijsko podatkovno bazo lahko naletimo na naslednje probleme (Bourret, 2004, str. 14):

1. **Podatkovni tipi**
   XML v osnovi ne podpira podatkovnih tipov. V primeru, da v shemi ni določena struktura XML dokumenta, lahko vse podatke smatramo kot besedilo (niz). Prva možnost je, da programska oprema na osnovi XML sheme (ali DTD datoteke) pretvori XML dokument v druge tipe podatkov. Druga možnost obravnava primer, ko XML shema ne obstaja. V tem primeru si ustvarimo meta-shemo, kamor zapišemo informacije za uparitev.

   Dodatne težave lahko pričakujemo pri pretvorbi datumskih podatkov, saj obstaja množica mednarodnih možnosti zapisa. Isto velja za številčne podatke.

2. **Binarni podatki**
   Večina orodij za pretvorbo ne podpira binarnih podatkov, kar predstavlja osnovni problem.

3. **Nedoločene vrednosti (NULL values)**
   V podatkovni bazi nedoločen podatek (NULL) pomeni, da podatek ne obstaja. To ne pomeni, da ima podatek vrednost 0 ali da je niz dolžine nič. V XML terminologiji pojem nedoločenih podatkov rešujemo z neobveznimi (opcijskimi) elementi in atributi. Če vrednost za določen element ne obstaja, je ne zapišemo v dokument. Pri delu z neobveznimi elementi in atributi je potrebno paziti, da so uparjeni s polji v podatkovni bazi, ki dopuščajo nedoločene vrednosti (NULLABLE). V nasprotnem primeru zapis podatkov v podatkovno bazo ne bo uspešen.

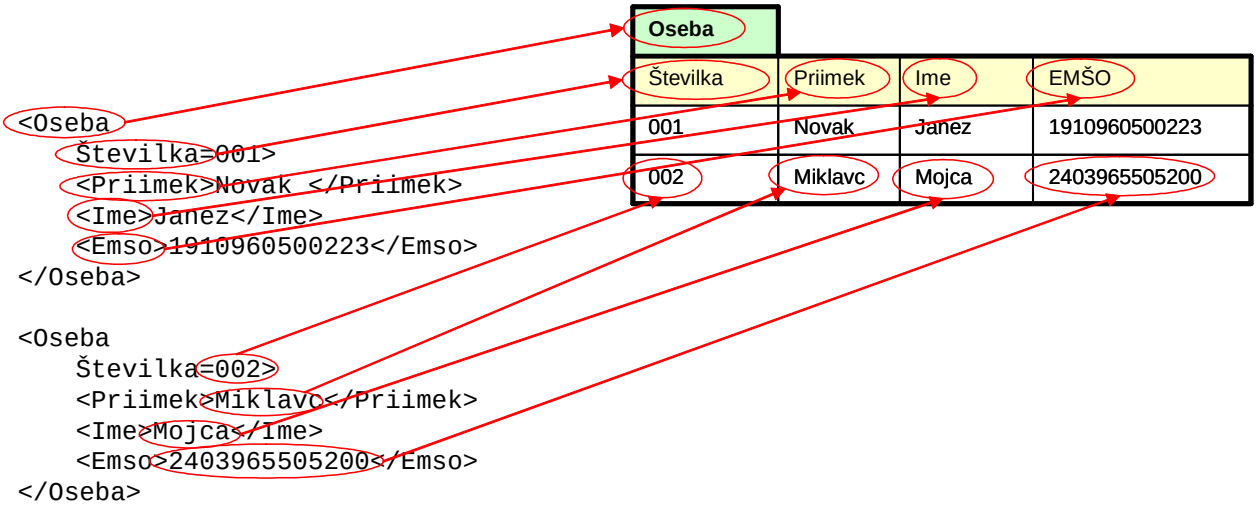
4. **Kodna stran zapisa podatkov**
   XML dokument lahko vsebuje katerikoli znak po Unicode zapisu, z izjemo določenih rezerviranih znakov. V kolikor naša podatkovna baza ne podpira Unicode kodne strani, moramo poskrbeti za ustrezno pretvorbo znakov, kar predstavlja dodaten napor.

5. **Opombe in navodila za obdelavo**
   Večina orodij za pretvorbo XML dokumentov ignorira opombe in navodila za obdelavo. V kolikor te podatke potrebujemo, jih lahko shranimo v posebne tabele, kjer jih opremimo s tujimi ključi.

Postopek izdelave relacijskega modela iz XML dokumenta, ki ga ponazarja slika 16 je sestavljen iz več korakov (Bourret, 2004, str. 16):

- Za vsak sestavljen element izdelamo tabelo in glavni ključ.
- Za vsak preprost element/atribut sestavljenega elementa in za vsak preprost element/atribut dodamo stolpec v tabelo. V kolikor poznamo tip podatka, temu ustrezno prilagodimo zapis stolpca. V nasprotnem primeru, ko imamo opravka z DTD datoteko, ali v primeru, ko XML shema ni na razpolago, stolpec zapišemo kot niz (CHAR) ali velik objekt (CLOB) v dolžini 255 znakov.
- Če je element/atribut v XML shemi neobvezen, določimo tudi stolpec kot neobvezen (NULLABLE).
- Za vsak večvrednostni (angl. multivalue) element/atribut in za vsak ponavljajoč element/atribut tvorimo ločeno tabelo, ki jo preko glavnega ključa povežemo z nadrejeno tabelo.
- Za vsak podrejen sestavljen element povežemo njegovo tabelo s tabelo nadrejenega sestavljenega elementa preko glavnega ključa.

Slika 16: Uparitev XML podatkov v relacijski model



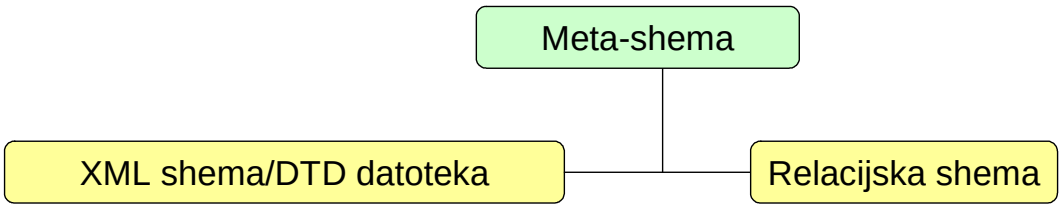
Opisani postopek služi predvsem kot ogrodje in izhodišče pri pretvorbi polstrukturiranega modela v relacijski. Težave pa lahko pričakujemo na naslednjih področjih:

- DTD datoteka ne vsebuje informacij o tipu podatka, zato je brez poznavanja logičnega modela nemogoče določiti tip podatka v podatkovni bazi.
- XML podatkovni model je običajno bolj zapleten od končne oblike relacijskega, saj relacijski model v poznejših fazah doživi še postopke

(de)normalizacije. XML dokument poleg tega lahko vsebuje še sestavljene elemente, ki pripomorejo zgolj k boljši berljivosti dokumenta, na relacijski model pa ne vplivajo.

Pri preprostih podatkovnih modelih lahko meta-shemo izdelamo ročno. Vendar je takšnih primerov v praksi zanemarljivo malo. Tovrstne potrebe so narekovale izdelavo programskih orodij, katerih poglavitna funkcionalnost je pretvorba polstrukturiranega modela v relacijski.

Slika 17: Meta-shema



Vir: Kappel, 2004, str. 366;

Funkcionalnosti, ki jih pričakujemo od takšnih orodij so (Guardalben, 2001, str. 4):

- Izdelava meta-sheme:
  Meta-shema naj temelji na W3C standardu za XML shemo in naj bo zapisana v XML jeziku. Vsebuje naj:
  - povezave element/atribut s stolpci relacijske podatkovne baze,
  - podatke referenčne integritete,
  - podatke o podatkovnih tipih in glavnih ključih tabel in
  - podporo združevanju: XML dokument lahko vsebuje dodatne sestavljene elemente, namenjene boljši preglednosti ali tipizaciji.

- Orodja za podporo uparjanju:
  Upoštevajoč reklo, da je sprememba edina stalnica v življenju, ugotovimo, da se posledično spreminja tudi podatkovni model. Naloga orodja je podpora takšnim spremembam podatkovnega modela, izdelava meta sheme, prijazen uporabniški grafični vmesnik in možnost avtomatizacije uparjanja glede na spremembe XML dokumenta.

  Po eni strani mora biti orodje sposobno ugotoviti strukturo XML dokumenta iz XML sheme ali DTD datoteke, na drugi strani pa mora poznati informacije iz kataloga podatkovne baze.

- Možnost povezave med poizvedbami (XQuery), dostopom (XPath) in pretvorbo podatkov (XSLT)
  Pretvorba podatkov iz relacijske podatkovne baze v XML obliko predstavlja težavo, saj XML dokumentov kot rezultata poizvedb običajno ne hranimo. To

pomeni, da je potrebno pretvorbo opravljati pri vsaki poizvedbi, kar je potratna operacija.

Pristop dekompozije/kompozicije XML podatkov v/iz relacijskega je primeren v naslednji primerih (Erl, 2004, str. 247):

- dinamično kreirani XML dokumenti z omejenim rokom trajanja, kjer XML služi zlasti kot transportni zapis obstoječih podatkov v relacijski podatkovni bazi in
- razni šifranti in parametrski dokumenti, shranjeni v pomnilniku aplikacije za čas trajanja aplikacije.

Prednosti pristopa:

- Z uporabo ustreznih orodij, ki predstavljajo nadgradnjo podatkovne baze (angl. database extensions), lahko hitro razvijemo informacijske rešitve.
- Ne vpliva na obstoječi podatkovni model.

Slabosti pristopa:

- Orodja, ki predstavljajo nadgradnjo podatkovne baze imajo omejene funkcionalnosti. Izhodni dokumenti, ki so rezultat teh orodij pogosto zahtevajo še dodatno obdelavo.
- Omenjena orodja so vezana na določeno podatkovno bazo, s čimer izgubimo temeljno lastnost XML dokumenta: prenosljivost.

3.2.3.2 SHRANJEVANJE IZVIRNIH XML DOKUMENTOV

V praksi obstajajo trije načini shranjevanja celotnih XML dokumentov (Bourret, 2004, str. 17):

1. kot datoteke na datotečni sistem,
2. kot velik objekt (BLOB) v relacijsko bazo podatkov ali
3. zapis XML dokumentov v podatkovno bazo s popolno podporo tehnologiji XML (angl. native XML database).

Zapis XML dokumentov na datotečni sistem

Ta postopek predstavlja najlažjo pot zapisa XML dokumentov in je primeren za manjšo količino podatkov in manjše informacijske sisteme. V podatkovni bazi se nahaja le povezava (sklic) na datoteko. Do podatkov v datotekah pridemo z ustreznimi orodji. Ker ta rešitev ni primerna za velike sisteme, je ne obravnavam podrobneje.

Zapis XML dokumentov kot velik objekt (BLOB) v relacijsko bazo podatkov

Naprednejšo rešitev predstavlja zapis dokumenta ali dela dokumenta kot dolg niz v relacijsko podatkovno bazo. S tem pridobimo vse mehanizme, ki jih relacijske baze

ponujajo, kot so: podpora transakcijam in podpora več uporabnikom, poleg tega pa še določene lastnosti XML.

Iskanje po XML dokumentu je tu izvedeno z indeksacijo podatkov v pomožnih tabelah (angl. side tables). Večina podatkovnih baz ponuja tudi mehanizme iskanja po polnem besedilu.

Obstajata dve različici tega pristopa:

1. v prvem primeru XML shema ni zapisana v relacijsko podatkovno bazo in je kontrola in obdelava dokumentov prepuščena aplikacijskemu strežniku;
2. v drugem primeru pa je XML shema zapisana v relacijsko podatkovno bazo in preko relacijske sheme povezana z XML dokumenti.

V literaturi zasledimo določena priporočila (Balmin, 2005, str. 39) glede zapisa XML kot velik objekt:

- shranjujmo zgolj dele XML dokumenta, ki so predmet poizvedb,
- tabele z velikimi objekti ločimo od tabel obstoječih podatkov, saj ti podatki običajno zavzamejo znatno količino prostora in tako vplivajo na velikost obstoječih tabel.

Pristop je uporaben v naslednjih primerih:

- uporabno za statične podatke,
- shranjevanje obstoječih podatkov v XML obliki,
- shranjevanje dokumentov, ki odražajo določeno stanje (časovni presek) in
- uporabno za pogoste poizvedbe v spletnih aplikacijah.

Prednosti pristopa:

- ni potrebe po povezovanju med tabelami in sortiranju zapisov, saj iz tabel pridobimo že shranjene dele XML dokumenta;
- pretvorbe podatkov iz relacijskega modela v XML zapis niso potrebne, kar ugodno vpliva na odzivne čase informacijskega sistema. Ta lastnost je bistvenega pomena pri spletnih aplikacijah, ki zahtevajo dobro odzivnost sistema pri velikem številu poizvedb in uporabnikov.

Slabosti pristopa:

- shranjevanje podatkov je neučinkovito, saj v vsak dokument shranimo še njegovo podatkovno shemo. Podatkovne sheme, sestavljena iz imen elementov/atributov (angl. tag names), namreč zavzema od 20 do 60% vsebine dokumenta (Ramon, 2004, str. 3).

Zapis XML dokumentov v podatkovno bazo s popolno podporo tehnologiji XML

Navkljub prodoru sodobnih tehnologij ostajajo relacijske podatkovne baze trdno v osrčju informacijskega sistema podjetij. V preteklih poglavjih sem že prikazal, na kakšne težave utegnemo naleteti pri uvajanju tehnologij, povezanih z XML. Prav tu lahko najdemo priložnost za uporabo podatkovnih baz s popolno podporo tehnologiji XML. Z razumevanjem priložnosti, ki jih te podatkovne baze ponujajo, jih lahko z obilo koristi umestimo v informacijski sistem podjetja. Res pa je, da se tudi relacijske baze podatkov razvijajo v smeri podpore funkcionalnosti, ki jih ponujajo baze s popolno podporo tehnologiji XML.

Možni razlogi in področja uporabe:

1. **Zapis dokumentno usmerjenih XML dokumentov**
   Pri uvajanju XML tehnologije se takoj srečamo z vprašanjem shranjevanja in dostopa do XML dokumentov. Prav specifičnost XML tehnologije narekuje v določenih primerih uporabo teh specializiranih podatkovnih baz, namenjenih hrambi izvirnih XML dokumentov, neodvisno od vsebine XML dokumenta. In prav na tem področju relacijske podatkovne baze največkrat odpovedo. Čeprav nam, kot sem prikazal v prejšnjih poglavjih, relacijske podatkovne baze lahko koristijo pri podatkovno usmerjenih XML dokumentih.

2. **Integrirana podpora XML shemi**
   V arhitekturah prikazanih zgoraj smo lahko XML shemo zapisali v relacijsko podatkovno bazo kot celoten dokument. V tem primeru podatkovna baza obravnava XML shemo zgolj kot niz znakov in ne kot podatkovni model medtem, ko se v podatkovnih bazah s popolno podporo tehnologiji XML, XML shema uporablja za kontrolo celovitosti in integritete XML dokumentov. Vendar gre tudi razvoj relacijskih podatkovnih baz v smeri popolne integracije XML sheme kot podatkovnega modela. Trenutno stanje bom prikazal v poglavju primerjalne analize sistemov za upravljanje s podatkovnimi bazami.

3. **Poizvedbe in dostop do podatkov**
   Podatkovne baze s popolno podporo tehnologiji XML upravljajo z vsebino XML dokumentov na drugačen način kot relacijske podatkovne baze. Osnova tega je razlika med hierarhičnim modelom, ki je drevesne strukture in relacijskim podatkovnim modelom. Posledica tega je tudi drugačen indeks, ki je prilagojen dokumentnemu modelu. Takšni indeksi omogočajo učinkovit in hiter dostop do velikih količin podatkov. Podobno je s pregledovanjem in dostopom do posameznih delov dokumenta, ki je neprimerno učinkovitejši v primerjavi z relacijskimi podatkovnimi bazami. Poizvedovalni jeziki so prilagojeni podatkovnemu modelu in se razlikujejo od jezikov, značilnih za relacijske podatkovne baze.

4. **Začasno shranjevanje XML dokumentov**
   Stalna pretvorba obstoječih podatkov, zapisanih v relacijskih podatkovnih bazah v XML zapis, je potratna in zahtevna operacija. Če upoštevamo še preverjanje strukture z XML shemo, najdemo tu priložnost za podatkovne baze s popolno podporo tehnologiji XML.

   V kolikor aplikacijski strežnik zahteva podatke v XML obliki, ki se nahajajo v relacijski podatkovni bazi, lahko tovrstna baza deluje kot posrednik, ki skrbi za pretvorbo in začasno shranjevanje XML podatkov in objektov.

3.3. POIZVEDBE

V svetu relacijskih podatkovnih baz so se skozi njihovo zgodovino razvijali tudi strukturirani poizvedovalni jeziki, kot je SQL. Le-ti so zadostili potrebam po dostopu do podatkov, tako pri ad-hoc poizvedbah, kot pri uporabi v programskih jezikih. Ti programski jeziki so pisani na kožo relacijskemu podatkovnemu modelu in izvrstno delujejo pri povezovanju tabel.

Če pa spremenimo podatkovni model, si s poizvedovalnim jezikom SQL ne moremo kaj prida pomagati. Razvoj jezikov za poizvedovanje po XML podatkih zasledimo na treh področjih (Bourret, 2004, str. 10):

1. <u>Šablonsko zasnovani poizvedovalni jeziki</u>
   V tem primeru so SQL ukazi SELECT vsebovani v šabloni, katere rezultat je XML dokument. Tovrstni poizvedovalni jeziki so izredno prilagodljivi, saj lahko SQL ukaze umestimo kamorkoli v šablono. Prav tako podpirajo uporabo spremenljivk, zank in pogojnih stavkov. Uporabljajo se predvsem pri izdelavi XML dokumentov iz podatkov, zapisanih v relacijskih podatkovnih bazah.

2. <u>Poizvedovalni jeziki, temelječi na SQL standardu</u>
   Poizvedovalni jeziki, temelječi na SQL standardu, uporabljajo ukaze SELECT in tvorijo rezultat v XML obliki. Leta 2000 je kot posledica skupnega dela več podjetij nastala ISO SQL specifikacija, bolj poznana kot SQL/XML. Ta specifikacija uvaja XML podatkovni tip in omogoča oblikovanje XML elementov iz relacijskih podatkov.

3. <u>XML poizvedovalni jeziki</u>
   XML poizvedovalni jezik lahko uporabimo pri poizvedbi nad kakršnimkoli XML dokumentom, ne glede na to, ali je zapisan v relacijski ali XML podatkovni bazi. Pomembno je le, da je zapisan v XML obliki.

Tabela 15 prikazuje osnovne razlike med jeziki, ki so uporabljeni v relacijskih podatkovnih bazah in na XML tehnologiji temelječimi poizvedovalnimi jeziki.

Tabela 15: Primerjava poizvedb in indeksiranja

	Relacijske baze podatkov	XML
Poizvedovalni jezik	Večina komercialnih sistemov za upravljanje baz podatkov podpira standard SQL. Določene od njih nudijo še nadgradnjo temu standardu.	Poizvedbe do XML dokumentov so omogočene preko programskih vmesnikov (angl. API), ki uporabljajo model DOM ali SAX, za sklicevanje na dele dokumenta pa izraze v XPath jeziku. Najbližje poizvedovalnemu jeziku SQL je XQuery, katerega skladnja (angl. syntax) omogoča poizvedbe v množici dokumentov.
Rezultat poizvedbe	SQL nudi številne ukaze za združevanje, sortiranje in nadaljnjo obdelavo podatkov.	XSLT in XQuery[5] lahko uporabimo za oblikovanje podatkov. Oba ponujata ukaze za združevanje in sortiranje podatkov.
Poizvedbe preko več virov	Z SQL ukazi lahko povežemo različne podatkovne vire, v kolikor podpirajo standard SQL.	XPath[6] ne moremo uporabiti za poizvedbe na več dokumentih. V ta namen si pomagamo z jezikom XQuery.
Indeksi	Zapleteni mehanizmi indeksiranja nudijo učinkovite poizvedbe v relacijskem podatkovnem modelu.	Indeksiranje XML podatkov se šele razvija. Obstoječe rešitve ponujajo iskanje po polnem besedilu, ali indeksi vsebine dokumentov, ki so zapisana v relacijski podatkovni zbirki.
Dostop do podatkov	Dostop do podatkov preko poizvedovalnega jezika SQL. Podatke pridobimo preko povezave več tabel.	Neposreden dostop do izvirnih XML dokumentov.

Vir: Erl, 2004, str. 240; Champion, 2001, str 53.

[5] Več o poizvedovalnem jeziku XQuery v poglavju 3.3.1.
[6] Več o jeziku za naslavljanje XML dokumentov XPath v poglavju 3.3.2.

3.3.1 POIZVEDOVALNI JEZIK XQUERY

Z naraščajočo količino XML dokumentov narašča tudi potreba po pregledovanju in iskanju vsebine teh dokumentov. Z obstoječimi poizvedovalnimi jeziki, ki jih poznajo relacijske podatkovne baze, si tu ne moremo kaj prida pomagati.

V ta namen so se razvili poizvedovalni jeziki, namenjeni iskanju po podatkih zapisanih v XML obliki, med katerimi prevladuje poizvedovalni jezik XQuery, razvit v W3C. XQuery se je razvil kot derivat več poizvedovalnih jezikov: Quilt, XQL, XML-QL, SQL, OQL, Lorel and YATL (Lehti, 2001, str. 19).

XQuery je pravzaprav poizvedovalni jezik sestavljen iz treh jezikov (Katz, 2003, str. 3). Prva plast je površinska, vidna plast, s katero se sreča uporabnik ob pisanju poizvedbe. Druga plast je v XML zapis preoblikovana prva plast. Zapisana je v jeziku XQueryX, ki je XML zapis za XQuery. Ta oblika zapisa ni najbolj berljiva, je pa primerna za obdelavo s programsko opremo, saj je zapisana v XML obliki. Tretjo, zadnjo plast, predstavlja algebraični jezik, ki opisuje notranje delovanje poizvedovalnega jezika.

XPath si kot veja, iz katere se je razvil XQuery, deli s poizvedovalnim jezikom XQuery skupen podatkovni model in se uporablja za naslavljanje delov dokumenta.

Osnovni gradniki XQuery izrazov so (Jurič, 2001, str. 33):

- izrazi za pot (angl. path expressions),
- konstruktorji elementov (angl. element constructors),
- FLWR izrazi,
- operatorji in funkcije,
- pogojni izrazi (angl. quantified constructors) in
- oblikovalci seznamov (angl. list constructors).

3.3.2 IZRAZI ZA POT

Z izrazi za pot naslavljamo določene dele drevesno organiziranega XML dokumenta. V ta namen se uporablja naslovni jezik XPath. Tako tehnologiji XQuery kot XSLT, obe razviti na W3C, temeljita na osnovah naslovnega jezika XPath. Primere naslavljanja na XML elemente/atribute z naslovnim jezikom XPath prikazuje tabela 16.

Z Xpath izrazi lahko naslavljamo vozlišča (eno ali več) v dokumentu, posamezne elemente ali atribute. Glavne Xpath ukaze povzema tabela 17.

Tabela 16: Glavni ukazi za izbiro vozlišč v Xpath

Ukaz	Opis
`vozlišče`	Izbere vsa podrejena vozlišča v vozlišču.
`/`	Izbere vsa vozlišča od korenskega elementa naprej.
`//`	Izbere vsa vozlišča dokumenta od trenutnega vozlišča navzdol.
`.`	Izbere trenutno vozlišče.
`..`	Izbere nadrejeno vozlišče trenutnega vozlišča.
`@`	Naslavlja atribut.
`*`	Izbere katerikoli element.
`[]`	Vsebuje pogoj za filtriranje.

Vir: W3Schools/XPath syntax, 2005

Tabela 17: Primeri XPath izrazov

Izraz za pot	Rezultat
`Zavarovanec`	Izbere vsa vozlišča, ki so otroci elementa »zavarovanec«.
`/posiljka`	Izbere korenski element »posiljka«. Če se izraz za pot prične z znakom »/«, to predstavlja absolutno naslavljanje elementa.
`zavarovanec/naslov`	Izbere vse elemente z imenom »naslov«, ki so otroci vozlišča »zavarovanec«.
`//naslov`	Izbere vse elemente z imenom »naslov«, ne glede na njihovo lego v dokumentu.
`zavarovanec//naslov`	Izbere vse elemente z imenom »naslov«, ki so otroci vozlišča »zavarovanec«, ne glede na to, kje znotraj elementa »zavarovanec« se nahajajo.
`//@lang`	Izbere vse atribute z imenom »lang«.
`zavarovanec/naslov[1]`	Izbere prvi element z imenom »naslov«, ki je otrok vozlišča »zavarovanec«.
`zavarovanec/naslov[last()]`	Izbere zadnji element z imenom »naslov«, ki je otrok vozlišča »zavarovanec«.

Xpath ukazi vračajo množico vozlišč, niz znakov, številko ali logično vrednost. Primer unije dveh XML vozlišč prikazuje tabela 18. XPath ukazi vsebujejo običajne aritmetične ukaze za računanje elementov/atributov, prav tako za primerjavo elementov/atributov med seboj.

Tabela 18: Primeri XPath ukazov

Ukaz	Opis	Primer	Rezultat
`|`	Unija dveh vozlišč	`//naslov | //ime`	Vrne seznam vozlišč z imenom »naslov« ali »ime«.

Vir: W3Schools/XPath operators, 2005

3.3.3 KONSTRUKTORJI ELEMENTOV

S konstruktorji izrazov lahko ustvarimo nov XML element ali atribut. Konstruktor sestavlja začetna oznaka, končna oznaka in seznam izrazov, ki sestavlja vsebino elementa.

Slika 18: XML kontruktor elementa

```
<Zavarovanec>
     <Emso>$e</Emso>
     <Priimek>$p</Priimek>
     <Ime>$i</Ime>
</Zavarovanec>
```

Slika 18 prikazuje konstruktor za sestavljen element Zavarovanec. Vrednosti posameznih elementov so določene s spremenljivkami, ki kažejo na druga mesta v poizvedbi.

Konstruktorje delimo na neposredne (angl. direct element constructors) in izračunane (angl. computed constructors). Konstruktor lahko uporabimo za izdelavo elementa, atributa ali sestavljenega elementa. Vsebino elementa ali atributa pa lahko določamo tudi dinamično, pri čemer govorimo o izračunanem konstruktorju. Le te lahko npr. uporabimo pri izdelavi programskih rešitvah, ki podporo v različnih jezikih, kar prikazuje slika 19.

Slika 19: Izračunan konstruktor

```
<entry word=“Ulica">
   <variant lang=“English">Street</variant>
   <variant lang="German">Strasse</variant>
   <variant lang="Italian">Strada</variant>
</entry>
```

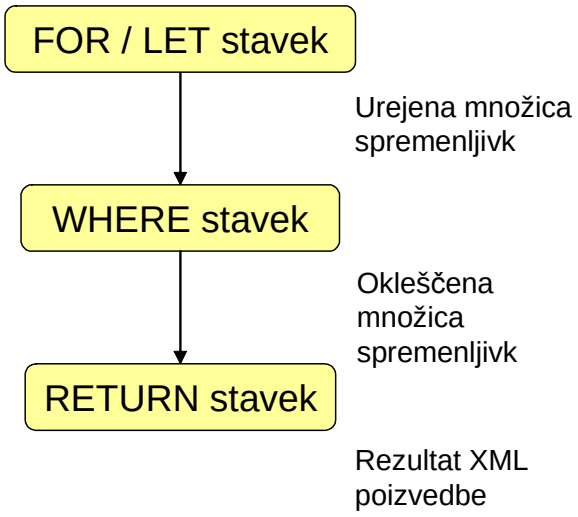
3.3.4 FLWR IZRAZI

FLWR izrazi so del poizvedovalnega jezika XQuery. Beseda FLWR predstavlja akronim besed FOR, LET, WHERE in RETURN. To so imena ukazov, ki si morajo slediti v točno določenem zaporedju (slika 20).

FLWR izraz predstavlja telo poizvedbe, v katerem povežemo več spremenljivk v končni rezultat.

Ukaz FOR uporabimo za ponavljanje ukazov v zankah. Z ukazom LET pa spremenljivkam prirejamo vrednosti, ki so lahko matematični izrazi ali vrednosti iz dokumenta. Ukazi LET se ponovijo v vsaki ponovitvi zanke.

Slika 20: Predpisano zaporedje FLWR ukazov



Vir: Jurič, 2001, str. 33

Množico spremenljivk, ki jih dobimo pri FOR in LET ukazu nato še oklestimo z ukazom WHERE. Samo tiste vrednosti, ki ustrezajo WHERE pogoju sestavljajo rezultat poizvedbe, ki ga vrne ukaz RETURN. Slika 21 prikazuje primer poizvedbe, ki v dokumentu `posiljka.xml` poišče priimke in imena zavarovancev, ki so sklenili zavarovanja v letu 2005.

Slika 21: Primer poizvedbe v XQuery

```
FOR $p IN document(“posiljka.xml")//zavarovanje
WHERE year-from-dateTime($p/DatumZacetka) = 2005
RETURN $p/Priimek, $p/Ime
```

3.3.5 OPERATORJI IN FUNKCIJE

XQuery ponuja množico ukazov in funkcij za delo z nizi, števili, datumski podatki in logičnimi izrazi.

Slika 22: Primer uporabe XQuery operatorjev

```
FOR $p IN document(“posiljka.xml")//zavarovanje
WHERE year-from-dateTime($p/DatumZacetka) = 2005
RETURN concat($p/Priimek, “ “, $p/Ime)
```

Slika 22 prikazuje primer poizvedbe, ki v dokumentu `posiljka.xml` poišče priimke in imena zavarovancev, ki so sklenili zavarovanja v letu 2005 in vrne sešteta niza priimka in imena, ločena s presledkom.

Celoten nabor ukazov najdemo na spletnem naslovu:
http://www.w3.org/TR/xquery-operators/

3.3.6 POGOJNI IZRAZI

Pogojne izraze uporabljamo tam, kjer je vsebina rezultata odvisna od določenega pogoja. Pogojne izraze lahko poljubno gnezdimo in uporabimo povsod, kjer potrebujemo logičen izračun vrednosti nekega izraza. Slika 23 predstavlja primer, ko lahko na osnovi spola osebe določimo njeno predpono.

Slika 23: Primer uporabe XQuery logičnih izrazov

```
FOR $p IN document(“posiljka.xml")//zavarovanje
RETURN
concat(IF $p/Spol = “M”
             THEN “g. “
             ELSE “ga. “,
     $p/Priimek, “ “, $p/Ime)
```

Celoten nabor ukazov najdemo na spletnem naslovu:
http://www.w3.org/TR/xquery-semantics/#id-logical-expressions

4. PRIMERJALNA ANALIZA SUBP

V poglavju 3.2.2 sem podatkovne baze glede na podporo XML tehnologiji razdelil v dve skupini: podatkovne baze z delno podporo tehnologiji XML in podatkovne baze s popolno podporo tehnologiji XML. V prvem primeru gre za pretežno relacijske podatkovne baze, nadgrajene z določenimi XML funkcionalnostmi, v drugem pa za specializirane podatkovne baze, ki so namenjene shranjevanju XML dokumentov.

Od sistema za upravljanje baz podatkov pričakujem, da se čimbolj približa sodobnim poslovnim potrebam, ki jih narekuje elektronsko poslovanje in sodobne tehnologije.

Predmet primerjalne analize sistemov za upravljanje podatkovnih baz bodo predvsem relacijske podatkovne baze in sicer zaradi naslednjih razlogov:

1. Obstoječi podatki se večinoma nahajajo v relacijskih podatkovnih bazah. Delež podatkov, ki jih najdemo v specializiranih podatkovnih bazah s popolno podporo XML tehnologiji je zanemarljiv.

2. Iz podatkov je razvidno, da podjetja še vedno skoraj polovica sredstev, namenjenih informacijski tehnologiji, namenjajo relacijskim podatkovnim bazam (tabela 19).

3. Njihov delež v primerjavi z XML podatkovnimi bazami v zadnjih nekaj letih zanemarljivo pada. To lahko pripišemo zakasnelim odzivom proizvajalcev

programske opreme na prihajajoče tehnologije elektronskega poslovanja (tabela 20). Podatki v tabeli za leto 2003, 2004, 2005 predstavljajo napoved vrednosti na osnovi predvidevanj avtorja članka.

Tabela 19: Deleži sredstev, potrošenih za shranjevanje podatkov glede na arhitekturo

Način shranjevanja podatkov	Delež [%]
Relacijske podatkovne baze	49
Datotečni sistem	41
Spletna stran	12
Dokumentni sistem	10
Imeniški sistemi	5
Polstrukturirane shrambe	3
Ostalo	1

Vir: Trippe, 2002, str. 25

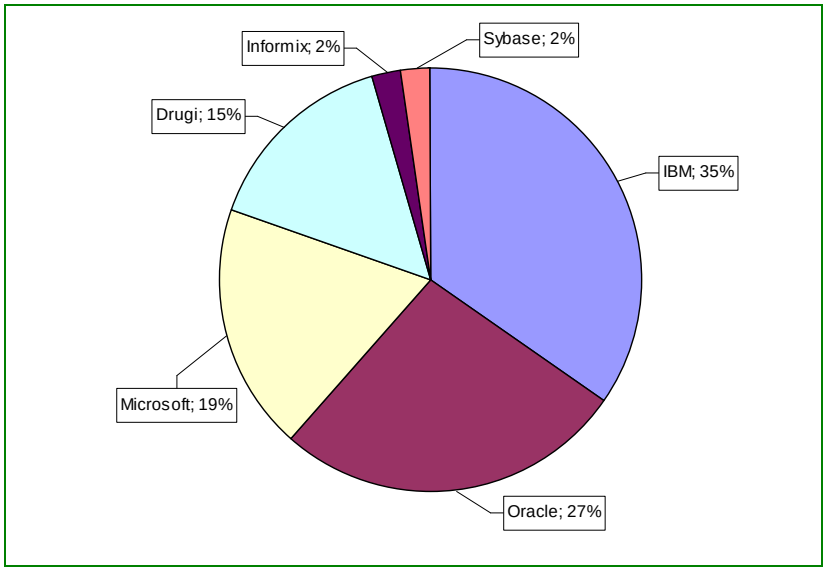
Tabela 20: Tržni deleži podatkovnih baz v odstotkih

	2000	2001	2002	2003 (napoved)	2004 (napoved)	2005 (napoved)
Relacijske	85	85	84	77	74	70
Objektne	15	13	11	8	6	5
XML	0	2	5	15	20	25

Vir: Trippe, 2002, str. 25

Na tržišču se pojavljajo trije glavni proizvajalci sistemov za upravljanje podatkovnih baz, ki skupaj predstavljajo delež preko 80% prodanih licenc, kar prikazuje slika 24.

Slika 24: Delež prodanih licenc po proizvajalcih



Vir: Gartner media relations, 2003

Zaradi deleža, ki ga omenjeni trije ponudniki zasedajo na tržišču, bodo njihovi sistemi za upravljanje podatkovnih baz predstavljali variante primerjalne analize.

V literaturi zasledimo določene raziskave na tem področju (Seng, 2003; Cincom, 2003; Kappel, 2004). Vsaka od raziskav se problema loteva na svoj način glede na cilje raziskave.

Ugotovitve teh raziskav lahko strnemo v sledeče:

- XML tehnologija se šele razvija in uveljavlja;
- pri razvoju XML ima ključno vlogo XML shema, ki je pod okriljem W3C;
- standardi, kot je XML shema, so pomembni pri premostitvi razlik med relacijskim in XML podatkovnim modelom;
- relacijske podatkovne baze so primerne za shranjevanje dobro strukturiranih in podatkovno usmerjenih dokumentov in
- relacijske podatkovne baze ponujajo zelo omejene funkcionalnosti pri delu z XML podatki.

Rezultati teh raziskav niso več nujno aktualni, saj vsaka nova verzija programske opreme prinese nove funkcionalnosti, kar vpliva na oceno določenega orodja. Namen tega poglavja je osvežiti navedene ugotovitve ter prispevati nova spoznanja na tem področju.

V poglavju bom primerjal tri sisteme za upravljanje podatkovnih baz:

- **DB2 V8:**
  DB2 sistem za upravljanje baz podatkov je izdelek podjetja IBM. Predstavlja robustno platformo za shranjevanje podatkov s podporo odprtim standardom. Podpira razvojna okolja J2EE in Microsoft .NET. Največ primerov uporabe srečamo na večjih centralnih računalnikih tipa IBM (IBM: DB2 Product Family, 2005).

- **Oracle 10g:**
  Oracle Database 10g je sistem za upravljanje baz podatkov istoimenskega proizvajalca. Sistem za upravljanje baz podatkov vključuje razvojno okolje za razvoj aplikacij ter podpira mrežno računalništvo (angl. grid computing). Verzija 10g prinaša podpora poizvedovalnemu jeziku XQuery, podporo izvirnim XML dokumentom ter izboljšave pri XSL pretvorbah
  (Drake, 2005, str. 4).

- **SQL Server 2005:**
  SQL Server 2005 sistem za upravljanje baz podatkov je izdelek podjetja Microsoft. V verziji 2005 so dodane nove funkcionalnosti za delo z XML podatki (Microsoft Corporation: XML Options in Microsoft SQL Server 2005,

2005). Novosti se nanašajo predvsem na uporabo jezika XPath pri poizvedbah, predstavitev obstoječih podatkov v XML obliki, zapis XML podatkov v relacijsko podatkovno bazo.

Celotno poglavje 3 je namenjeno analizi razlik med relacijskimi podatkovnimi bazami in tehnologijo XML. Namen tega poglavja pa je analizirati pripravljenost izdelkov, ki jih najdemo na tržišču na tehnologije elektronskega poslovanja.

4.1. ODLOČITVENI MODEL

Hierarhični večparametrski odločitveni modeli temeljijo na razgradnji odločitvenega problema na manjše podprobleme. Pri tem določimo kriterije, ki vplivajo na oceno posamezne variante. Rezultat tega je drevo kriterijev. Pri tem skušamo upoštevati priporočilo, da vsakemu kriteriju sledijo trije podrejeni kriteriji.

Sledi izdelava funkcije koristnosti, s katero določimo vplivnost posameznih parametrov na končno odločitev.

4.1.1 DOLOČITEV KRITERIJEV

Določitev kriterijev za primerjalno analizo je prvi korak na poti izdelave odločitvenega modela.

Pri tem je potrebno upoštevati naslednje zahteve:

- kriteriji naj bodo merljivi,
- upoštevati je treba načelo polnosti: zajeti moramo vse kriterije, ki pomembno vplivajo na odločitev in
- zaželena doseči ortogonalnost (kriteriji naj se ne prekrivajo) kriterijev. V kolikor pri definiciji kriterijev naletimo na kriterije, ki med seboj niso popolnoma ortogonalni, je potrebno združevanje takšnih kriterijev.

Kot sem že omenil, od relacijske podatkovne baze pričakujem, da ponuja funkcionalnosti, ki jih podatkovne baze s popolno podporo XML tehnologiji ponujajo že od svojih zametkov. Pričakujem torej konvergenco tehnologije relacijskih podatkovnih baz in XML tehnologije v največji možni meri.

Osnova za izbiro kriterijev bodo torej lastnosti, ki jih že imajo baze s popolno podporo XML tehnologiji. Večja, kot bo konvergenca relacijske podatkovne baze, boljša bo funkcija koristnosti.

V mojem primeru sem kriterije razdelil na naslednje sklope:

1. **Upravljanje podatkov**
   1.1. Zapis podatkov iz relacijske podatkovne baze v XML obliko
   1.2. Zapis XML dokumentov v relacijsko podatkovno bazo
   1.3. Shranjevanje izvirnih XML dokumentov
   1.4. Mehanizmi za upravljanje vsebine XML dokumentov

2. **Uparitev podatkov**[7]
   2.1. Uparitev XML podatkov v relacijsko shemo
   2.2. Izdelava meta-sheme
   2.3. Zapis meta-sheme

3. **Kontrola podatkov**
   3.1. Način kontrole
   3.2. Razčlenjevalnik

4. **Poizvedbe**
   4.1. Podpora poizvedovalnim jezikom
   4.2. Optimiranje poizvedb s pomočjo indeksov

4.1.1.1 UPRAVLJANJE PODATKOV

V tem poglavju obravnavam kriterije, ki zadevajo upravljanje XML podatkov v relacijskih podatkovnih bazah. Ti kriteriji zajemajo možnosti zapisa XML dokumentov v relacijsko podatkovno bazo bodisi kot izviren XML dokument bodisi kot množico zapisov v relacijski shemi. Prav tako me zanima možnost oblikovanja XML dokumentov iz obstoječih podatkov v relacijski podatkovni bazi.

Zapis podatkov iz relacijske podatkovne baze v XML obliko

Zapis podatkov iz relacijske podatkovne baze v XML obliko uporabljamo običajno pri posredovanju in pretvorbi obstoječih podatkov v sodobnejše oblike elektronskega poslovanja, kjer tehnologija XML služi kot transportni mehanizem. Pri tem gre lahko za računalniško izmenjavo podatkov med fizičnimi ali pravnimi osebami, dinamično kreiranje vsebine spletnih strani, spletne storitve, ipd… Zapisu podatkov iz relacijske podatkovne baze v XML obliko pravimo tudi kompozicija XML dokumenta.

Namen tega kriterija je določiti, v kolikšni meri relacijska podatkovna baza podpira kompozicijo XML dokumentov.

[7] Kriterija 1.1 ter 1.2 nista popolnoma ortogonalna s kriteriji 2.1, 2.2 in 2.3. V primeru združitve teh kriterijev bi model preveč izgubil na preglednosti, zato obdržim navedeno strukturo kriterijev.

Zaloga vrednosti kriterija:

- **slabo**: iz relacijske podatkovne baze ni mogoče pridobiti podatkov v XML obliki;
- **sprejemljivo**: iz relacijske podatkovne baze je mogoče pridobiti podatke v XML obliki z zelo omejeno funkcionalnostjo in v določenih primerih;
- **dobro**: iz relacijske podatkovne baze lahko v večini primerov pridobimo podatke v XML obliki;
- **odlično**: iz relacijske podatkovne baze je mogoče pridobiti podatke v XML obliki;

Zapis XML dokumentov v relacijsko podatkovno bazo

Pri tem postopku polstrukturiran podatkovni model XML dokumenta pretvorimo v relacijski podatkovni model. Pri tem iz celotnega (ali dela) XML dokumenta nastane množica zapisov, ki so razporejeni v tabelah relacijske podatkovne baze. Takšnemu postopku običajno pravimo dekompozicija XML dokumenta.

Namen tega kriterija je določiti, v kolikšni meri relacijska podatkovna baza podpira dekompozicijo XML dokumentov.

Zaloga vrednosti kriterija:

- **slabo**: dekompozicije XML dokumenta ni mogoče opraviti;
- **sprejemljivo**: dekompozicijo XML dokumenta lahko opravimo v omejenem obsegu;
- **dobro**: XML dokumente je mogoče zapisati v podatkovno bazo v omejeni obliki;
- **odlično**: dekompozicijo XML dokumenta lahko v vsakem primeru opravimo.

Shranjevanje izvirnih XML dokumentov

Shranjevanje izvirnih XML dokumentov nam pride prav v naslednjih primerih:

1. zapis dokumentno usmerjenih XML dokumentov,
2. zapis XML sheme v relacijsko podatkovno bazo,
3. arhiviranje izvornih XML dokumentov,
4. poizvedbe po XML dokumentih in
5. začasno shranjevanje XML dokumentov.

S tem kriterijem ugotavljam, ali podatkovna baza nudi podporo zapisu izvornih XML dokumentov. Na oceno dodatno vpliva dejavnik, v kolikšni meri zapisani XML dokument ohrani svoje lastnosti (sklicevanje na podatke v dokumentu z uporabo XPath, popravljanje dokumenta, ohranitev izvirnika v smislu ohranitve praznega prostora, opomb in digitalnih podpisov).

Zaloga vrednosti kriterija:

- **slabo**: XML dokumenta ni mogoče shraniti v podatkovno bazo;

- **sprejemljivo**: XML dokument lahko shranimo kot velik objekt (LOB ali CLOB). XML podatkovni tip ni podprt. Določene lastnosti XML, kot so prazen prostor ter opombe in navodila za obdelavo se ne ohranijo;
- **dobro**: XML dokument lahko shranimo kot velik objekt (LOB ali CLOB). XML podatkovni tip ni podprt. Ohrani se prazen prostor ter opombe in navodila za obdelavo;
- **odlično**: XML dokument lahko shranimo v podatkovno bazo v izvirni obliki. Podatkovni tip omogoča vse postopke upravljanja z XML podatki.

Mehanizmi za upravljanje vsebine XML dokumentov (dodajanje, popravljanje in brisanje vsebine)

Uporabo XML dokumentov lahko razširimo na področje, kjer nam le-ti, poleg prenosnega medija in hranjenja izvirnikov, predstavljajo podatkovno zbirko. V takšnem primeru prihaja do popravljanja vsebine XML dokumentov. S tem kriterijem opišem možnost popravljanja vsebine XML dokumentov zapisanih v podatkov zbirki.

Zaloga vrednosti kriterija:

- **slabo**: upravljanje vsebine XML dokumentov ni mogoče;
- **sprejemljivo**: upravljanje vsebine XML dokumentov je delno podprto;
- **dobro**: omogočena je večina postopkov dela z XML podatki;
- **odlično**: vsi postopki dela z XML podatki so omogočeni.

4.1.1.2 UPARITEV PODATKOV

Pri pretvorbi polstrukturiranega podatkovnega modela v relacijski je potrebno izvesti uparitev med modeloma. Uparitev obeh modelov zahteva izdelavo t.i. meta-sheme, ki povezuje elemente/atribute XML dokumenta s polji tabel v relacijski podatkovni bazi.

Uparitev XML podatkov v relacijsko shemo

Kriterij opredeljuje možnost preslikave XML podatkov s podatki iz relacijske podatkovne baze s pomočjo meta-sheme. Meta shema naj vsebuje naslednje informacije:

- povezave element/atribut s stolpci relacijske podatkovne baze,
- podatke referenčne integritete,
- podatke o podatkovnih tipih in glavnih ključih tabel in
- podporo združevanju: XML dokument lahko vsebuje dodatne sestavljene elemente, namenjene boljši preglednosti ali tipizaciji.

Brez uporabe meta-sheme so podatki o povezavah med podatki XML dokumenta in podatkovno bazo običajno zapisani kar v programski kodi. Tak pristop ni priporočljiv z vidika prilagodljivosti in vzdrževanja programske opreme.

Zaloga vrednosti kriterija:

- **ni podprto**: izdelava meta-sheme ni mogoča;
- **podprto**: izdelava meta-sheme je mogoča;

Izdelava meta-sheme

Kriterij opredeljuje način izdelave meta-sheme. Pri tem lahko meta-shemo izdelamo ročno, vendar je tak pristop priporočljiv le pri zelo enostavnih podatkovnih modelih, ki se ne spreminjajo pogosto.

V praksi se poslovne zahteve stalno spreminjajo, z njimi pa običajno tudi podatkovni model. In najmanj česar si želimo, je pogosto ročno spreminjanje povezav med podatki XML dokumenta in podatkovno bazo, posledica česar so napake na povezavah ter manjkajoči in nepopolni podatki.

Avtomatizacija takšnih postopkov s pomočjo namenskih orodij je zelo zaželena predvsem pri kompleksnejših podatkovnih modelih. Pri tem nam kot dobro izhodišče za nabor podatkov služi kar XML shema.

Zaloga vrednosti kriterija:

- **slabo**: meta-shemo je potrebno izdelati ročno.
- **sprejemljivo**: meta-shema je zapisana v XML jeziku, kar omogoča urejanje z ostalimi orodji za oblikovanje XML dokumentov.
- **dobro**: meta-shemo je mogoče izdelati z grafičnimi orodji, ki so sestavni del sistema za upravljanje baz podatkov. Ob vsaki spremembi relacijske sheme ali XML dokumenta so potrebne ročne dopolnitve meta-sheme.
- **odlično**: meta-shema je zapisana v obliki dopolnjene XML sheme (angl. annotated schema). Izdelava meta-sheme je enostavna, podprta z grafičnimi orodji. Mogoča je avtomatska prilagoditev meta-sheme ob spremembi relacijske sheme ali XML dokumenta.

Zapis meta-sheme

Kriterij opredeljuje način zapisa meta-sheme. Pri tem je na razpolago več alternativ:

1. Trdo kodirano ob izvajanju pretvorbe
   povezave med podatki XML dokumenta in podatkovno bazo so kodirane v programski kodi. Takšen način predstavlja najslabšo možnost izdelave meta-sheme, saj je zelo občutljiv na spremembe. Vsaka sprememba relacijske sheme

ali XML dokumenta zahteva spremembe v programski kodi.

2. V datoteki (XML dokument)
   V tem primeru so povezave med podatki XML dokumenta in podatkovno bazo zapisane v XML obliki. Meta-shema v XML zapisu se lahko nahaja na datotečnem sistemu, ali pa je zapisana v relacijski podatkovni bazi kot XML objekt.

3. V relacijski podatkovni bazi
   V tem primeru so povezave med podatki XML dokumenta in podatkovno bazo zapisane v relacijski podatkovni bazi. Vsaka vrstica v tabeli predstavlja povezavo med enim XML elementom/atributom in stolpcem v relacijski tabeli.

Zaloga vrednosti kriterija:

- **slabo**: meta-shema je zapisana v programski kodi;
- **sprejemljivo**: meta-shema je zapisana v dokumentu, ki ni del sistema za upravljanje baz podatkov;
- **dobro**: meta-shema je kot XML dokument zapisana v bazi kot del sistema za upravljanje baz podatkov;
- **odlično:** meta-shema je kot XML dokument zapisana v bazi kot del sistema za upravljanje baz podatkov. Sistem za upravljanje baz podatkov podpira upravljanje z verzijami meta-shem.

4.1.1.3 KONTROLA PODATKOV

Pred zapisom XML podatkov v relacijsko podatkovno bazo je smiselno preveriti njihovo ustreznost. S tem mislim na skladnost podatkov glede na predpisano strukturo, zapisano v DTD datoteki ali XML shemi.

Način kontrole

Ta kriterij opredeljuje, na kakšen način sistem za upravljanje baz podatkov podpira kontrolo XML dokumentov.

Podpora XML shemi kot naprednejši in zmogljivejši obliki definiranja XML modela je vsekakor zaželena, saj ponuja veliko več možnosti kot pa DTD datoteka. Naprednejše možnosti se kažejo predvsem v novih tipih podatkov, definiranju zaporedja elementov, možnosti definiranja oblike zapisa, ipd…

Določeni razčlenjevalniki XML dokumentov še ne podpirajo uporabe XML sheme.

Zaloga vrednosti kriterija:

- **slabo**: ni podpore za kontrolo XML dokumentov;
- **sprejemljivo**: kontrola samo s pomočjo DTD datoteke;
- **dobro**: omejena kontrola s pomočjo DTD datoteke ali XML sheme;
- **odlično**: popolna kontrola s pomočjo DTD datoteke ali XML sheme.

Razčlenjevalnik

Naloga razčlenjevalnika je kontrola XML dokumenta glede na XML shemo oz. DTD datoteko in pretvorba XML dokumenta v objekt drevesno strukturo (DOM), ki jo lahko obdelujemo s programsko opremo.

S tem kriterijem opredelim, ali sistem za upravljanje baz podatkov nudi podporo razčlenjevalnikom XML dokumentov in v kakšni meri so razčlenjevalniki integrirani v podatkovno bazo.

V praksi se pojavljata dva načina pregledovanja XML dokumentov: SAX in DOM. Značilnosti in razlike med njima podrobneje opredeljuje poglavje 2.3.3.

Zaloga vrednosti kriterija:

- **slabo**: ni podpore za XML razčlenjevalnik;
- **sprejemljivo**: podpora samo enemu XML razčlenjevalniku;
- **dobro**: omejena podpora obema tipoma XML razčlenjevalnika;
- **odlično**: tako SAX kot DOM razčlenjevalnik sta v celoti podprta.

4.1.1.4 POIZVEDBE

Podpora poizvedovalnim jezikom

S tem kriterijem ugotavljam, v kolikšni meri sistem za upravljanje baz podatkov nudi podporo poizvedovalnim jezikom, temelječih na tehnologiji XML.

Zaloga vrednosti kriterija:

- **slabo**: ni podpore za XPath in XQuery;
- **sprejemljivo**: delna podpora za XPath ali XQuery;
- **dobro**: popolna podpora XPath ali XQuery in delna podpora XPath ali XQuery;
- **odlično**: popolna podpora XPath in XQuery.

Optimiranje poizvedb s pomočjo indeksov

Ta kriterij določa, ali sistem za upravljanje baz podatkov ponuja možnost uporabe indeksov pri iskanju XML dokumentov oz. njihove vsebine, z namenom izboljšanja učinkovitosti dostopa do podatkov.

Tipi indeksov, ki se pojavljajo za optimiranje poizvedb:

- **drevesni indeksi:** ti indeksi so v relacijskih podatkovnih bazah poznani že od začetka njihove uporabe. Zgradimo jih na osnovi vsebine zapisov, zapisanih v relacijskih tabelah;

- **XML indeksi**: so drevesne strukture, sestavljene iz poti in vrednosti, zapisanih v XML dokumentu. Uporabljamo jih za popravljanje vsebine XML dokumentov in za izvajanje XPath in SQL/XML ukazov;

- **indeksi za iskanje po polnem besedilu**: te indekse lahko zgradimo na stolpcih, ki so tipa XML. Vsebino teh indeksov predstavlja vsebina XML elementov, pri čemer se XML oznake zanemarijo. Prav tako se zanemarijo vrednosti XML atributov.

Zaloga vrednosti kriterija:

- **slabo**: podpora drevesnim indeksom. Ni podpore XML indeksom ter indeksom za iskanje po polnem besedilu;
- **sprejemljivo**: podpora drevesnim indeksom, delna podpora XML indeksom. Ni podpore indeksom za iskanje po polnem besedilu;
- **dobro**: podpora drevesnim indeksom, popolna podpora XML indeksom. Ni podpore indeksom za iskanje po polnem besedilu;
- **odlično**: podpora drevesnim indeksom, popolna XML indeksom in indeksom za iskanje po polnem besedilu.

4.1.2 FUNKCIJA KORISTNOSTI

Pri izgradnji večparametrskega odločitvenega modela sem si pomagal z lupino ekspertnega sistema DEX-i. Zgoraj opisanim kriterijem sem določil uteži in na ta način določil funkcijo koristnosti odločitvenega modela. Uteži posameznih kriterijev prikazuje slika 25.

Kriterijem, ki opisujejo upravljanje podatkov, sem naklonil večjo težo (50%), saj predstavljajo jedro funkcionalnosti podatkovne baze glede na XML tehnologijo. Kriterijem, ki opisujejo možnosti povezane z meta-shemo ter poizvedbami sem dodal petinsko (20%) utež. Najmanjši vpliv imajo kriteriji, povezani s kontrolo podatkov (10%), saj lahko to opravimo z orodji tudi izven podatkovne baze.

Izbira uteži je v tem primeru subjektivna in odvisna od ocenjevalca, kar smatram za slabost. S povečanjem števila ocenjevalcev bi lahko pripomogli k boljšemu odločitvenemu modelu. Menim, da so izbrani kriteriji dovolj merodajni in v največji meri ortogonalni. V primeru kriterijev, ki se tičejo izdelave in upravljanja z meta-

shemo, ortogonalnosti nisem uspel doseči, saj se kriteriji ne izključujejo med seboj v popolnosti. Zato sem kljub pomembnosti, ki jo predstavlja celoten sklop kriterijev, namenil 20% vpliv na končno odločitev.

Slika 25: Uteži posameznih kriterijev v odločitvenem modelu

Kriterij	Lokalne	Globalne	Lok.norm.	Glob.norm.
XML podpora v podatkovnih bazah				
Upravljanje podatkov	50,0	50,0	50,0	50,0
Zapis podatkov v XML obliki	25,0	12,5	25,0	12,5
Zapis XML v podatkovno bazo	25,0	12,5	25,0	12,5
Shranjevanje izvirnih XML dokumentov	25,0	12,5	25,0	12,5
Upravljanje z vsebino XML dokumentov	25,0	12,5	25,0	12,5
Uparitev podatkov	20,3	20,3	20,3	20,3
Uparitev z relacijsko shemo	51,2	10,4	34,4	7,0
Izdelava meta-sheme	27,6	5,6	37,0	7,5
Zapis meta sheme	21,3	4,3	28,6	5,8
Kontrola podatkov	9,5	9,5	9,5	9,5
Način kontrole	50,0	4,7	50,0	4,7
Pregledovalnik	50,0	4,7	50,0	4,7
Poizvedbe	20,3	20,3	20,3	20,3
Podpora poizvedovalnim jezikom	65,0	13,2	65,0	13,2
Optimiranje poizvedb	35,0	7,1	35,0	7,1

Opisani model pokriva le kriterije, ki se tičejo XML tehnologije. To pomeni, da na izbiro sistema za upravljanje baz podatkov v praksi vpliva še množica dejavnikov, ki v modelu niso zajeti. Primeri takšnih dejavnikov so: obstoječe programsko okolje v podjetju, število uporabnikov, stroški investicije, ipd… V takšnih primerih je potrebno dopolniti tako drevo kriterijev, kot odločitvena pravila (funkcijo koristnosti). Model, ki bi bil v celoti prenosljiv v drugo okolje, v tem primeru težko najdemo.

4.2. OPIS VARIANT

Vse, v nadaljevanju napisane ugotovitve, so pridobljene iz literature, ki se nahaja na spletnih mestih posameznih podjetij in zato niso empiričnega značaja.

4.2.1 DB2 V8

DB2 uporablja orodje, imenovano XML Extender, za upravljanje podatkov v XML obliki. XML Extender v verziji 8 še ni povsem integriran v podatkovni bazi in ga lahko namestimo kot dodatek k podatkovni bazi.

4.2.1.1 UPRAVLJANJE PODATKOV

Zapis podatkov iz relacijske podatkovne baze v XML obliko

Pri kompoziciji XML dokumentov XML Extender uporablja shranjene procedure. Predpogoj za uporabo shranjenih procedur pa je izdelava DAD datoteke, ki opredeljuje preslikavo med XML dokumentom in relacijskimi tabelami.

DAD datoteka je podrobneje opisana v poglavju 5.3.3.

DB2 ponuja dva načina kompozicije XML dokumentov:

1. Dokumenti, ki jih občasno popravljamo
   V ta namen obstaja shranjena procedura »dxxGenXML« za kreiranje XML dokumentov. Procedura v ta namen uporablja DAD datoteko. XML datoteka se kreira iz podatkov, shranjenih v zbirki tabel. Zbirka tabel je določena v DAD datoteki z elementom »Xcollection«.

   Shranjena procedura za vsak izdelan XML dokument tvori vrstico v relacijski tabeli. Ta tabela običajno vsebuje samo en stolpec, ki je lahko tipa VARCHAR, CLOB, XMLVARCHAR ali XMLCLOB.

   V kolikor želimo preveriti veljavnost XML dokumenta, XML Extender v tabelo doda še en stolpec, kamor zapiše veljavnost XML dokumenta (1 za veljaven in 0 za neveljaven XML dokument).

2. Dokumenti, ki jih pogosto popravljamo
   Pri dokumentih, do katerih često dostopamo in katerih vsebina se pogosto spreminja, uporabimo shranjeno proceduro »dxxRetrieveXML«. V tem primeru gre za ponavljajoč postopek in je odzivnost sistema odločilna. V tem primeru XML Extender namesto DAD datoteke uporabi ime XML zbirke kot vhod in preko nje dostopa do DAD datoteke.

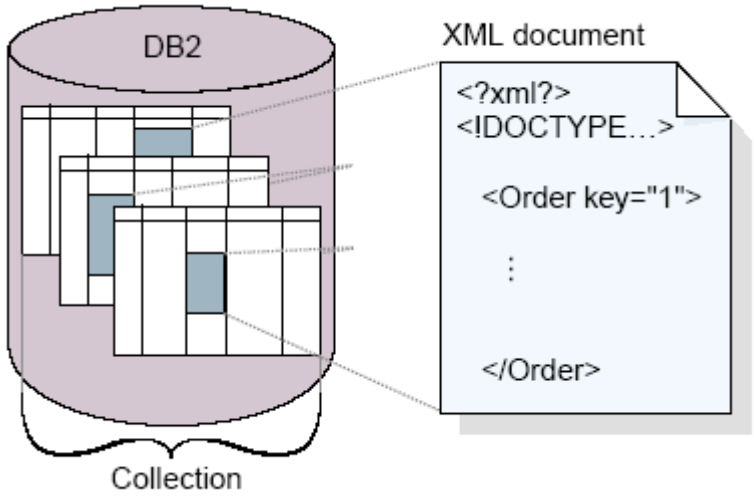
DB2 ne podpora uporabe SQL/XML funkcij. Podatke v XML obliki lahko pridobimo le preko uporabe shranjenih procedur.

Zapis XML dokumentov v relacijsko podatkovno bazo

DB2 uporablja za zapis XML podatkov v relacijske tabele t.i. metodo XML zbirke (angl. XML collection). XML zbirka je skupina tabel, ki vsebujejo podatke iz XML dokumentov, kar prikazuje slika 26. DAD dokument je uporabljen za preslikavo DTD dokumenta v relacijsko podatkovno shemo.

Metode za dostop in shranjevanje nam omogočajo pripravo XML dokumentov iz obstoječih podatkov, zapisanih v relacijski podatkovni bazi in zapis XML podatkov v relacijske tabele.

Slika 26: XML dokument, shranjen kot XML zbirka



Vir: IBM Redbook: XML Extender Administration and Programming Version 8, str. 119

Shranjevanje izvirnih XML dokumentov

DB2 Extender uporablja metodo XML stolpec (angl. XML column) za shranjevanje celotnih XML dokumentov. V ta namen DB2 ponuja tri nove uporabniško določene tipe (angl. user defined type - UDT):

1. XMLCLOB,
   v katerem shranimo vsebino XML dokumenta kot velik objekt (CLOB).

2. XMLVARCHAR,
   v katerem shranimo vsebino XML dokumenta kot besedilo spremenljive dolžine (VARCHAR).

3. XMLFile,
   v katerem shranimo povezavo na XML dokument, ki se nahaja na datotečnem sistemu.

Na ta način lahko shranimo v stolpec XML dokument (slika 27). Lahko pa ta pristop uporabimo tudi za shranjevanje DTD datotek ali XML sheme.

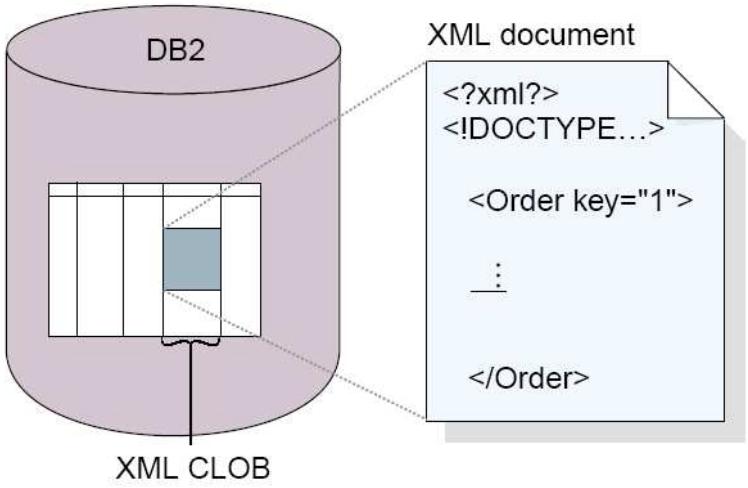
Pri tem se ohrani tudi prazen prostor (angl. white spaces), ki ga poznamo v XML tehnologiji. Prav tako se ohranijo meta podatki, kot so opombe in navodila za obdelavo (angl. processing instructions).

Mehanizmi za upravljanje vsebine XML dokumentov

Z uporabo omenjenih treh tipov podatkov lahko izvajamo nad podatki naslednje operacije:

- hranimo XML dokumente v podatkovni bazi,
- pridobimo XML podatke iz relacijske baze,
- popravljamo vsebino XML dokumentov in
- brišemo XML podatke ali dokumente.

Slika 27: XML dokument, shranjen kot DB2 stolpec



Vir: IBM Redbook: XML Extender Administration and Programming Version 8, str. 98

XML Extender v ta namen ponuja uporabniško določene funkcije (angl. user-defined functions - UDF). Vsebino XML dokumentov lahko popravljamo na več načinov:

Popravljanje vsebine celotnega XML dokumenta

Za vsak uporabniško določen tip obstaja uporabniško določena funkcija za uvoz podatkov iz zunanjih virov. Slika 28 prikazuje zamenjavo vsebine XML stolpca z vsebino XML dokumenta, ki se nahaja na datotečnem sistemu, v primerih, kjer se pojavlja določen referent.

Slika 28: Ažuriranje vsebine XML stolpca

```
UPDATE narocila
set narocilo = XMLVarcharFromFile('c:\narocilo.xml')
WHERE referent = ’Janez Novak’
```

Popravljanje vsebine posameznih elementov/atributov

Pri popravljanju posameznih elementov/atributov se z XPath sklicujemo na element/atribut, čigar vsebino želimo popravljati. Slika 29 prikazuje primer, kjer v XML stolpcu z imenom »narocilo« vrednost elementov »/Narocilo/Stranka/Naziv« zamenjamo z »Lek«.

Pri tem je potrebno omeniti, da se v takšnem primeru zamenja vrednost vseh elementov/atributov, ki ustrezajo rezultatu XPath poizvedbe. Vrednosti v pomožnih tabelah se osvežijo samodejno.

Slika 29: Ažuriranje vsebine elementa v XML stolpca

```
UPDATE narocila
set narocilo = Update(narocilo, '/Narocilo/Stranka/Naziv', 'Lek')
WHERE referent = 'Janez Novak'
```

Ob uporabi funkcije »UPDATE« se zgodijo določene neželene spremembe:

- opombe izven korenskega elementu se izgubijo,
- prazen prostor izven oznak se ne ohrani,
- navodila za obdelavo se ohranijo in
- spremenljivke in sklici se nadomestijo z vrednostmi.

Te spremembe so posledica izhodne strukture, ki jo vrača razčlenjevalnik.

Brisanje XML dokumentov

Brisanje XML dokumentov izvedemo z brisanjem zapisov, ki vsebujejo XML dokumente v XML stolpcih. Pri tem se sklicujemo na vsebino pomožnih tabel, kamor se pišejo vrednosti elementov/atributov. Slika 30 prikazuje primer brisanja vseh XML dokumentov, ki vsebujejo izdelek s ceno višjo od 2.500 SIT.

Slika 30: Brisanje XML dokumentov iz podatkovne baze

```
DELETE from narocila
WHERE stevilka_narocila in
     (SELECT stevilka_narocila from narocila_side_tab
      WHERE  cena > 2500)
```

4.2.1.2 UPARITEV PODATKOV

Uparitev XML podatkov v relacijsko shemo

XML Extender uporablja dokument imenovan DAD (angl. document access definition) za preslikavo XML elementov in atributov v relacijske tabele (XML Extender Administration and Programming, str. 203). DAD datoteka je zapisana v XML obliki. Isti dokument se uporablja tudi za indeksiranje elementov in atributov, z namenom pohitritve iskanja, ali za zapis podatkov iz relacijske podatkovne baze v XML obliko. DAD datoteke so shranjene kot nizi (CLOB). Njihova velikost ne sme preseči 100KB.

DAD datoteka opredeljuje tudi DTD datoteko, ki se uporablja za preverjanju pravilnosti XML dokumenta.

Funkcija DAD datoteke je odvisna od načina dela z XML dokumenti.

DAD datoteka za XML stolpec

Za XML stolpce se DAD datoteka uporablja predvsem za pohitritev iskanja t.j. za izdelavo indeksa.

V DAD datoteko v tem primeru zapišemo le povezave, za katere želimo ustvariti indeks. V DAD datoteko za vsako povezavo XML elementa/atributa s podatkovno bazo zapišemo informacije:

- **name**: ime stolpca v pomožni tabeli,
- **type**: SQL podatkovni tip v pomožni tabeli,
- **path**: pot v XML dokumentu ter
- **multi_occurance**: ali se XML element/atribut, opredeljen pod »path«, lahko pojavi večkrat v XML dokumentu.

Slika 31 prikazuje vsebino DAD datoteke, za primer, ko v stolpec shranimo celotno naročilo. V DAD datoteko v tem primeru zapišemo podatke, po katerih pogosto povprašujemo, npr. številko naročila ali količino.

Slika 31: Primer DAD datoteke za XML stolpec

```
<table name="narocila">
<column name ="stevilka_narocila"
     type="integer"
     path="/narocilo/stevilka"
     multi_occurrence="NO"/>

<column name ="kolicina"
     type="integer"
     path="/narocilo/kolicina"
     multi_occurrence="NO"/>
</table>
```

DAD datoteka za XML zbirko

V primeru XML zbirke se DAD datoteka uporabi za preslikavo XML dokumenta v relacijske podatkovne tabele. Na osnovi te preslikave lahko nato iz vsebine tabel sestavimo XML dokument. Na enak način lahko tudi opravimo dekompozijo XML dokumenta.

DAD datoteka v tem primeru vsebuje naslednje informacije:

- oznako <Xcollection>:
  ta opredeljuje, da so XML podatki zapisani v XML zbirki. XML zbirka je skupina običajnih relacijskih tabel. Razlika je le v tem, da vsebujejo podatke, preslikane iz XML dokumentov.

- strukturo XML dokumenta, sestavljeno iz :
  - root_node: opredeljuje korenski element dokumenta,
  - element_node: element, ki je lahko bodisi korenski ali otrok,

- text_node: predstavlja vsebino elementa in
- attribute_node: predstavlja atribut elementa.

Za razliko od DAD datoteke za XML stolpec, ki služi predvsem za izdelavo indeksov na podatkih je DAD datoteka za XML zbirko namenjena preslikavi XML podatkov v relacijske tabele. Primer preslikave XML podatkov o naročilu v relacijske tabele prikazuje slika 32.

Slika 32: Primer DAD datoteke za XML zbirko

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM ""c:\dtd\dad.dtd">
<DAD>
...
<Xcollection>
<SQL_stmt>
...
</SQL_stmt>

<prolog>?xml version="1.0"?</prolog>
<doctype>!DOCTYPE Order SYSTEM
""c:\dtd\narocilo.dtd""</doctype>
<root_node>
<element_node name = "Narocilo">   --> opredeli element »Narocilo«
     <attribute_node name = "stevilka"> --> opredeli attribut »stevilka«
     <column name = "stevilka"/>
     </attribute_node>

     <element_node name="Stranka">
          <text_node>
          <column name="stranka">
          </text_node>
     </element_node>
...
</element_node>
...
</root_node>
</Xcollection>
</DAD>
```

Izdelava meta-sheme

Meta-shemo v obliki DAD datoteke lahko napišemo ročno. Primeri vsebine so navedeni v prejšnjem poglavju. To opravilo je naporno, dolgotrajno in dopušča možnost napak.

IBM ponuja za izdelavo in urejanje meta sheme, zapisane v DAD datoteki, urejevalnik imenovan »RDB to XML Mapping Editor«. Le-to orodje deluje v navezi z XML Extender-jem (Russell, str. 2).

Urejevalnik omogoča naslednje postopke:

- grafični vmesnik za preslikavo med XML dokumentom in relacijskimi podatki;
- avtomatsko izdelavo DAD datoteke, s čimer se izognemo napakam, ki bi nastale ob ročnem urejanju datoteke in
- pretvorbo relacijskih podatkov v XML dokumente.

Zapis meta-sheme

V primeru DB2 je meta-shema zapisana v DAD datoteki, ki je zapisana v XML obliki. DAD datoteka se nahaja na datotečnem sistemu, lahko pa jo zapišemo v podatkovno bazo.

4.2.1.3 KONTROLA PODATKOV

Način kontrole

XML Extender ponuja možnost preverjanja ustreznosti XML dokumenta (angl. validation) v obeh primerih: XML zbirki ali XML stolpcu.

V kolikor želimo izvajati preverjanje ustreznosti XML dokumenta, to označimo v DAD datoteki. Pri tem je potrebno upoštevati sledeče:

- preverjanje ustreznosti XML dokumenta se izvaja pred zapisom v relacijsko bazo;
- preverjanje ustreznosti XML podatkov vpliva na odzivnost sistema.

DB2 XML Extender v namen preverjanja ustreznosti ponuja dve uporabniško določeni funkciji (angl. user defined function - UDF), ki preverita skladnost XML podatkov z DTD datoteko oz XML shemo:

- **db2xml.svalidate**: preveri skladnost XML podatkov z XML shemo in
- **db2xml.dvalidate**: preveri skladnost XML podatkov s DTD datoteko.

Funkciji vračata vrednost 1, če je dokument veljaven in vrednost 0, če element ni veljaven. Sporočilo o morebitni napaki se izpiše v datoteko dnevnika.

V DAD deklaraciji se določi le DTD datoteka. Možnost uporabe XML sheme ni omenjena.

Razčlenjevalnik

XML Extender uporablja SAX razčlenjevalnik. Ker DOM razčlenjevalnik ni podprt lahko naletimo na težave pri XSL pretvorbah, saj le-ta v tem primeru deluje bolje kot SAX razčlenjevalnik. Prav tako v programski kodi ni mogoče uporabljati XPath izrazov za naslavljanje XML elementov/atributov, saj tovrstno naslavljanje podpira le DOM razčlenjevalnik.

4.2.1.4 POIZVEDBE

Podpora poizvedovalnim jezikom

XML Extender uporablja samo del XPath skladnje za sklicevanje na XML elemente/atribute in XSLT za pretvorbo XML podatkov (Nicola, 2004, str. 81).

XQuery in FLRW izrazi niso podprti.

Optimiranje poizvedb s pomočjo indeksov

V primeru XML zbirke uporabimo indekse, kot pri običajnih relacijskih tabelah.

V primeru XML stolpca pa lahko za iskanje po vsebini XML dokumenta pohitrimo z uporabo pomožnih tabel (angl. side tables). Z izdelavo indeksov na teh tabelah nato izboljšamo učinkovitost iskanja po podatkih.

Pred uporabo pomožnih tabel je potrebno poznati poslovne potrebe. Kar pomeni, da morami poznati podatke in pogostost izvajanja poizvedb, po katerih povprašujemo. Na teh podatkih potem izdelamo indekse.

Vsak stolpec v pomožnih tabelah identificiramo z ustreznim elementom/atributom v XML stolpcu. XML Extender samodejno vzdržuje vsebino teh tabel (indekse), ko shranimo XML dokument v XML stolpec.

4.2.2 ORACLE 10G

Oracle uporablja orodje XML DB za delo z XML podatki. XML DB predstavlja v tem primeru funkcionalnosti, povezane z XML tehnologijo, ki jih ponuja podatkovna baza.

Oracle ponuja dodatno podporo v obliki knjižnice z imenom XSU (XML SQL Utility). Ta ponuja naslednje funkcionalnosti na področju razvoja v programskem jeziku Java in SQL (Lee, 2005, str. 2):

- pretvorba podatkov iz objektno relacijskih podatkovnih baz v XML;
- zapis podatkov iz XML dokumenta v relacijsko podatkovno bazo;
- popravljanje ali brisanje vsebine podatkov v relacijski podatkovni bazi na osnovi XML dokumenta.

Ta pripomoček načeloma deluje s katerimkoli JDBC gonilnikom, čeprav najbolje deluje s sistemom za upravljanje baz podatkov Oracle.

4.2.2.1 UPRAVLJANJE PODATKOV

Oracle pozna dva načina dela shranjevanja XML dokumentov: strukturirani in nestrukturirani. Prvi se uporablja za zapis podatkov v relacijske tabele, drugi za zapis izvirnih XML dokumentov.

Zapis podatkov iz relacijske podatkovne baze v XML obliko

Oracle XML DB omogoča pretvorbo podatkov, zapisanih v relacijskih tabelah v XML obliko. Pri tem uporablja SQL/XML funkcije (slika 33), iz katerih sestavimo SQL stavek, ki opravi pretvorbo. Rezultat pretvorbe je eden ali več XML dokumentov, ki se nahajajo v t.i. XML View-u.

Slika 33: Primer uporabe in rezultat SQL/XML funkcij

```
SELECT XMLElement("Znamenitost", XMLAttributes(javno AS javno),
          XMLElement("Naziv",Znamenitost_naziv),
          XMLElement("Kraj",Kraj),
          XMLElement("URL",Znamenitost_url))
FROM Znamenitost;

<Znamenitost javno="DA">
  <Naziv>Postojnska jama</Naziv>
  <Kraj>Postojna</Kraj>
  <URL>http://www.postojnska-jama.si/</URL>
</Znamenitost>

<Znamenitost javno="DA">
  <Naziv>Triglavski narodni park</Naziv>
  <URL>http://www.sigov.si/tnp/</URL>
</Znamenitost>
```

Zapis XML dokumentov v relacijsko podatkovno bazo

Oracle XML DB na osnovi registracije XML sheme[8] samodejno izdela relacijske pomožne tabele, kamor se ob obdelavi XML dokumenta zapišejo podatki. Pomembno pri tem je, da zaporedje podatkov ostane enako kot v XML dokumentu.

Te tabele so uporabne v več namenov:

- uporaba običajnih SQL poizvedb nad relacijskimi tabelami,
- izdelava indeksov za pohitritev poizvedb in
- optimalna hramba podatkov brez označb (angl. tags) in meta podatkov.

Shranjevanje izvirnih XML dokumentov

Oracle v verziji 10 prinaša nov podatkovni tip, imenovan XMLType, namenjen shranjevanju izvirnih XML dokumentov v podatkovno bazo. Pri tem se ohrani tudi

[8] Več o tem v poglavju 4.2.2.2

prazen prostor (angl. white spaces), ki ga poznamo v XML tehnologiji. Prav tako se ohranijo meta podatki, kot so opombe in navodila za obdelavo (angl. processing instructions).

Novost v verziji 10 je še XML DB Repository. XML DB Repository je shramba, ki omogoča shranjevanje XML dokumentov ali map. Organizacija te shrambe je hierarhična, podobna datotečnemu sistemu.

Mehanizmi za upravljanje vsebine XML dokumentov

Oracle XML DB uporablja standardne SQL/XML funkcije za upravljanje vsebine XML dokumentov:

- **`insertChildXML`**
  doda podrejen XML element/atribut na podano mesto.

- **`insertXMLbefore`**
  doda XML podatke (lahko tudi sestavljene) pred podano mesto.

- **`appendChildXML`**
  doda XML podatke (lahko tudi sestavljene) na zadnje mesto pod nadrejeni element.

- **`deleteXML`**
  briše XML podatke na podanem mestu XML dokumenta.

- **`updateML`**
  omogoča popravljanje vsebine XML dokumenta na osnovi XPath izrazov.

4.2.2.2 UPARITEV PODATKOV

Uparitev XML podatkov v relacijsko shemo

Uparitev podatkov v relacijsko shemo poteka na osnovi obstoječe XML sheme, ki jo dopolnimo z navodili (angl. annotations), kam naj se preslikajo XML podatki. Ta navodila so v obliki XML atributov, ki se imenujejo »xdb:SQLName«, vrednosti teh atributov pa odražajo stolpec v relacijski tabeli, kamor se zapiše XML podatek.

Postopek prilagoditve XML sheme je enak bodisi v primeru lastne XML sheme bodisi v primeru, ko nam XML shemo posreduje poslovni partner. Primer dopolnjene XML sheme za naročilo s podatki o preslikavi prikazuje slika 34.

Slika 34: Meta-shema preslikave XML podatkov v relacijske v Oracle XML DB

```
<xs:complexType name="Narocilo" xdb:SQLType="XDBPO_TYPE">
 <xs:sequence>
     <xs:element ref="Sklic"/>
     <xs:element name="Partner" type="Partner" minOccurs="0"
     xdb:SQLName="PARTNERID"/>

     <xs:element ref="StroskovnoMesto"/>
     <xs:element name="Navodila" type="Navodila" xdb:SQLName="NAVODILA"/>

     <xs:element name="Postavke" type="Postavke"
     xdb:SQLName="POSTAVKE"/>
 </xs:sequence>
</xs:complexType>
```

Izdelava meta-sheme

Oracle XML DB na osnovi registracije XML sheme samodejno izdela relacijske pomožne tabele, kamor se ob obdelavi XML dokumenta razporedijo podatki. Slika 35 prikazuje primer registracije XML sheme »narocilo.xsd«.

Slika 35: Registracija XML sheme

```
dbms_xmlschema.registerSchema
('http://127.0.0.1:8080/mojiPodatki /xsd/narocilo.xsd', 'narocilo.xsd')
```

Oracle XML DB ponuja omejen grafični vmesnik, s katerim določimo, kateri XML elementi/atributi, se bodo prepisali v relacijske tabele.

Zapis meta-sheme

Ob registraciji XML sheme se v podatkovni besednjak (angl. data dictionary) zapišejo vsi SQL objekti in XMLType tabele, potrebne za zapis podatkov iz XML dokumenta.

4.2.2.3 KONTROLA PODATKOV

Način kontrole

ORACLE XML DB podpira vse funkcionalnosti, ki jih prinaša XML shema. Podatka o podpori DTD datotekam nisem zasledil. Ker gre za starejšo tehnologijo, se mi podpora temu tipu kontrole ne zdi odločujoč dejavnik.

Razčlenjevalnik

ORACLE XML DB podpira obe vrsti razčlenjevalnikov: DOM in SAX. To je dobrodošlo, saj se oba tipa razčlenjevalnikov med seboj razlikujeta, tako po namenu uporabe kot po svojih tehničnih značilnostih. Medtem, ko je DOM razčlenjevalnik namenjen izdelavi ali popravljanju XML dokumentov, je SAX razčlenjevalnik namenjen predvsem učinkovitemu branju XML le-teh.

4.2.2.4 POIZVEDBE

Podpora poizvedovalnim jezikom

Oracle XML DB nudi podporo poizvedovalnemu jeziku XQuery za poizvedovanje po XML dokumentih. V verziji Oracle Database 10g Release 2 najdemo podporo za FLWR (FLWOR) izraze (Banerjee, 2005, str. 22).

Optimiranje poizvedb s pomočjo indeksov

Oracle XML DB podpira štiri vrste indeksov:

- **drevesni indeksi (B-Tree index)**. Ti indeksi so uporabni pri zapisu XML podatkov v relacijske tabele (strukturiran pristop). Indeksi temeljijo na SQL tipih, ki sestavljajo relacijske tabele in se uporabljajo pri SQL poizvedbah.

- **XML indeksi**. Uporabni so tako v strukturiranem kot nestrukturiranem načinu dela z XML podatki. Uporabljamo jih za popravljanje vsebine XML dokumentov in za izvajanje XPath in SQL/XML ukazov.

- **Namenski indeksi**. Te indekse lahko zgradimo na stolpcih ali tabelah, ki so tipa XMLType. Njihova uporaba je primerna za poizvedbe v XPath poizvedovalnem jeziku.

- **Indeksi za iskanje po polnem besedilu**. Te indekse lahko zgradimo na stolpcih ali tabelah, ki so tipa XMLType.

4.2.3 SQL SERVER 2005

SQL Server 2005 v primerjavi s prejšnjimi verzijami SQL Server prinaša številne nove možnosti upravljanja z XML podatki. Te novosti se nanašajo predvsem na podporo shranjevanju izvirnih XML dokumentov in izboljšavam pri izdelavi indeksov za učinkovitejše iskanje po XML podatkih.

4.2.3.1 UPRAVLJANJE PODATKOV

Zapis podatkov iz relacijske podatkovne baze v XML obliko

SQL Server 2005 uporablja t.i. tehnologijo XML pogledov (angl. XML view) za zapis obstoječih podatkov v XML obliko. Nad XML pogledi lahko izvajamo poizvedbe v jeziku XPath. XPath poizvedba se nato pretvori v SQL poizvedbo, ki se izvede nad relacijskimi tabelami.

SQL Server 2005 ponuja tudi hibridni model. Takšen model je uporaben v primeru, ko se del podatkov nahaja v relacijskih tabelah, preostali del podatkov pa v XML obliki. Vzroki za takšen model so lahko izboljšanje odzivnosti in učinkovitosti sistema, težave z zaklepanjem podatkov ali optimiranje prostora, ki ga podatki zasedajo. Na drugi strani pa takšen model prinaša večjo odgovornost za vzdrževanje podatkovnega modela in povezav med podatki.

Zapis XML dokumentov v relacijsko podatkovno bazo

XML pogled lahko uporabimo za zapis XML dokumentov v relacijske tabele oz. popravljanje vsebine relacijskih na osnovi podatkov, zapisanih v XML dokumentih.

Shranjevanje izvirnih XML dokumentov

SQL Server 2005 ponuja dve možnosti shranjevanja izvirnih XML dokumentov:

- **Kot XML podatkovni tip (XML data type):**
  Podatki se na ta način shranijo v XML obliki. Pri tem se ohrani vsebina XML elementov/atributov, hierarhija podatkov in zaporedje zapisov. Pomembno pri tem je, da vsebina XML dokumenta ni verna kopija izvirnika, saj se izgubijo naslednje značilnosti: prazen prostor (angl. white spaces), imenska področja elementov, XML deklaracija in zaporedje atributov.

  Ta možnost ponuja izdelavo indeksov za učinkovito iskanje.

- **Kot velik objekt (angl. LOB [n]varchar(max) in varbinary(max)):**
  V tem primeru se ohrani točna kopija izvirnika. V tem primeru izdelava indeksov ni mogoča, kar znatno vpliva na učinkovitost iskanja podatkov. Če so poizvedbe po izvirnikih pogoste, izdelamo dvojnik podatkov kot XML podatkovni tip, in izvedemo XQuery poizvedbo.

Mehanizmi za upravljanje vsebine XML dokumentov

Vsebino XML dokumentov lahko popravljamo z uporabo funkcije »modify«. XQuery ukaz opremimo z ukazi »insert«, »delete« in »update«, s katerimi dodajamo, popravljamo ali brišemo XML elemente. Slika 36 prikazuje dodajanje XML elementa s pomočjo uporabe funkcije »modify«. V tem primeru za prvim poglavjem dodamo podatke o drugem poglavju.

Slika 36: Primer dodajanja XML elementa s pomočjo funkcije »modify«

```
UPDATE docs SET xCol.modify('
     insert
     <poglavje števila = "2">
          <naslov>Uvod</naslov>
     </poglavje>
     after (/doc/poglavje[@številka=1])[1]')
```

Slika 37 prikazuje popravljanje vsebine XML elementa s pomočjo funkcije »modify«, kjer zapisu z določeno ISBN številko popravimo ceno na 1.500 SIT.

Slika 37: Primer popravljanja vsebine XML elementa s pomočjo funkcije »modify«

```
UPDATE Knjige
SET    Document.modify ('
       declare namespace kn = "http://mojeKnjige";
       replace value of (/kn:knjigarna/kn:knjiga
       [@ISBN="1-861003-11-0"]/kn:cena)[1] with 1500.00')
```

4.2.3.2 UPARITEV PODATKOV

Uparitev XML podatkov v relacijsko shemo

SQL Server 2005 uporablja dopolnjeno XML shemo (angl. annotated schema - AXSD) za uparitev XML podatkov v relacijsko shemo. Z uporabo dopolnjene XML sheme se XML dokument razgradi v stolpce relacijskih tabel, pri čemer se ohrani njegova hierarhična struktura (Microsoft Corporation, 2005, str. 1).

Obstoječo XML shemo dopolnimo z naslednjimi navodili - atributi:

- **sql:relation** - priredi XML element relacijski tabeli in
- **sql:field** - priredi element ali atribut stolpcu relacijske tabele.

Če XML shema ne vsebuje navodil za uparitev se upoštevajo privzete nastavitve. To pomeni, da se sestavljeni elementi preslikajo v tabelo, enostavni Elementi/atributi pa v stolpce z enakim imenom.

Izdelava meta-sheme

SQL Server 2005 ne ponuja ustreznega orodja za izdelavo meta sheme. Preslikavo XML modela v relacijskega je potrebno opraviti ročno. Pri tem si lahko pomagamo z drugimi urejevalniki XML shem.

Zapis meta-sheme

Meta-shema je zapisana kot dopolnjena XML shema. Meta-shema se lahko nahaja bodisi na datotečnem sistemu ali v podatkovni bazi. Meta-sheme lahko naložimo v

predpomnilnik (angl. caching). S tem ugodno vplivamo na odzivnost sistema, saj se obdelava XML podatkov izvaja v pomnilniku.

4.2.3.3 KONTROLA PODATKOV

Način kontrole

SQL Server 2005 ponuja zbirke XML shem (angl. XML schema collection) kot možnost upravljanja s standardnimi W3C shemami. XML dokument, zapisan kot XML podatkovni tip v relacijski bazi, nato povežemo z XML zbirko in na ta način zagotovimo upoštevanje pravil, zapisanih v XML shemi.

Z XML zbirko je omogočen razvoj in prilagodljivost XML sheme, saj lahko na ta način združimo različne verzije XML sheme, ki se je spreminjala skozi čas.

SQL Server 2005 pozna dva tipa XML dokumentov:

1. **tipiziran XML dokument** je XML dokument povezan z XML shemo in
2. **netipiziran XML dokument**, ki ne pripada XML shemi.

Zbirka XML shem lahko vsebuje eno ali več XML shem, ki med seboj niso nujno povezane. XML sheme v XML zbirki se med seboj ločijo po svojih imenskih področjih.

Razčlenjevalnik

SQL Server 2005 podpira obe vrsti razčlenjevalnikov: DOM in SAX. To je dobrodošlo, saj se oba tipa razčlenjevalnikov med seboj razlikujeta, tako po namenu uporabe kot po svojih tehničnih značilnostih. Medtem, ko je DOM razčlenjevalnik namenjen izdelavi ali popravljanju XML dokumentov, je SAX razčlenjevalnik namenjen predvsem učinkovitemu branju XML le-teh.

4.2.3.4 POIZVEDBE

Podpora poizvedovalnim jezikom

SQL Server 2005 nudi podporo poizvedovalnemu jeziku XQuery za poizvedovanje po XML dokumentih ter podporo za FLWOR izraze (Vithanala, 2005, str. 3). Poizvedbe, napisane v XQuery poizvedovalnem jeziku, lahko izvajamo nad XML pogledi. Pri tem se XML pogled obnaša kot navidezen XML dokument.

Optimiranje poizvedb s pomočjo indeksov

SQL Server 2005 podpira dve vrsti indeksov:

- **Glavni XML indeks** (angl. primary XML index):
  Pri tem se vrednosti XML elementov/atributov in njihovih poti shranijo v notranji

tabeli v obliki B+ drevesa.

- **Pomožni XML indeksi** (angl. secondary XML indexes):
  Pomožni indeksi dodatno povečajo učinkovitost izvajanja poizvedb in popravljanja vrednosti podatkov. Obstajajo tri vrste pomožnih indeksov:
  - PATH: tovrstne indekse uporabimo pri sklicevanju na dele XML dokumenta npr.: `/knjiga[@ISBN = "0-2016-3361-2"`. Še posebej se obnesejo pri dolgih izrazih za pot.

  - PROPERTY: ta indeks uporabimo, kadar v SELECT stavku navajamo več vrednosti iz XML dokumenta.

  - VALUE: ta indeks je uporaben pri sklicevanju na dele besedila, ki vsebujejo potomce (Xpath ukaz //) in nadomestne znake (angl. wildcard), npr.: `/knjiga[@* = "roman"]`.

- **Indeksi za iskanje po polnem besedilu**
  Neodvisno od prej omenjenih indeksov lahko izdelamo indekse za iskanje po polnem besedilu. Le-ti vsebujejo zgolj vrednosti XML elementov, odstranijo pa označbe in XML atribute. Indeksov za iskanje po polnem besedilu torej ne moremo uporabiti za iskanje po XML atributih.

  Indeksi za iskanje po polnem besedilu so uporabni pri slabo strukturiranih dokumentih z bogato vsebino.

4.3. REZULTATI VREDNOTENJA VARIANT

V tem poglavju združujem ugotovitve, pridobljene pri analizi posameznih variant. Pri vrednotenju variant sem si pomagal z lupino ekspertnega sistema DEX-i. Izbranim kriterijem sem dodal uteži in na ta način dobil funkcijo koristnosti.

Vsako varianto sem ovrednotil po že opisanih kriterijih. Na ta način je nastala slika 38, ki prikazuje vrednotenje variant.

V nadaljevanju komentiram še ocene posameznih variant po sklopih kriterijev.

Slika 38: Rezultati vrednotenja variant

Kriterij	DB2 V8	Oracle 10g	SQL Server 2005
XML podpora v podatkovnih bazah	dobro	odlično	dobro
Upravljanje podatkov	dobro	odlično	dobro
Zapis podatkov v XML obliki	dobro	odlično	dobro
Zapis XML v podatkovno bazo	dobro	odlično	dobro
Shranjevanje izvirnih XML dokumentov	dobro	odlično	odlično
Upravljanje z vsebino XML dokumentov	dobro	odlično	odlično
Uparitev podatkov	dobro	dobro	dobro
Uparitev z relacijsko shemo	podprto	podprto	podprto
Izdelava meta-sheme	sprejemljivo	dobro	sprejemljivo
Zapis meta sheme	dobro	dobro	dobro
Kontrola podatkov	dobro	odlično	odlično
Način kontrole	sprejemljivo	odlično	odlično
Pregledovalnik	dobro	odlično	odlično
Poizvedbe	sprejemljivo	odlično	odlično
Podpora poizvedovalnim jezikom	sprejemljivo	odlično	odlično
Optimiranje poizvedb	dobro	odlično	odlično

4.3.1 UPRAVLJANJE PODATKOV

Vse tri podatkovne baze v osnovi ponujajo zahtevane funkcionalnosti.

Oracle XML DB ponuja bogate možnosti za delo z XML dokumenti in se v funkcionalnostih najbolj približa bazam s popolno podporo XML tehnologiji. Tudi v SQL Server so možnosti za delo z XML že znotraj podatkovne baze.

DB2 Extender v tej fazi predstavlja zgolj nadgradnjo podatkovne baze. Prav tako ni podpore SQL/XML funkcijam. Podatke v XML obliki lahko pridobimo le preko uporabe shranjenih procedur.

Skupne ocene za posamezne sisteme za upravljanje baz podatkov na področju upravljanja podatkov prikazuje slika 39.

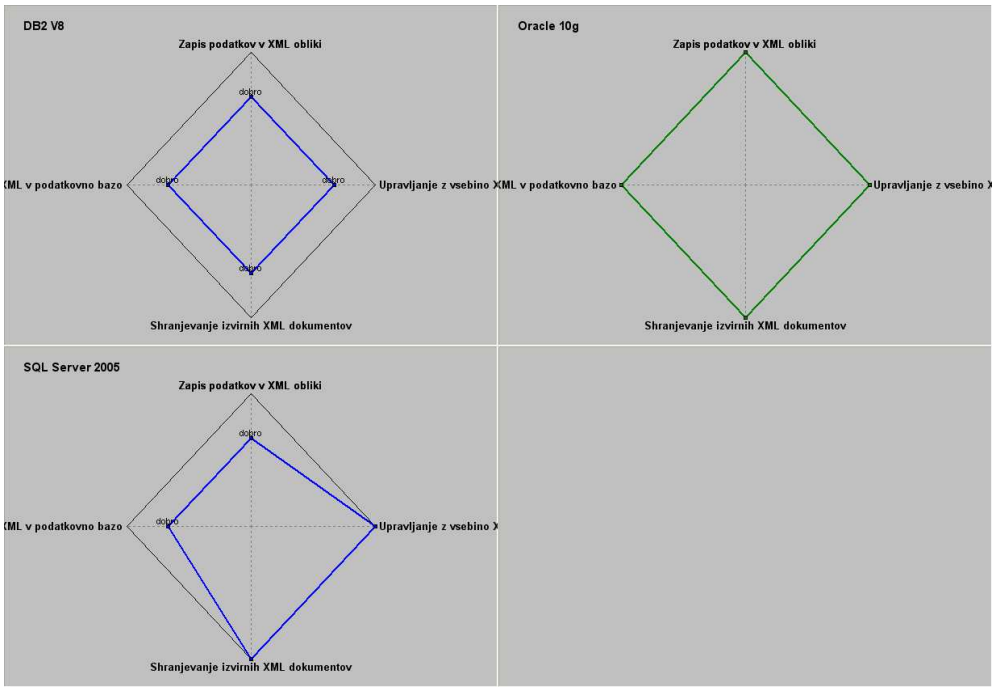
4.3.2 UPARITEV PODATKOV

Vse variante podpirajo preslikavo XML podatkovnega modela v relacijski. V vseh primerih je meta-shema napisana v XML jeziku. Pri DB2 je meta-shema napisana kot DAD datoteka, kar ocenjujem kot slabost, v ostalih dveh primerih je v zapisu XML sheme po standardu W3C.

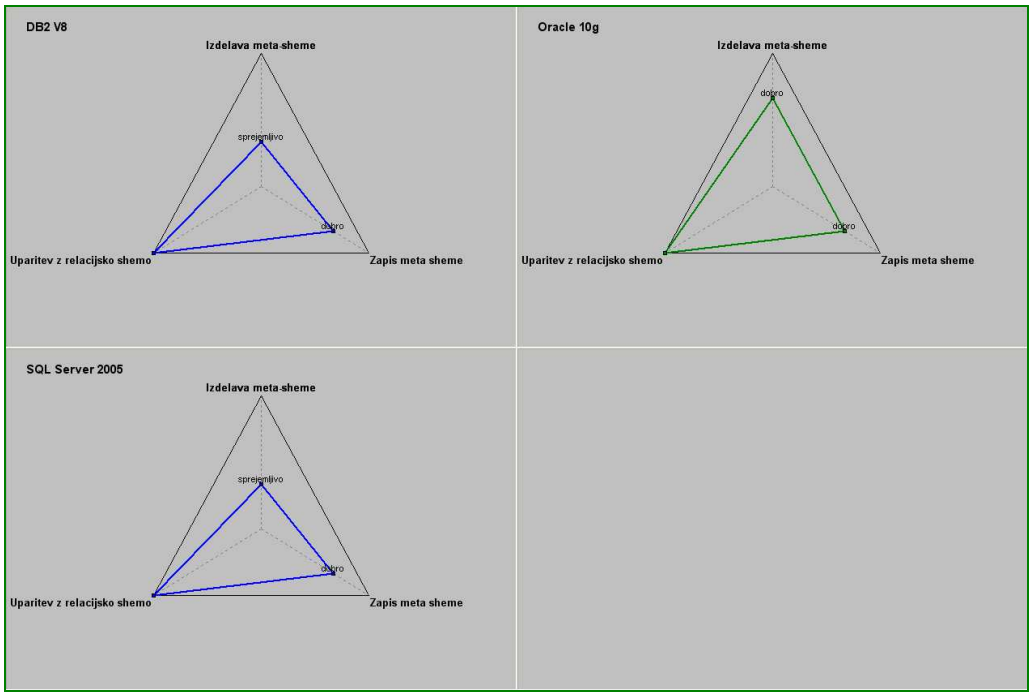
V primeru Oracle in SQL Server obstoječo XML shemo dopolnimo z ustreznimi SQL atributi, ki določajo preslikavo, kar predstavlja prednost, saj meta-sheme ni potrebno izdelati od začetka. Oracle ob registraciji XML sheme poskrbi za izdelavo privzete

(angl. default) meta-sheme in SQL objektov. Slika 40 predstavlja ocene na področju preslikave podatkov.

Slika 39: Ovrednotene zahteve po upravljanju podatkov



Slika 40: Ovrednotene zahteve po preslikavi podatkov



Nobena od variant ne ponuja učinkovitega orodja za preslikavo zahtevnejših XML shem v relacijski model, kar smatram za pomanjkljivost.

4.3.3 KONTROLA PODATKOV

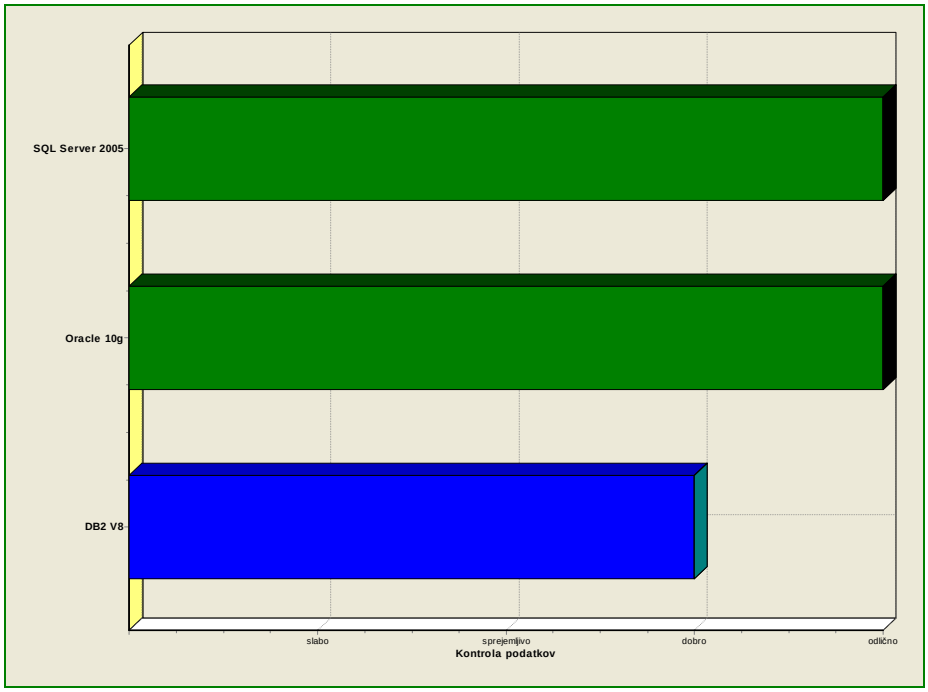
Oracle in SQL Server podpirata uporabo XML sheme po W3C standardu.

DB2 podpira delo z XML shemo v omejenem obsegu. Večinoma uporablja še DTD datoteke za določanje strukture XML dokumenta. Isto velja za zapis meta-sheme.

Oracle in SQL Server podpirata uporabo obeh razčlenjevalnikov: SAX in DOM. DB2 podpira delo s razčlenjevalnikom SAX, iz česar sledi slabša ocena.

Ocene za področje kontrole podatkov prikazuje slika 41.

Slika 41: Ovrednotene zahteve glede kontrole podatkov

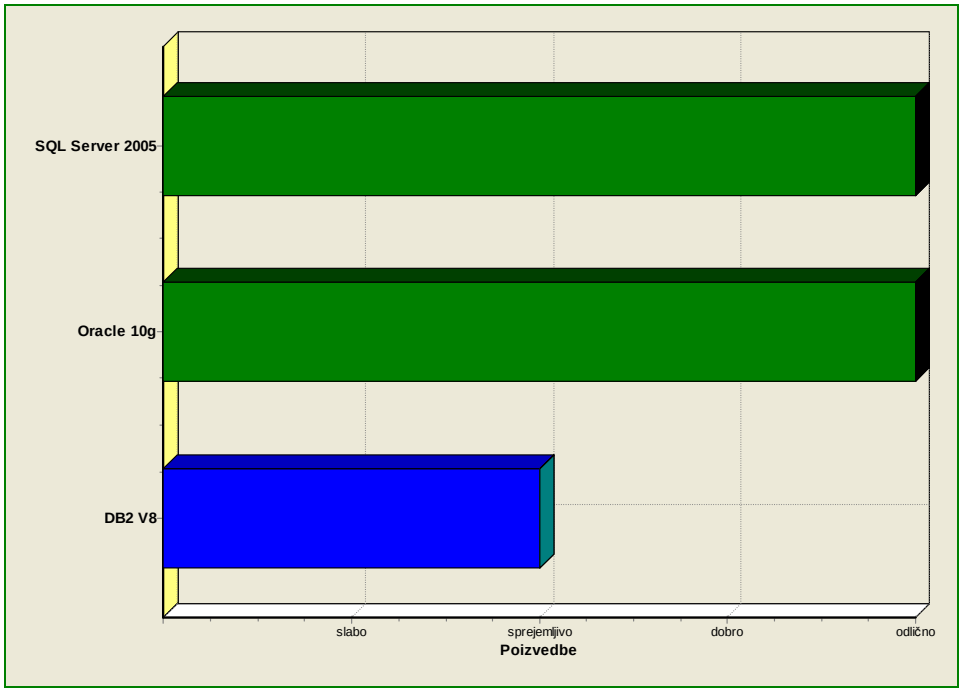


4.3.4 POIZVEDBE

S kriterijema »podpora poizvedovalnim jezikom« in »optimiranje poizvedb s pomočjo indeksov« ugotavljam, v kolikšni meri sistem za upravljanje baz podatkov nudi podporo poizvedovalnim jezikom, temelječih na tehnologiji XML ter ali sistem za upravljanje baz podatkov podpira uporabo indeksov pri iskanju podatkov.

Ovrednotene zahteve prikazuje slika 42. DB2 XML Extender ne podpira poizvedb, napisanih v XQuery in FLRW izrazov. Prav tako ni podpore za iskanje po polnem besedilu. Obstaja le možnost izdelave indeksov na stranskih tabelah.

Slika 42: Ovrednotene zahteve glede poizvedb po podatkih



Oracle in SQL Server v celoti podpirata XQuery, pripadajoče FLWR izraze in iskanje po polnem besedilu.

5. PRIMER IMPLEMENTACIJE UPORABE XML V SUBP

Namen tega poglavja je potrditi ali ovreči teoretična spoznanja, pridobljena v prejšnjih poglavjih in na ta način ovrednotiti praktično uporabnost podpore tehnologiji XML v sistemih za upravljanje baz podatkov. Z uporabo sistema za upravljanje baz podatkov DB2 bom pokazal implementacijo uporabe jezika XML v podatkovni bazi.

Za primer iz prakse sem si izbral prijavljanje v obvezno zdravstveno zavarovanje (v nadaljevanju: prijavljanje). Le-to trenutno poteka z uporabo papirnatih obrazcev. Podatke za izmenjavo bom zapisal v XML obliki in jih zapisal v relacijsko podatkovno bazo.

Moram še omeniti, da ne bom modeliral celotnega poslovnega procesa prijavljanja, saj bi se v tem primeru preveč oddaljil od namena te naloge in zelo verjetno krepko prekoračil predvideni obseg tega dela. Za želeno ponazoritev povsem zadošča segment tega modela, na katerem bom prikazal zapis XML podatkov v podatkovno bazo.

5.1. ANALIZA PRIMERA V PRAKSI

5.1.1 UVOD

Zavod za zdravstveno zavarovanje Slovenije (v nadaljevanju: Zavod), ustanovljen na podlagi Zakona o zdravstvenem varstvu in zdravstvenem zavarovanju izvaja obvezno zdravstveno zavarovanje. Zavod za pokojninsko in invalidsko zavarovanje, Zavod za zdravstveno zavarovanje Slovenije in Zavod RS za zaposlovanje so leta 1980 začeli izvajati projekt o racionalizaciji sporočanja in zbiranja istovrstnih podatkov za vodenje in vzdrževanje administrativnih evidenc o zaposlenih osebah in zavarovancih (Lavrih, 2001, str. 2).

Od 1.1.1987 vsi navedeni zavodi in statistični urad uporabljajo podatke, zbrane iz skupnih obrazcev M-1, M-2, M-1A, M-3, M-3A in M-DČ, ki služijo za enotno prijavljanje oseb v zdravstveno, pokojninsko in invalidsko zavarovanje.

V zadnjih letih so se pojavile pobude o posodobitvi nekaterih Zavodovih storitev in komunikacij s partnerji. Ena od aktualnih nalog je tudi vzpostavitev elektronskega posredovanja prijav, odjav in sprememb za obvezno zdravstveno, pokojninsko in invalidsko zavarovanje.

5.1.2 ZBIRANJE IN ANALIZA ZAHTEV

Obrazce izpolnjujejo in vlagajo zavezanci za plačevanje prispevka za socialno zavarovanje. Za osebe v delovnem razmerju in po teh osebah zavarovane družinske člane prijavo izvajajo njihovi delodajalci. Osebe, ki so same zase zavezanci za plačilo prispevka (samostojni podjetniki, lastniki podjetij, nosilci kmetijske dejavnosti, vrhunski športniki, …), se prijavijo same. Podrobno so zavezanci določeni tudi za druge kategorije zavarovancev (brezposelni, prejemniki denarne pomoči, vojaški invalidi, …).

Za zbiranje prijav in vnos podatkov so zadolžene prijavno - odjavne službe Zavoda. Prijave lahko posredujejo neposredno v prijavno - odjavno službo ali jih pošljejo po pošti. Prijavno - odjavna služba po uspešno opravljeni prijavi vrača izvod potrjenega obrazca vlagatelju kot dokazilo, da je bila prijava vložena.

Podatki, vneseni na Zavodu, se interno uporabljajo za potrebe izvajanja postopkov obveznega zdravstvenega zavarovanja. Isti podatki so podlaga za osvežitev podatkov na karticah zdravstvenega zavarovanja, ki predstavljajo medij za prenos podatkov o zavarovancih in njihovem zdravstvenem zavarovanju k izvajalcem zdravstvenih storitev.

Potrebno je posodobiti poslovni proces prijavljanja z obrazci in vzpostaviti učinkovit elektronski komunikacijski kanal med Zavodom in zavezanci, t.j. vsebino obrazcev zapisati v elektronski obliki, jih zapisati v obstoječo relacijsko bazo podatkov in na ta način vzpostaviti model elektronskega poslovanja med podjetji (B2B).

Podatki, ki se pojavljajo na obrazcu, so podatki o zavezancih, zavarovanih osebah in njihovih družinskih članih ter zavarovanjih.

5.2. PODATKOVNI NAČRT

Konceptualni načrt predstavlja model problema, pisno in vidno predstavitev razmerij med gradniki sistema. Konceptualni načrt vsebuje glavne lastnosti temeljnih gradnikov.

Koraki pri oblikovanju konceptualnega načrta (Welzer, 2005, str. 15):

1. podatkovna analiza in zbiranje zahtev,
2. oblikovanje E-R modela in
3. normalizacija

5.2.1 KONCEPTUALNI MODEL

Podatkovna analiza vsebuje opredelitev skupin uporabnikov in področij uporabe, analizo operativnega okolja in zahtev procesiranja ter proučitev izvorov informacij in podatkov.

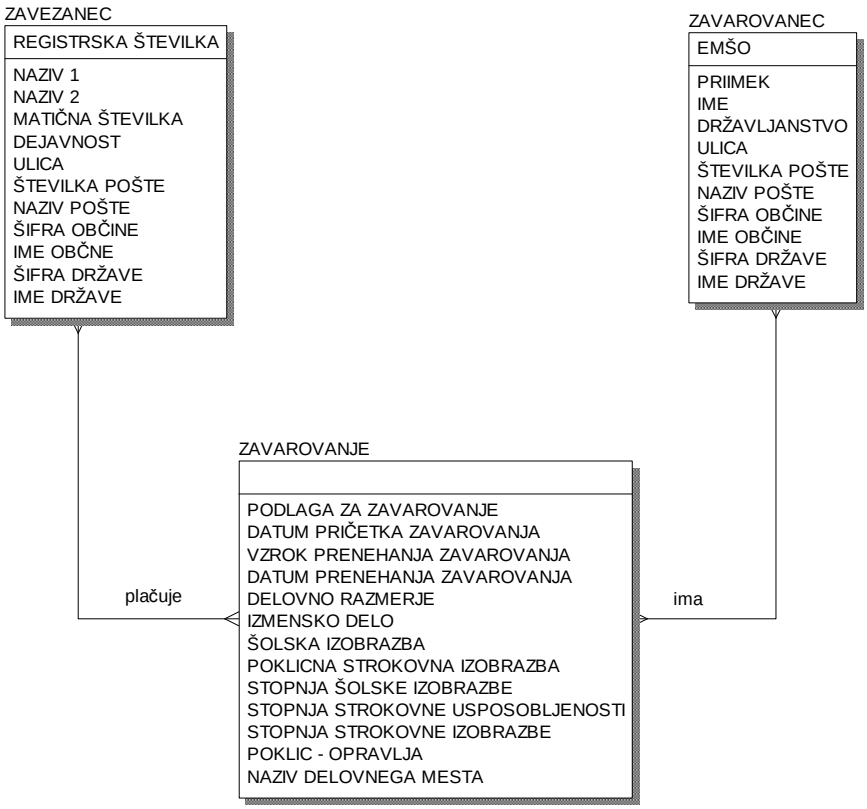
V našem primeru se bom osredotočil na proučitev izvorov informacij in podatkov, saj ne gre za klasično programsko rešitev, kjer uporabniki vnašajo podatke, temveč za elektronsko izmenjavo podatkov med podjetji, kjer uporabniki praktično nimajo stika z aplikacijo.

Vir podatkov predstavljajo že omenjeni M obrazci, ki služijo za enotno prijavljanje oseb v zdravstveno, pokojninsko in invalidsko zavarovanje. Na osnovi tega lahko izdelam konceptualni model, ki ga prikazuje slika 43.

Za podatkovno modeliranje smo uporabili orodje ERwin Data Modeler.

Na osnovi proučitve izvorov informacij in podatkov lahko izluščim tri sklope podatkov, ki v modelu predstavljajo entitete: zavezanec, zavarovanec in zavarovanje.

Slika 43: Konceptualni model



Podrobnejše informacije o entitetah konceptualnega modela se nahajajo v prilogi 9.1.

5.2.2 LOGIČNI NAČRT

Logični načrt je preslikava konceptualnega načrta v podatkovni model sistema za upravljanje baz podatkov.

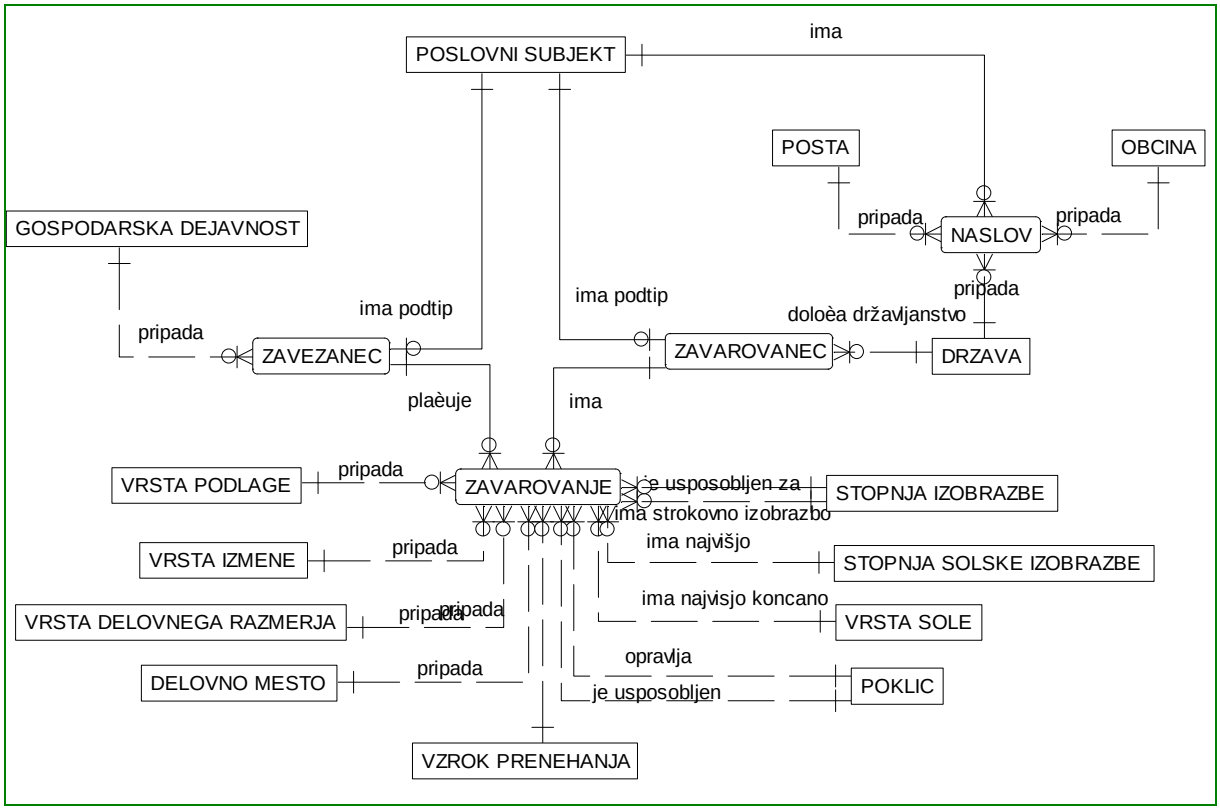
Model, ki ga prikazuje slika 44, je normaliziran oblika konceptualnega modela. Narejen je v James Martinovi notaciji in prikazuje celoten model prijavljanja v obvezno zdravstveno zavarovanje.

Ker je model preveč zahteven in presega namen prikaza zapisa XML podatkov v relacijsko podatkovno bazo, bom v logičnem modelu uporabil le osnovne entitete.

Podrobnejše informacije o entitetah konceptualnega modela se nahajajo v prilogi 9.2.

Podrobnejše informacije fizičnem modelu najdemo v prilogi 9.3.

Slika 44: Entitetno-relacijski model prijavljanja v obvezno zdravstveno zavarovanje



5.3. VPELJAVA FIZIČNEGA NAČRTA IN XML OBJEKTOV

Na osnovi logičnega modela sem izdelal fizični načrt podatkovne baze ter XML shemo, ki je podlaga za elektronsko izmenjavo podatkov. Podatke iz XML sheme je potrebno upariti z fizičnim načrtom, kar storimo v meta-shemi. Na koncu še zapišemo podatke iz XML dokumenta v podatkovno bazo.

5.3.1 IZDELAVA IN PRIPRAVA PODATKOVNE BAZE

Podatkovno bazo izdelamo na osnovi logičnega načrta. V tem primeru sem izdelal tabele ter glavne in pomožne indekse. Za izdelavo objektov uporabimo DDL (angl. data definition language) ukaze (slika 45).

5.3.2 IZDELAVA XML SHEME

Na osnovi podatkovnega načrta sem z orodjem XML Spy izdelal XML shemo. V tem postopku pride do delne denormalizacije, saj to pripomore k berljivosti XML dokumenta. Prav tako pride do izraza objektna naravnanost XML sheme, saj lahko določimo nekatere sestavljene XML elemente (npr. Naslov), ki se pojavlja tako pri zavezancu kot zavarovancu in s tem omogočimo njihovo ponovno uporabnost.

Slika 45: DDL ukazi za izdelavo podatkovne baze

```
db2 "create table Naslovi(STSUBJEK CHAR(13) not null primary key, TXULICAH
CHAR(30) not null, SIPOSTE SMALLINT not null, TXPOSTE CHAR(30) not null,
SIOBCINE SMALLINT  not null, TXOBCINE CHAR(30) not null, SIDRZAVE SMALLINT
not null, TXDRZAVE CHAR(30) not null)"

db2 "create table Zavarovanci(TXMATSTE  CHAR(13) not null primary key,
SIDRZVLJ SMALLINT not null, TXPRIIME CHAR(24) not null, TXIMEOSE CHAR(18)
not null)"

db2 "create table Zavarovanja(TXMATSTE CHAR(13) not null, TXREGZAV CHAR(10)
not null, SIPODLAG  SMALLINT not null, DTZACZAV DATE not null, SIVZRPRE
SMALLINT, DTKONZAV  DATE, SIDELRAZ SMALLINT not null, SIIZMDEL  SMALLINT
not null, SISLSIZB SMALLINT not null, SIPKLIZB SMALLINT not null, SISTSLIZ
SMALLINT not null, SISTSTUS SMALLINT not null, SISTSTIZ SMALLINT not null,
SIPOKLIC SMALLINT not null, TXDELMES CHAR(30) not null)"

db2 "create table Zavezanci(TXREGZAV CHAR(10) not null primary key,
TXNAZIV1 CHAR(30) not null, TXNAZIV2 CHAR(30) not null, TXMATSTE CHAR(10)
not null, SIDEJAVN SMALLINT)"
```

Ker DB2 V8 po dokumentaciji ne podpira XML sheme, je bilo potrebno XML shemo preoblikovati v DTD datoteko (slika 46). Pri tem se seveda izgubi večina možnosti, ki jih XML shema ponuja.

Pri oblikovanju meta-sheme sem naletel na določene omejitve pri poimenovanju elementov, saj le-ta ne dopušča elementov z istimi imeni, kar se pojavi pri uporabi kompleksnih tipov, kot je naslov. Težave podrobneje opisujem v poglavju 5.3.3.

5.3.3 IZDELAVA META-SHEME

V meta-shemi uparimo podatke iz XML dokumenta s podatkovno bazo.

IBM v ta namen ponuja dve orodji, ki pa nista integrirani v podatkovno bazo in ju je potrebno namestiti ločeno:

- **XML Extender Administration tool:**
  orodje omogoča izdelavo meta-sheme (v primeru DB2 je to v obliki DAD datoteke). Uporaba orodja je nerodna in ni uporabniku prijazna, zato sem ga uporabil samo v testne namene. Meta-shemo za naš primer sem zaradi tega v celoti napisal ročno.

- **DAD Checker:**
  orodje omogoča preverjanje strukture DAD datoteke. Ker nima grafičnega vmesnika ga poganjamo iz ukazne vrstice. Kot rezultat se pojavi XML datoteka napak.

Slika 46: DTD datoteka za elektronsko prijavo v zavarovanje

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Tadej Poljsak
(ZZZS IC) -->
<!--DTD generated by XML Spy v4.4 U (http://www.xmlspy.com)-->
<!ELEMENT PosiljkaPIZZ (Zavezanec)>
<!ELEMENT Zavezanec (NazivPrviDel, NazivDrugiDel?, MatStevPRSZapos?,
RegistrskaStevilka, Dejavnost, Naslov, Obrazec_M1_M2_M3*)>
<!ELEMENT NazivPrviDel (#PCDATA)>
<!ELEMENT NazivDrugiDel (#PCDATA)>
<!ELEMENT MatStevPRSZapos (#PCDATA)>
<!ELEMENT RegistrskaStevilka (#PCDATA)>
<!ELEMENT Dejavnost (#PCDATA)>
<!ELEMENT Naslov (Ulica, SifraPoste?, KrajPoste, SifraObcine?, ImeObcine?,
SifraDrzave, ImeDrzave)>
<!ELEMENT Obrazec_M1_M2_M3 (Obrazec, Zavarovanec, Zavarovanje)>
<!ELEMENT Ulica (#PCDATA)>
<!ELEMENT Ulica2 (#PCDATA)>
<!ELEMENT SifraPoste (#PCDATA)>
<!ELEMENT SifraPoste2 (#PCDATA)>
<!ELEMENT KrajPoste (#PCDATA)>
<!ELEMENT KrajPoste2 (#PCDATA)>
<!ELEMENT SifraObcine (#PCDATA)>
<!ELEMENT SifraObcine2 (#PCDATA)>
<!ELEMENT ImeObcine (#PCDATA)>
<!ELEMENT ImeObcine2 (#PCDATA)>
<!ELEMENT SifraDrzave (#PCDATA)>
<!ELEMENT SifraDrzave2 (#PCDATA)>
<!ELEMENT ImeDrzave (#PCDATA)>
<!ELEMENT ImeDrzave2 (#PCDATA)>
<!ELEMENT Obrazec (Vrsta, VrstaPrijave?, DatumPriprave, OpombaZavezanca1?,
OpombaZavezanca2?, VzrokSpremembe?)>
<!ELEMENT Zavarovanec (Emso, Drzavljanstvo, Priimek, Ime, StalniNaslov)>
<!ELEMENT Zavarovanje (Podlaga, DatumPricetka, VzrokPrenehanja?,
DatumPrenehanja?, DelovnoRazmerje?, IzmenskoDelo?, SolskaIzobrazba?,
PoklicnaStrokovnaIzobrazba?, StopnjaSolskeIzobrazbe?,
StopnjaStrokovneUsposobljenosti?, StopnjaStrokovneIzobrazbe?,
PoklicOpravlja?, NazivDelovnegaMesta?)>
<!ELEMENT Vrsta (#PCDATA)>
<!ELEMENT VrstaPrijave (#PCDATA)>
<!ELEMENT DatumPriprave (#PCDATA)>
<!ELEMENT OpombaZavezanca1 (#PCDATA)>
<!ELEMENT OpombaZavezanca2 (#PCDATA)>
<!ELEMENT VzrokSpremembe (#PCDATA)>
<!ELEMENT Emso (#PCDATA)>
<!ELEMENT Drzavljanstvo (#PCDATA)>
<!ELEMENT Priimek (#PCDATA)>
<!ELEMENT Ime (#PCDATA)>
<!ELEMENT StalniNaslov (Ulica2?, SifraPoste2?, KrajPoste2, SifraObcine2?,
ImeObcine2?, SifraDrzave2, ImeDrzave2)>
<!ELEMENT Podlaga (#PCDATA)>
<!ELEMENT DatumPricetka (#PCDATA)>
<!ELEMENT VzrokPrenehanja (#PCDATA)>
<!ELEMENT DatumPrenehanja (#PCDATA)>
```

```
<!ELEMENT DelovnoRazmerje (#PCDATA)>
<!ELEMENT IzmenskoDelo (#PCDATA)>
<!ELEMENT SolskaIzobrazba (#PCDATA)>
<!ELEMENT PoklicnaStrokovnaIzobrazba (#PCDATA)>
<!ELEMENT StopnjaSolskeIzobrazbe (#PCDATA)>
<!ELEMENT StopnjaStrokovneUsposobljenosti (#PCDATA)>
<!ELEMENT StopnjaStrokovneIzobrazbe (#PCDATA)>
<!ELEMENT PoklicOpravlja (#PCDATA)>
<!ELEMENT NazivDelovnegaMesta (#PCDATA)>
```

Pri izdelavi meta-sheme sem naletel na težave pri podvajanju XML elementov. Orodje namreč javi napako, ko naleti na dva elementa z enakim imenom. Kar pomeni, da uporaba kompleksnih tipov ni podprta. Posledica tega je sprememba XML sheme (DTD datoteke) in preimenovanje določenih XML elementov.

Meta-shemo za primer prijavljanja v obvezno zdravstveno zavarovanje prikazuje slika 47.

Izdelava meta-sheme je zahtevno opravilo, še posebej, če ni podprto z ustreznimi grafičnimi orodji. Kompleksnost meta-sheme narašča z zahtevnostjo XML dokumenta in podatkovne baze. Metodološko sodi to opravilo na področje modeliranja in projektiranja. Ko pa skušamo to izvesti v praksi, naletimo na slabo podporo pri tovrstnih orodjih, ki naj bi bila integrirana v podatkovni bazi. Ročna izdelava meta-sheme zato trenutno še vedno sodi v domeno programerjev.

5.3.4 PRIPRAVA PODATKOVNE BAZE

Podatkovno bazo je potrebno pripraviti na delo z XML dokumenti, t.j. omogočiti dodatne funkcionalnosti za delo z XML podatki. Ob kreiranju podatkovne baze le-te niso omogočene. To storimo z ukazom »dxxadm enable_db« ali »dxxadm enable_collection«.

Ob pripravi podatkovne baze na delo z XML dokumenti nastane nekaj dodatnih tabel (tabela 21).

5.3.5 ZAPIS XML V PODATKOVNO BAZO

Za zapis XML podatkov kot XML zbirko v podatkovno bazo sem uporabil dve shranjeni proceduri, ki jih prinaša XML Extender:

- »dxxShredXML« in
- »dxxInsertXML«.

Slika 47: Meta-shema v obliki DAD datoteke

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dada\dad.dtd">
<DAD>
  <dtdid>m123dc.dtd</dtdid>
  <validation>NO</validation>
  <Xcollection>
    <prolog>?xml version="1.0"?</prolog>
    <doctype>!DOCTYPE Order SYSTEM "c:\dad\m123dc.dad"</doctype>
    <root_node>
      <element_node name="PosiljkaPIZZ">
        <RDB_node>
          <table name="Naslovi" key="STSUBJEK"/>
          <table name="Zavarovanci" key="TXMATSTE"/>
          <table name="Zavarovanja" key="TXMATSTE TXREGZAV"/>
          <table name="Zavezanci" key="TXREGZAV"/>

          <condition>Zavarovanci.TXMATSTE=Naslovi.STSUBJEK AND
                     Zavezanci.TXREGZAV=Naslovi.STSUBJEK AND
                     Zavarovanja.TXMATSTE=Zavarovanci.TXMATSTE AND
                     Zavarovanja.TXREGZAV=Zavezanci.TXREGZAV</condition>
        </RDB_node>

        <element_node name="Zavezanec">
          <element_node name="NazivPrviDel">
            <text_node>
              <RDB_node>
                <table name="Zavezanci"/>
                <column name="TXNAZIV1" type="CHAR(30)"/>
              </RDB_node>
            </text_node>
          </element_node>

         ...

        </element_node> <!-- konec elementa Zavezanec -->
      </element_node> <!-- konec elementa PosiljkaPIZZ -->
    </root_node>
  </Xcollection>
</DAD>
```

dxxShredXML

To proceduro uporabimo za občasno zapisovanje vsebine elementov/atributov v XML dokumentu v relacijske tabele.

Predpogoji za uporabo te procedure so obstoj vseh tabel, ki nastopajo v DAD datoteki in ujemanje podatkovnih tipov, navedenih v DAD datoteki. Shranjena procedura ima dva vhodna parametra: DAD datoteko in XML dokument (slika 48). Procedura nato zapiše XML podatke v relacijske tabele in vrača dve vrednosti: povratno kodo in povratno sporočilo.

Tabela 21: Dodatne tabele ob pripravi podatkovne baze na delo z XML dokumenti

Tabela	Opis
DTD_REF	Tabela vsebuje informacije o aktualnih DTD datotekah. Prav tako vsebuje vsebino DTD datoteke in število sklicev na DTD datoteko.
XML_USAGE	Tabela vsebuje informacije o XML stolpcih v podatkovni bazi.

Slika 48: Primer klica shranjene procedure »dxxShredXML«

```
EXEC SQL CALL DB2XML.dxxShredXML(
»m123dc.dad«,
»m123dc.xml«,
:povratnaKoda,
:povratnoSporocilo);
```

dxxInsertXML

To proceduro uporabimo za pogosto zapisovanje vsebine elementov/atributov v XML dokumentu v relacijske tabele.

Predpogoji za uporabo te procedure so enaki kot v prvem primeru. Shranjena procedura ima dva vhodna parametra: XML zbirko in XML dokument. Procedura nato zapiše XML podatke v relacijske tabele in vrača dve vrednosti: povratno kodo in povratno sporočilo. Primer uporabe prikazuje slika 49.

Slika 49: Primer klica shranjene procedure »dxxInsertXML«

```
EXEC SQL CALL DB2XML.dxxInsertXML(
»m123dc_zbirka«,
»m123dc.xml«,
:povratnaKoda,
:povratnoSporocilo);
```

6. UGOTOVITVE

V tem poglavju so strnjene ugotovitve, pridobljene v poglavjih, ki sestavljajo jedro tega dela. Zaradi preglednosti so temeljne ugotovitve namenoma zbrane na koncu, v posebnem poglavju. To bralcu omogoča hiter vpogled v cilj in namen tega dela, ne da bi se bilo potrebno prebijati preko vseh poglavij.

Pričnem z rezultati analize razlik med tehnologijo XML in sistemi za upravljanje podatkovnih baz. Nadaljujem z ugotovitvami primerjalne analize glavnih izdelkov na tržišču podatkovnih baz glede podpore XML in končam z izsledki uporabe XML v podatkovni bazi na primeru iz prakse.

6.1. RAZLIKE MED PARADIGMAMA

Raziskava je pokazala, da dejansko obstajajo razlike med sodobnim elektronskim poslovanjem in relacijskimi podatkovnimi bazami. Poslovne potrebe in pritiski narekujejo podjetjem hitre in učinkovite odzive na spremembe v okolju. Podatkovne baze, ki se nahajajo v jedru informacijskih sistemov podjetij bodo zaradi tega doletele velike spremembe.

Ugotovitev:

1. Razlike med relacijskim in polstrukturiranim podatkovnim modelom ne bodo izginile.

2. Glavnino težav lahko pričakujemo pri preslikavi obeh podatkovnih modelov, t.j. pri izdelavi meta-sheme.

6.2. PRIMERJALNA ANALIZA SUBP

V času pojava XML tehnologije okrog leta 2000 so manjša podjetja izkoristila praznino na tržišču in ponudila specializirane podatkovne baze, ki so podpirale vse funkcionalnosti za delo z XML podatki. V tem času so se pojavili izdelki kot sta Ipedo XML Database in Tamino XML Server. Veliki ponudniki programske opreme so tedaj še spali na lovorikah.

Vendar pa sodobno elektronsko poslovanje prinaša preveč sprememb, da bi glavni igralci na trgu relacijskih podatkovnih baz ostali križem rok. Dejstvo je, da vsi obravnavani ponudniki sistemov za upravljanje baz podatkov počasi konvergirajo k podatkovnim bazam s popolno podporo XML tehnologiji. Vsaka nova verzija podatkovne baze tako prinaša nove XML funkcionalnosti.

To je zelo ugodno za podjetja, saj investicija v dodatno ozko usmerjeno programsko orodje ni več potrebna. Nakup takšnega orodja namreč stane nekaj deset tisoč ameriških dolarjev (Ipedo XML Database 29,000$ na procesno enoto, Tamino XML Server 45,000$ na procesno enoto (Dyck, 2002, str. 2)). Relacijska baza (sistem za upravljanje baz podatkov) sedaj poleg obstoječih in neizpodbitnih lastnosti ponuja (skoraj) vse kar potrebujemo za delo z XML podatki.

Primerjalna analiza je pokazala, da v času raziskave noben od proizvajalcev ne podpira vseh potrebnih funkcionalnosti. Iz literature je namreč razvidno, da nobena od obravnavanih podatkovnih baz ne ponuja učinkovitega in prijaznega orodja za obojestransko preslikavo relacijskega modela v XML. Manjka torej podpora za premostitev največje razlike med paradigmama.

Še najslabše ocene si je prislužilo orodje DB2 V8, kjer XML podpora še ni podpora integrirana v podatkovno bazo in DB2 Extender predstavlja zgolj njeno nadgradnjo. Prav tako manjka podpora poizvedovalnemu jeziku XQuery.

Oracle 10g in SQL Server 2005 se v vseh obravnavanih zahtevah XML tehnologiji bolj približata.

Ugotovitve:

1. Vse tri obravnavane podatkovne baze izpolnjujejo temeljne zahteve za delo z XML tehnologijo. Pričakovati je, da se bodo funkcionalnosti v naslednjih verzijah popolnoma zlile s ozko namenskimi podatkovnimi bazami. Še vedno ni zadostne podpore za preslikavo obeh podatkovnih modelov, t.j. izdelavo meta-sheme.

2. Na tržišču ostaja odprta niša za podjetja, ki so ozko specializirana na integracijo poslovnih procesov in projektov elektronskega poslovanja ter bolj pripravljena prisluhniti potrebam strank. Investicija v takšno orodje je za podjetje, ki vpeljuje elektronsko poslovanje, priporočljiva, saj ni pričakovati razvoja podatkovnih baz na tem področju. Določene takšne funkcionalnosti zasledimo že v orodjih za modeliranje (Erwin) oz. orodjih za oblikovanje XML dokumentov (XML Spy).

3. Slabost odločitvenega modela predstavlja dejstvo, da sem se pri izbiri kriterijev in ocenjevanju le-teh odločal samostojno, na osnovi prebrane literature. To slabost je mogoče odpraviti z uporabo Delphi metode (Ryerson University, 2005), kjer v procesu odločanja sodeluje več strokovnjakov, kar pripomore k natančnejšim odločitvam. Prav tako sem naletel na določene težave pri priporočilu o ortogonalnosti kriterijev, saj popolne ortogonalnosti nisem uspel zagotoviti.

4. Odločitveni model ni prenosljiv v drugačno okolje, saj zajema predvsem kriterije, povezane z XML tehnologijo. Na izbiro sistema za upravljanje baz podatkov v praksi vpliva še množica drugih dejavnikov, ki pa se od primera do primera razlikujejo. Tempo razvoja sistemov za upravljanje baz podatkov in tehnologije nasploh narekuje pogosto revizijo odločitvenega modela, tako kriterijev, kot ocen posameznih kriterijev.

6.3. PRIMER IMPLEMENTACIJE UPORABE XML V SUBP

Šele uporaba orodja v praski pokaže le-to v pravi luči. Objavljena dokumentacija (angl. white papers) in priročniki na spletu mnogokrat ne odražajo dejanskega obnašanja orodja ob uporabi. Nepodprte funkcionalnosti orodja običajno niso omenjene ali pa so slabo opisane. Na ta način lahko bralca navedejo k napačnim odločitvam.

K izbiri DB2 kot testnega orodja me je vodilo dejstvo, da je to orodje že vpeljano v našem podjetju in predstavlja temelj našega informacijskega sistema. Izsledki tega poglavja so posledica nekajtedenskega boja s podatkovno bazo in preizkušanja XML funkcionalnosti.

Ugotovitve:

1. Uporaba zmogljivih in uporabniku prijaznih razvojnih orodij za preslikavo XML sheme v relacijski model je nuja. Ročno uparjanje podatkov vodi v napake in nepravilnost modela. Poleg tega je potrebno upoštevati, da XML dokumente in meta-sheme pogosto razvijajo projektanti in ne programerji, ki so bolj vešči dela z bogatimi grafičnimi orodji.

2. Z zahtevnostjo XML podatkovnega modela eksponentno narašča tudi kompleksnost meta-sheme. To dejstvo je potrebno upoštevati že pri razvoju XML dokumentov. Enostavneje je bolje.

3. Pri zapisu XML podatkov v podatkovno bazo sem naletel na težave pri istoimenskih elementih, ki se v dokumentu pojavljajo na različnih mestih, saj orodje ni dopuščalo izdelave meta-sheme s takšnimi elementi. Tudi takšne posebnosti je potrebno upoštevati pri oblikovanju XML dokumentov.

4. Elektronsko poslovanje s svojimi značilnostmi odločilno vpliva na oblikovanje podatkovne baze. Posebnosti, ki jih prinaša XML tehnologija je potrebno upoštevati že pri prehodu iz konceptualnega modela v logični in ga po potrebi prilagoditi XML dokumentu.

5. V kolikor imamo opravka z zahtevnim modelom in obstoječimi podatki je bolje zgraditi vzporeden relacijski podatkovni model, ki se bolje prilega XML modelu. Še posebej se to nanaša na tuje ključe, ki se pojavljajo med podatki in vsebinske kontrole, ki so pri zapisu XML težko izvedljive. Podatke iz vzporednega modela nato ustrezno preverimo in prepišemo v obstoječ podatkovni model.

6. Globalizacija bo pripomogla k razvoju mednarodnih področnih standardov (besednjakov) na področju XML. Ti bodo prinesli standardne bolj ali manj kompleksne XML sheme za posamezne dokumente. Verjetnost, da bo podatkovni model teh dokumentov vsaj približno podoben našim obstoječim podatkom je minimalna. Pričakujemo lahko, da bodo ti besednjaki na razpolago v določenih imenikih (podobno, kot to pri spletnih storitvah rešuje UDDI) in ne bodo vključeni v sisteme za upravljanje baz podatkov. Pričakujemo lahko, da bodo ti imeniki razdeljeni po gospodarskih panogah (zavarovalništvo, avtomobilska industrija, ipd…). Podjetja bodo standardne XML sheme po potrebi vključevala v svoj informacijski sistem kot običajne, doma razvite sheme.

7. ZAKLJUČEK

Razširljiv označevalni jezik (XML) zaradi svojih lastnosti, kot so prilagodljivost, berljivost in prenosljivost, postaja standard v elektronskem poslovanju. Količina podatkov, izmenjana med poslovnimi subjekti in zapisana v XML obliki, narašča eksponentno. Določene raziskave (Fingar, 2002, str. 1) kažejo, da uporaba podatkovnih baz s popolno podporo XML tehnologiji znatno (do 70%) prispeva h krajšanju poslovnih procesov in s tem ugodno vpliva na konkurenčno prednost podjetja. Spletne storitve, temelječ na XML tehnologiji, predstavljajo temeljni kamen prihodnosti elektronskega poslovanja. Z globalizacijo se odpira množica možnih poslovnih partnerjev v neskončnost. To le še prispeva k pomenu standardizacije in možnosti kontrole v elektronskem poslovanju, kar prinaša XML tehnologija.

XML se danes uporablja predvsem kot oblika zapisa podatkov pri elektronski izmenjavi podatkov ali za predstavitev obstoječih podatkov, zapisanih v relacijskih podatkovnih bazah v XML obliki. Z nadaljnjim razmahom elektronskega poslovanja se bo pojavila tudi potreba po hrambi podatkov, zapisanih v XML obliki, ter poizvedovanju po takšnih podatkih.

Vsi trije obravnavani sistemi za upravljanje baz podatkov že podpirajo osnovne funkcionalnosti za delo z XML podatki. Te funkcionalnosti se nanašajo predvsem na shranjevanje XML podatkov v relacijsko podatkovno bazo. Relacijske podatkovne baze vztrajno konvergirajo k XML tehnologiji, saj se izdelovalci programske opreme za upravljanje podatkov zavedajo potreb, ki jih narekuje elektronsko poslovanje. In le vprašanje časa je, kdaj bo popolnoma izginila razlika med relacijskimi podatkovnimi bazami ter bazami s popolno podporo XML tehnologiji. To pomeni, da bodo relacijske baze podprle oba podatkovna modela. Na drugi strani pa to ne pomeni, da bodo izginile tudi razlike med obema podatkovnima modeloma.

Prav tako lahko pričakujemo, da bodo v nekaj letih izginile razlike med prevladujočimi relacijskimi sistemi za upravljanje baz podatkov. Nabor XML funkcionalnosti, ki jih relacijske podatkovne baze ponujajo, se namreč tako hitro spreminja, da so članki, ki datirajo dve leti v preteklost, že zgodovina.

Največja pomanjkljivost, ki jo očitam obravnavanim sistemom za upravljanje baz podatkov, je premostitev razlik med obema podatkovnima modeloma, ki je udejanjena v obliki meta-sheme. V nobenem primeru namreč nisem naletel na učinkovito orodje za izdelavo meta-sheme. Zahtevnost le-te narašča s kvadratom zahtevnosti podatkovnega modela, zato oblikovanje meta-sheme brez učinkovitih orodij ni smiselno. Izdelave meta-sheme postaja domena projektantov in oblikovalcev informacijskih sistemov in ne programerjev. Sistemi za upravljanje baz

podatkov določene, večinoma neučinkovite, rešitve na tem področju že ponujajo. Revolucionarnega razvoja na tem področju ni pričakovati. Smatramo lahko, da bodo to tržno nišo zasedli izdelovalci programske opreme, ki se ukvarjajo z integracijo programski rešitev in poslovnih procesov.

Uporaba namenskih baz s popolno podporo XML tehnologiji, kot sta Ipedo ali Tamino, je smiselna samo v naslednjih primerih:

- ko integracija z obstoječimi informacijskimi rešitvami ni potrebna in
- ko je investicija v takšno orodje bistveno manjša od investicije v relacijsko podatkovno bazo;

Dopolnjena XML shema se uveljavlja kot standard pri izdelavi XML sheme. V tem primeru gre za dopolnitev obstoječe XML sheme z oznakami, ki povedo pot preslikave v relacijski podatkovni model. Tako Oracle kot Microsoft že podpirata takšen način izdelave meta-sheme. Tudi IBM je že oznanil namero, da bo v prihodnjih verzijah DB2 dopolnjena XML shema zamenjala obstoječo DAD datoteko.

Uporabi XML tehnologije v elektronskem poslovanju se zaradi prednosti, ki jih prinaša, dolgoročno ne moremo izogniti. Na osnovi prebrane literature lahko rečem, da je število uporabnikov XML že preseglo kritično maso, s čimer je ta tehnologija prešla v fazo uporabe. Ne glede na XML možnosti, ki jih sistemi za upravljanje baz podatkov že ponujajo, nas še vedno čaka najtrši oreh v obliki premostitve vrzeli, ki obstaja med obema podatkovnima modeloma. Z ustrezno izbiro orodja na tem področju pa lahko bistveno vplivamo na dolžino trajanja in porabljena sredstva projektov elektronskega poslovanja ali integracije poslovnih procesov.

8. LITERATURA IN VIRI

Literatura

1. Abiteboul Serge, Buneman Peter, Suciu Dan: Data on the Web. San Francisco: Morgan Kaufmann Publishers, 2000, 258 str.

2. Abiteboul Serge, Hull Richard, Vianu Victor: Foundations of Databases. Reading, Addison-Wesley Publishing Company, 1995, 685 str.

3. Appelquist Daniel K.: XML and SQL: developing Web applications. Boston: Addison-Wesley, 2002, 224 str.

4. Balmin Andrey, Papakonstantinou Yannis: Storing and querying XML data using denormalized relational databases. The VLDB Journal, 2005, 14, str. 30-49.

5. Banerjee Sandeepan: XML Query (XQuery) Support in Oracle Database 10g Release 2. White Paper. Redwood Shores: Oracle Corporation, 2005, 44 str.

6. Barbosa Denilson: On Storage Methods for Semistructured Data Web. [URL: http://pages.cpsc.ucalgary.ca/~denilson/docs/ssd_storage.pdf], 15.05.2005.

7. Bourret Ronald: XML and Databases. [URL: http://www.rpbourret.com/xml/XMLAndDatabases.htm], 20.05.2005.

8. Bourret Ronald: XML / Database Links. [URL: http://www.rpbourret.com/xml/XMLDBLinks.htm], 21.05.2005.

9. Bourret Ronald: XML Database Products. [URL: http://www.rpbourret.com/xml/XMLDatabaseProds.htm#xmlenabled], 16.05.2005.

10. Bates Chris: Web Programming. West Sussex: John Wiley & Sons Ltd., 2002, 601 str.

11. Champion Michael: Storing XML in Databases. eAI Journal, 2001, str. 53-55.

12. Connoly Thomas, Carolyn Begg, Anne Strachan: Database systems. Harlow: Addison Wesley Longman Limited, 1998, 1093 str.

13. DeJesus Edmund X: XML enters the DBMS arena. Computerworld, Framingham, 34 (2000), 44, str. 80-82.

14. Drake Mark, Banerjee Sandeepan, Lee Geoff: Oracle Database 10g Release 2 XML DB. Oracle White Paper. Oracle Corporation. Redwood Shores, 2005, 97 str.

15. Dyck Timothy: XML database doubts. eWeek, 18 (2001), 47.

16. Erl Thomas: Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services. B.k.: Prentice Hall PTR, 2004, 560 str.

17. Fingar Peter: The forth tier. Internet World, 8 (2002), 4, str. 24.

18. Goldman Roy et al.: A Standard Textual Interchange Format for the Object Exchange Model (OEM).
[URL: http://www-db.stanford.edu/~mchughj/oemsyntax/oemsyntax.html], 13.10.2005.

19. Gottschalk Karl: Web Services architecture overview.
[URL: http://www-128.ibm.com/developerworks/webservices/library/w-ovr/?dwzone=webservices], 13.06.2005.

20. Graves Mark: Designing XML databases. Upper Saddle River: Prentice Hall, 2002, 655 str.

21. Guardalben Giovanni, Atre Shaku: Integrating XML and Relational Database Technologies.
[URL:
http://www.hitsw.com/products_services/whitepapers/integrating_xml_rdb/index.html], 12.04.2005.

22. Harold Elliotte Rusty, Means W. Scott: XML in a nutshell, 3. edition. Sebastopol: O'Reilly, 2004, 689 str.

23. Hsieh Chang-Tseh, Lin Binshan: Impact of standardization on EDI in B2B development. Industrial Management & Data Systems, 104 (2003), 1, str. 68–77.

24. Jerman Blažič Borka et al.: Elektronsko poslovanje na internetu. Ljubljana, GV Založba, 2001, 207 str.

25. Jurič B. Matjaž: Osnove jezika XML. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, 2001, 41. str.

26. Kappel Gerti, Kapsammer Elisabeth, Retschitzegger Werner: Integrating XML and Relational Database Systems. World Wide Web: Internet and Web Information Systems, Dunaj, 2004, 7, str. 343-384.

27. Katz Howard: An introduction to XQuery.
[URL: http://www-106.ibm.com/developerworks/xml/library/x-xquery.html], 20.05.2005.

28. Lee Geoff: Mastering XML Generation in Oracle Database 10g Release 2. Oracle White Paper. Oracle Corporation, Redwood Shores, 2005, 22 str.

29. Lee Geoff: Oracle Database 10g Release 2 XML DB Technical Overview. Oracle White Paper. Oracle Corporation, Redwood Shores, 2005a, 31 str.

30. Lehti Patrick: Design and Implementation of a Data Manipulation Processor for an XML Query Language. Diplomsko delo. Darmstadt: Technische Universität Darmstadt, 2001, 82 str.

31. Lim Billy B. L., Wen H. Joseph: The impact of next generation XML. Information Management & Computer Security, 2002, 10/1, str. 33-40.

32. M2 Communications Ltd: All databases XML enabled by 2009? Mid Tier XML Market Opportunities, Strategies, and Forecasts, 2004 to 2009 to its offerings, 2004.

33. Malcolm Graeme: Retrieving XML from SQL Server with Mapping Schemas. Apress, 2003, 25 str.

34. Mertz David: XML Matters: Putting XML in context with hierarchical, relational,and object-oriented models. [URL: http://www-106.ibm.com/developerworks/xml/library/x-matters8/], 02.07.2005.

35. Mittra S. Sitansu: Principles of relational database systems. New Jersey: Prentice-Hall Inc., 1991, 323 str.

36. Mohorič Tomaž: Načrtovanje relacijskih podatkov baz. Ljubljana: BI-TIM, 1997, 206 str.

37. Muench Steve: Building Oracle XML Applications. B.k.: O'Reilly, 810 str.

38. Nicola Matthias: XML seminar. Opatija: SQL Adria Conference, 2004, 212 str.

39. Page Colin: XML and DB2. Ljubljana: SQL Adria Conference, 2004.

40. Ramon Lawrence: The space efficiency of XML, Information and Software Technology, Iowa City, 46 (2004), 11, str. 753-759.

41. Rhyno Art: XML and relational databases: uses and opportunities for libraries. OCLC Systems & Services, Ontario, 18 (2002), 2, str. 97-103.

42. Rob Peter, Coronel Carlos: Database systems: design, implementation, and management. Boston: Thomson, 2004, 795 str.

43. Rooney Paula, Darrow Barbara: XML database war. CRN, Jericho, 2001, 943, str. 3-5.

44. Russell David: Exploring the RDB to XML Mapping Editor. IBM WebSphere Developer Technical Journal, Toronto, april 2002.

45. Scardina Mark, Chang Ben, Wang Jinyu: Oracle Database 10g XML & SQL: Design, Build, & Manage XML Applications in Java, C, C++, & PL/SQL. B.k.: McGraw-Hill Osborne Media, 2004, 600 str.

46. Seng Jia-Lang, et al.: An analytic study of XML database techniques. Industrial Management & Data Systems, 2003, 103/2, str. 111-120.

47. Smith D. Alan: Exploring potential strategic impacts of XML-related technologies. Industrial Management & Computer Security, Pittsburgh, 11/2 (2003), str. 92-100.

48. Sunderic Dejan: SQL Server 2000 Stored Procedure & XML Programming, 2. edition. B.k.: McGraw-Hill Osborne Media, 2003, 696 str.

49. Tae-Sun Chung, Hyoung-Joo Kim: Techniques for the evaluation of XML queries. Data & Knowledge Engineering, Seoul, 46 (2003), 2, str. 225-246.

50. Tatarinov Igor et al.: Storing and Querying Ordered XML Using a Relational Database System. [URL: http://www.cs.cornell.edu/people/jai/papers/OrderedXML.pdf], 17.07.2005.

51. Trippe Bill: XML hits the big time: Major database players get in XML. EContent, Wilton, 25 (2002), 9, str. 24-28.

52. Turban Efraim, et al.: Electronic Commerce 2004: A Managerial Perspective. B.k.: Prentice Hall, 2004a, 752 str.

53. Turban Efraim, McLean Ephraim, Wetherbe James: Information technology for management - transforming business in the digital economy, 3. edition. B.k.: John Wiley, 2004b, 784 str.

54. Vakali Athena, Catania Barbara, Maddalena Anna: XML Data Stores: Emerging Practices. IEEE INTERNET COMPUTING, marec - april 2005, str 62-68.

55. Vithanala K. Prasadarao: Introduction to XQuery in SQL Server 2005. [URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsql90/html/sql2k5_xqueryintro.asp], 12.07.2005.

56. Vlist Eric van Der: XML Schema. Sebastopol: O'Reilly, 2002, 379 str.

57. Welzer Družovec Tatjana, Brumen Boštjan: Podatkovno modeliranje. Bled: 2005, 105 str.

Viri

1. AJAR: XSL Transformation. [URL:http://www.ajar.ch/de/aktuell/4d_2004/web_xml.html], 23.11.2005.

2. Cincom: Choosing an XML Database Solution. [URL: http://www.cincom.com/global/eng/news-

room/pressreleases/detail.html?id=348], 18.05.2005.

3. Gartner media relations.
[URL: http://www.dataquest.com/press_gartner/quickstats/databases.html], 06.07.2005.

4. Hewlett-Packard: Web services concepts - a technical overview.
[URL: http://www.hpmiddleware.com/downloads/pdf/web_services_tech_overview.pdf], 08.07.2005.

5. IBM Redbook: XML Extender Administration and Programming Version 8. B.k., 2002, 413 str.

6. IBM: DB2 Product Family.
[URL: http://www-306.ibm.com/software/data/db2/index.html], 14.10.2005.

7. Lavrih Vlasta et al.: Analiza možnosti vzpostavitve sistema elektronskega posredovanja prijav, odjav in sprememb za obvezno zdravstveno, pokojninsko in invalidsko zavarovanje, 2001, 22 str.

8. Microsoft Corporation: Annotations to the XDR Schema.
[URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsql/ac_mschema_1u7n.asp], 13.07.2005.

9. Microsoft Corporation: XML Options in Microsoft SQL Server 2005.
[URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnanchor/html/sqlserver.asp], 14.10.2005.

10. OASIS. [URL: http://www.oasis-open.org], 22.06.2005.

11. Oracle technology center.
[URL: http://www.oracle.com/technology/products/database/oracle10g/index.html], 14.10.2005.

12. Pal Shankar, et al.: XML Best Practices for Microsoft SQL Server 2005.
[URL: http://msdn.microsoft.com/SQL/sqlreldata/XML/default.aspx?pull=/library/en-us/dnsql90/html/sql25xmlbp.asp], 22.06.2005.

13. Pal Shankar, et al.: Performance Optimizations for the XML Data Type in SQL Server 2005.
[URL: http://msdn.microsoft.com/SQL/sqlreldata/XML/default.aspx?pull=/library/en-us/dnsql90/html/sqloptxml.asp#sqloptxml_topic4], 22.06.2005.

14. Pal Shankar, Fussell Mark, Dolobowsky Irwin: XML Support in Microsoft SQL Server 2005.
[URL: http://msdn.microsoft.com/library/en-us/dnsql90/html/sql2k5xml.asp],

22.06.2005.

15. Rambhia Ajay M: Parsing an XML Document. [URL: http://www.informit.com/articles/article.asp?p=27858&seqNum=3&rl=1], 22.11.2005.

16. RTV Slovenija: RSS novice. [URL: http://www.rtvslo.si/feeds/03.xml], 11.05.2005.

17. Ryerson University: The Delphi Method [URL: http://www.ryerson.ca/~mjoppe/ResearchProcess/841TheDelphiMethod.htm], 15.11.2005.

18. Štrumbelj Trontelj Irena et al.: Navodilo za prijavo podatkov o pokojninskem in invalidskem ter zdravstvenem zavarovanju, zavarovanju za starševsko varstvo, zavarovanju za primer brezposelnosti in o sklenitvi delovnega razmerja (M-1, M-2, M-3, M-3A, M-DČ), 1. izdaja. Ljubljana: Statistični urad Republike Slovenije, 2005, 139 str.

19. W3C/XML. [URL: http://www.w3.org/XML/], 15.05.2005.

20. W3C/XPath logical expressions. [URL: http://www.w3.org/TR/xquery-semantics/#id-logical-expressions], 21.06.2005.

21. W3C/XQuery operators. [URL: http://www.w3.org/TR/xquery-operators/], 20.06.2005.

22. W3Schools/SOAP tutorial. [URL: http://www.w3schools.com/soap/soap_intro.asp], 22.11.2005.

23. W3Schools/XPath operators. [URL: http://www.w3schools.com/xpath/xpath_operators.asp], 22.06.2005.

24. W3Schools/XPath syntax. [URL: http://www.w3schools.com/xpath/xpath_syntax.asp], 22.06.2005.

25. W3Schools/XSLT Browsers. [URL: http://www.w3schools.com/xsl/xsl_browsers.asp], 23.06.2005.

26. W3Schools/WSDL tutorial. [URL: http://www.w3schools.com/wsdl/default.asp], 23.11.2005.

27. World Wide Web Consortium. [URL: http://www.w3.org/XML/], 7.5.2005.

9. PRILOGE

9.1. ENTITETE V KONCEPTUALNEM MODELU

Seznam entitet v konceptualnem modelu

Entiteta	Opis
Zavarovanec	Zavarovanec je oseba, ki je vključena v sistem obveznega zdravstvenega zavarovanja.
Zavarovanje	Zavarovanje je predpisan poslovni odnos med zavarovano osebo, zavezancem in ZZZS.
Zavezanec	Zavezanec je pravna ali fizična oseba, ki je dolžna plačevati z zakonom predpisane prispevke za obvezno zdravstveno (socialno) zavarovanje. Za osebe v delovnem razmerju in po teh osebah zavarovane družinske člane plačujejo prispevke njihovi delodajalci. Osebe, ki so same zase zavezanci za plačilo prispevka so samostojni podjetniki, lastniki podjetij, nosilci kmetijske dejavnosti, vrhunski športniki. Podrobno so zavezanci določeni tudi za druge kategorije zavarovancev (brezposelni, prejemniki denarne pomoči, vojaški invalidi, …).

Podatki entitete Zavezanec

Podatek	Opis
Registrska številka	Registrska številka zavezanca za prispevke, dodeljena poslovnemu subjektu oziroma poslovni enoti, kjer opravlja oseba delo v času, na katerega se nanašajo sporočeni podatki.
Naziv 1	Ime in sedež zavezanca za prispevek (poslovnega subjekta oziroma poslovne enote, obrata, podružnice ali poslovnega prostora), kjer oseba opravlja delo, oziroma enote organa, ki je zavezancu priznal pravico.
Naziv 2	Drugi del imena in sedeža zavezanca.
Matična številka	Desetmestna matična številka Poslovnega registra Slovenije (PRS), ki jo sestavlja sedemmestna matična številka poslovnega subjekta in trimestna zaporedna številka podružnice oziroma drugega dela poslovnega subjekta, kjer oseba dejansko dela. Vpisana je na obvestilu o identifikaciji in razvrstitvi po dejavnosti za poslovne subjekte in podružnice oziroma druge dele poslovnih subjektov, ki ga izda AJPES.
Dejavnost	Šestmestna šifra dejavnosti je vpisana na obvestilu o identifikaciji in razvrstitvi po dejavnosti za poslovne subjekte in podružnice oziroma druge dele poslovnih subjektov, ki ga izda AJPES.
Ulica	Ulica in hišna številka.
Številka pošte	Štirimestna poštna številka po šifrantu pošt RS Slovenije.
Naziv pošte	Kraj pošte.
Šifra občine	Šifra občine iz seznama občin RS.
Ime občine	Šifra in naziv občine iz seznama občin RS.
Šifra države	Šifra države iz seznama držav, v kateri je stalni naslov.
Ime države	Naziv države iz seznama držav, v kateri je stalni naslov.

Podatki entitete Zavarovanec

Podatek	Opis
EMŠO	Enotna matična številka občana
Državljanstvo	Državljanstvo zavarovanca.
Priimek	Priimek zavarovanca.
Ime	Ime zavarovanca.
Ulica	Ulica in hišna številka.
Številka pošte	Štirimestna poštna številka po šifrantu pošt RS Slovenije.
Naziv pošte	Kraj pošte.
Šifra občine	Šifra občine iz seznama občin RS.
Ime občine	Šifra in naziv občine iz seznama občin RS.
Šifra države	Šifra države iz seznama držav, v kateri je stalni naslov.
Ime države	Naziv države iz seznama držav, v kateri je stalni naslov.

Podatki entitete Zavarovanje

Podatek	Opis
Podlaga	Šifra glede na vrsto zavarovanja, ki je podlaga za pokojninsko in invalidsko ter zdravstveno zavarovanje, zavarovanje za starševsko varstvo in zavarovanje za primer brezposelnosti.
Datum začetka zavarovanja	Datum pričetka zavarovanja.
Vzrok prenehanja	Šifra vzroka predčasnega prenehanja zavarovanja.
Datum prenehanja zavarovanja	Datum izteka zavarovanja.
Delovno razmerje	Šifra glede na vrsto delovnega razmerja zaposlene osebe oziroma pripravnika.
Izmensko delo	Šifra glede na razporeditev delovnega časa - izmensko delo.
Šolska izobrazba	Trimestna šifra šolske izobrazbe je vrsta najvišje končane šole. Vpisuje se iz šifranta Klasifikacija šol.
Poklicna strokovna izobrazba	Šifra najvišje priznane poklicne/strokovne izobrazbe zavarovane osebe, ki jo je dosegla z rednim izobraževanjem ali drugim uradno priznanim izobraževanjem. Za izpolnjevanje te rubrike je treba uporabiti šifrant poklicev, ki ga je izdala Zveza skupnosti za zaposlovanje Republike Slovenije in založil Gospodarski vestnik 1984, ter spremembe in dopolnitve Šifranta poklicev, ki jih je pripravil Republiški zavod za zaposlovanje leta 1994, ali Šifrant poklicne in strokovne izobrazbe, ki ga je izdal Zavod Republike Slovenije za zaposlovanje leta 2000, in je najnovejša različica šifranta.
Stopnja šolske izobrazbe	Stopnja šolske izobrazbe je vrsta šole (odsek, smer, program), s katero si je oseba pridobila najvišjo stopnjo v svojem izobraževanju. To pomeni, da je treba pri ugotavljanju in evidentiranju podatkov o stopnji šolske izobrazbe upoštevati samo najvišjo stopnjo šole, ki jo je oseba končala, ne upoštevamo pa šole, ki jo je sicer obiskovala, vendar je ni zaključila. Oseba je končala določeno šolo, če je končala zadnji razred oziroma opravila zadnji izpit (diplomski izpit, diplomsko delo ipd.).
Stopnja strokovne usposobljenosti	Stopnja strokovne usposobljenosti je stopnja zahtevnosti za opravljanje del in nalog po sistemizaciji. S strokovno usposobljenostjo za opravljanje določenih del oziroma nalog po sistemizaciji so mišljena splošna in strokovna znanja, praktična

Podatek	Opis
	usposobljenost in druge lastnosti, ki jih mora imeti oseba, da lahko uspešno opravlja dela in naloge v podjetju oziroma organizaciji. Stopnja strokovne usposobljenosti izhaja iz opisa delovnega mesta, na katerem oseba dela, stopnja strokovne izobrazbe pa je osebna oznaka.
Stopnja strokovne izobrazbe	Strokovna izobrazba je formalnopravna kvalifikacija, ki jo je oseba pridobila z enim od naslednjih možnih načinov: a) z zaključkom ustrezne šole; b) z zaključkom šole ali tečaja, ki mu je z odločbo o stopnji šol, tečajev in izpitov določena stopnja šole ali tečaja; c) s priznanjem stopnje strokovne izobrazbe priučenega (pol kvalificiranega) delavca po preverjanju njegove strokovnosti v podjetju ali drugi organizaciji; d) s strokovnim izpitom za pridobitev stopnje izobrazbe kvalificiranega ali visoko kvalificiranega delavca; e) s posebnim izpitom za priznanje strokovne izobrazbe po predpisih o javnih službah; f) s priznanjem strokovne izobrazbe (naziva) javnim uslužbencem po predpisih o javnih uslužbencih.
Poklic, ki ga opravlja	Sedemmestna šifra poklica iz Standardne klasifikacije poklicev (SKP) (Uradni list RS, 16/2000).
Naziv delovnega mesta	Naziv delovnega mesta, na primer: zidar, usmerjevalec pošte, komercialist na terenu, višji strokovni sodelavec I - sistemski inženir ipd. Če natančen naziv delovnega mesta ni znan, ga je potrebno opisati: sestavljavec električnih delov, upravljavec procesnih linij za predelavo kovin.

9.2. ENTITETE V LOGIČNEM NAČRTU

Seznam entitet v logičnem modelu

Entiteta	Opis
Naslovi	Naslovi poslovnih subjektov. V tej tabeli se nahajajo naslovi zavarovancev in zavezancev.
Zavarovanci	Zavarovanec je oseba, ki je vključena v sistem obveznega zdravstvenega zavarovanja.
Zavarovanja	Zavarovanje je predpisan poslovni odnos med zavarovano osebo, zavezancem in ZZZS.
Zavezanci	Zavezanec je pravna ali fizična oseba, ki je dolžna plačevati z zakonom predpisane prispevke za obvezno zdravstveno (socialno) zavarovanje. Za osebe v delovnem razmerju in po teh osebah zavarovane družinske člane plačujejo prispevke njihovi delodajalci. Osebe, ki so same zase zavezanci za plačilo prispevka so samostojni podjetniki, lastniki podjetij, nosilci kmetijske dejavnosti, vrhunski športniki. Podrobno so zavezanci določeni tudi za druge kategorije zavarovancev (brezposelni, prejemniki denarne pomoči, vojaški invalidi, …).

Entiteta Naslovi

Podatek	Podatkovni tip	Ključ	NULL
Številka subjekta	`CHAR(13)`	Da	Ne
Ulica	`CHAR(30)`		Ne
Številka pošte	`SMALL INTEGER`		Ne
Naziv pošte	`CHAR(30)`		Ne
Šifra občine	`SMALL INTEGER`		Ne
Ime občine	`CHAR(30)`		Ne
Šifra države	`SMALL INTEGER`		Ne
Ime države	`CHAR(30)`		Ne

Entiteta Zavarovanci

Podatek	Podatkovni tip	Ključ	NULL
EMŠO	`CHAR(13)`	Da	Ne
Državljanstvo	`SMALL INTEGER`		Ne
Priimek	`CHAR(24)`		Ne
Ime	`CHAR(18)`		Ne

Entiteta Zavarovanja

Podatek	Podatkovni tip	Ključ	NULL
EMŠO	`CHAR(13)`	Da	Ne
Registrska številka	`CHAR(10)`	Da	Ne
Podlaga	`SMALL INTEGER`		Ne
Datum začetka zavarovanja	`DATE`		Ne

Podatek	Podatkovni tip	Ključ	NULL
Vzrok prenehanja	SMALL INTEGER		Da
Datum prenehanja zavarovanja	DATE		Da
Delovno razmerje	SMALL INTEGER		Ne
Izmensko delo	SMALL INTEGER		Ne
Šolska izobrazba	SMALL INTEGER		Ne
Poklicna strokovna izobrazba	SMALL INTEGER		Ne
Stopnja šolske izobrazbe	SMALL INTEGER		Ne
Stopnja strokovne usposobljenosti	SMALL INTEGER		Ne
Stopnja strokovne izobrazbe	SMALL INTEGER		Ne
Poklic, ki ga opravlja	SMALL INTEGER		Ne
Naziv delovnega mesta	CHAR(30)		Ne

Entiteta Zavezanci

Podatek	Podatkovni tip	Ključ	NULL
Registrska številka	CHAR(10)	Da	Ne
Naziv 1	CHAR(30)		Ne
Naziv 2	CHAR(30)		Ne
Matična številka	CHAR(10)		Ne
Dejavnost	SMALL INTEGER		Ne

9.3. FIZIČNI NAČRT

Seznam tabel v fizičnem načrtu

Entiteta	Opis
Naslovi	Naslovi poslovnih subjektov. V tej tabeli se nahajajo naslovi zavarovancev in zavezancev.
Zavarovanci	Zavarovanec je oseba, ki je vključena v sistem obveznega zdravstvenega zavarovanja.
Zavarovanja	Zavarovanje je predpisan poslovni odnos med zavarovano osebo, zavezancem in ZZZS.
Zavezanci	Zavezanec je pravna ali fizična oseba, ki je dolžna plačevati z zakonom predpisane prispevke za obvezno zdravstveno (socialno) zavarovanje. Za osebe v delovnem razmerju in po teh osebah zavarovane družinske člane plačujejo prispevke njihovi delodajalci. Osebe, ki so same zase zavezanci za plačilo prispevka so samostojni podjetniki, lastniki podjetij, nosilci kmetijske dejavnosti, vrhunski športniki. Podrobno so zavezanci določeni tudi za druge kategorije zavarovancev (brezposelni, prejemniki denarne pomoči, vojaški invalidi, …).

Tabela Naslovi

Podatek	Naziv polja	Podatkovni tip	Ključ	NULL
Številka subjekta	STSUBJEK	CHAR(13)	Da	Ne
Ulica	TXULICAH	CHAR(30)		Ne
Številka pošte	SIPOSTE	SMALL INTEGER		Ne
Naziv pošte	TXPOSTE	CHAR(30)		
Šifra občine	SIOBCINE	SMALL INTEGER		
Ime občine	TXOBCINE	CHAR(30)		
Šifra države	SIDRZAVE	SMALL INTEGER		
Ime države	TXDRZAVE	CHAR(30)		

Tabela Zavarovanci

Podatek	Naziv polja	Podatkovni tip	Ključ	NULL
EMŠO	TXMATSTE	CHAR(13)	Da	Ne
Državljanstvo	SIDRZVLJ	SMALL INTEGER		Ne
Priimek	TXPRIIME	CHAR(24)		Ne
Ime	TXIMEOSE	CHAR(18)		Ne

Tabela Zavarovanja

Podatek	Naziv polja	Podatkovni tip	Ključ	NULL
EMŠO	TXMATSTE	CHAR(13)	Da	Ne
Registrska številka	TXREGZAV	CHAR(10)	Da	Ne
Podlaga	SIPODLAG	SMALL INTEGER		Ne
Datum začetka zavarovanja	DTZACZAV	DATE		Ne
Vzrok prenehanja	SIVZRPRE	SMALL INTEGER		Da
Datum prenehanja	DTKONZAV	DATE		Da

zavarovanja				
Delovno razmerje	SIDELRAZ	SMALL INTEGER		Ne
Izmensko delo	SIIZMDEL	SMALL INTEGER		Ne
Šolska izobrazba	SISLSIZB	SMALL INTEGER		Ne
Poklicna strokovna izobrazba	SIPKLIZB	SMALL INTEGER		Ne
Stopnja šolske izobrazbe	SISTSLIZ	SMALL INTEGER		Ne
Stopnja strokovne usposobljenosti	SISTSTUS	SMALL INTEGER		Ne
Stopnja strokovne izobrazbe	SISTSTIZ	SMALL INTEGER		Ne
Poklic, ki ga opravlja	SIPOKLIC	SMALL INTEGER		Ne
Naziv delovnega mesta	TXDELMES	CHAR(30)		Ne

Tabela Zavezanci

Podatek	Naziv polja	Podatkovni tip	Ključ	NULL
Registrska številka	TXREGZAV	CHAR(10)	Da	Ne
Naziv 1	TXNAZIV1	CHAR(30)		Ne
Naziv 2	TXNAZIV2	CHAR(30)		Ne
Matična številka	TXMATSTE	CHAR(10)		Ne
Dejavnost	SIDEJAVN	SMALL INTEGER		Ne

9.4. KRATICE

Kratica	Pomen
API	Application programming interface
ASCII	American Standard Code for Information Interchange
ASP	Active server pages
B2B	Business to business
BLOB	Binary large object
CLOB	Character large object
CPU	Central Processing Unit
DAD	Document access definition
DBMS	Database management Ssstem
DDL	Data definition language
DDL	Data Definition Language
DOM	Document object model
DTD	Document Type Definition
FLWR	FOR, LET, WHERE in RETURN ukazi v jeziku XQuery
GUI	Graphical User Interface
HTML	HyperText Markup Language
JDBC	Java Data Base Connectivity
LOB	Large object
ODBC	Open Data Base Connectivity
OEM	Object exchange model
OLEDB	OLE Data Base
PDF	Portable document format
RDBMS	Relational Data Base Management System
RSS	Really simple syndication, tudi rich site summary
RTF	Rich text format
SAX	Simple API for XML
SGML	Standard Generalized Markup Language
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
UDDI	Universal Description, Discovery and Integration
UDT	User-defined type
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	The World Wide Web Consortium
WSDL	Web Services Description Language
XML	EXtensible Markup Language

Kratica	Pomen
XPath	XML Path Language
XQuery	XML Query
XSL	EXtensible Stylesheet Language
XSLT	EXtensible Stylesheet Language Transformation
XSU	XML-SQL Utility

9.5. SLOVAR TUJK

Tujka	Pomen
brick-and-mortar	Podjetja, ki poslujejo na tradicionalen način, prodajajo blago preko trgovin in agentov.
caching	Shranjevanje podatkov v predpomnilnik, kjer bodo ob ponovni zahtevi hitreje priklicani.
composition	Sestavljanje, sestava.
computed constructor	Izračunan konstruktor.
database extensions	Programska koda, ki predstavlja nadgradnjo funkcionalnosti podatkovne baze.
data-centric document	Podatkovno usmerjen, dobro strukturiran dokument.
decomposition	Razstavljanje.
direct element constructors	Neposredni konstruktor elementa.
directory servers	Imeniški strežniki.
document-centric documents	Dokumentno usmerjen, slabo strukturiran dokument.
dynamic binding	Dinamično povezovanje
encapsulation	Ograjevanje
full-text index	Indeks za iskanje po polnem besedilu.
full-text search	Iskanje po polnem besedilu.
grid computing	mrežno računalništvo
index	Indeks, seznam vrednosti.
message passing	Posredovanje sporočil
multivalue	Podatek, ki vsebuje več vrednosti.
native XML database	Podatkovne baze s popolno podporo XML tehnologiji.
node	Vozel, vozlišče v dokumentu.
page level locking	Zaklepanje na nivoju podatkovne strani v podatkovni bazi.
parent/child	Odnos med dvema XML atributoma/elementoma.
parser	Razčlenjevalnik.
processing instructions	Navodila za obdelavo XML dokumenta.
rollback	Preklic sprememb opravljenih v podatkovni bazi.
root node	Korensko vozlišče.
round-tripping	Ohranitev vseh lastnosti XML dokumenta pri zapisu v podatkovno bazo.
row level locking	Zaklepanje podatkov v podatkovni bazi na nivoju vrstice.
scalability	Prilagodljivost, prenosljivost

Tujka	Pomen
service description	Opis storitve
side tables	Stranske tabele.
stored procedure	Shranjena procedura.
structural index	Strukturni indeks.
syntax	Skladnja
tag	Oznaka.
two-phase commit	Dvofazna potrditev sprememb v podatkovni bazi.
value index	Indeks vrednosti.
white paper	Dokument, ki obravnava socialno, politično ali drugo tematiko. Pogosto vsebuje razlago arhitekture določene rešitve ali rezultate in ugotovitve analiz.
XML enabled database	Podatkovna baza z delno podporo XML tehnologiji.

9.6. SEZNAM SLIK

Slika 1: Primer podatkov v XML obliki 6
Slika 2: XSL pretvorba 13
Slika 3: XML dokument z označeno XSL datoteko 13
Slika 4: XSL datoteka 14
Slika 5: V HTML zapis preoblikovana XML datoteka 14
Slika 6: Arhitektura spletnih storitev 17
Slika 7: Primer dela WSDL dokumenta 19
Slika 8: Teoretični hierarhični podatkovni model 22
Slika 9: Teoretični relacijski podatkovni model 23
Slika 10: Primer poizvedbe v SQL poizvedovalnem jeziku 24
Slika 11: Teoretični polstrukturiran podatkovni model 25
Slika 12: Sestavljen objekt po OEM notaciji (Goldman, 2005) 26
Slika 13: Podatkovno usmerjen XML dokument 29
Slika 14: RSS datoteka kot dokumentno usmerjen XML dokument 30
Slika 15: Navodila za obdelavo v XML dokumentu 37
Slika 16: Uparitev XML podatkov v relacijski model 38
Slika 17: Meta-shema 39
Slika 18: XML kontruktor elementa 47
Slika 19: Izračunan konstruktor 47
Slika 20: Predpisano zaporedje FLWR ukazov 48
Slika 21: Primer poizvedbe v XQuery 48
Slika 22: Primer uporabe XQuery operatorjev 48
Slika 23: Primer uporabe XQuery logičnih izrazov 49
Slika 24: Delež prodanih licenc po proizvajalcih 50
Slika 25: Uteži posameznih kriterijev v odločitvenem modelu 60
Slika 26: XML dokument, shranjen kot XML zbirka 62
Slika 27: XML dokument, shranjen kot DB2 stolpec 63
Slika 28: Ažuriranje vsebine XML stolpca 63
Slika 29: Ažuriranje vsebine elementa v XML stolpca 64
Slika 30: Brisanje XML dokumentov iz podatkovne baze 64
Slika 31: Primer DAD datoteke za XML stolpec 65
Slika 32: Primer DAD datoteke za XML zbirko 66
Slika 33: Primer uporabe in rezultat SQL/XML funkcij 69
Slika 34: Meta-shema preslikave XML podatkov v relacijske v Oracle XML DB 71
Slika 35: Registracija XML sheme 71
Slika 36: Primer dodajanja XML elementa s pomočjo funkcije »modify« 74
Slika 37: Primer popravljanja vsebine XML elementa s pomočjo funkcije »modify« 74
Slika 38: Rezultati vrednotenja variant 77
Slika 39: Ovrednotene zahteve po upravljanju podatkov 78
Slika 40: Ovrednotene zahteve po preslikavi podatkov 78
Slika 41: Ovrednotene zahteve glede kontrole podatkov 79
Slika 42: Ovrednotene zahteve glede poizvedb po podatkih 80
Slika 43: Konceptualni model 83

Slika 44: Entitetno-relacijski model prijavljanja v obvezno zdravstveno zavarovanje 84
Slika 45: DDL ukazi za izdelavo podatkovne baze 85
Slika 46: DTD datoteka za elektronsko prijavo v zavarovanje 86
Slika 47: Meta-shema v obliki DAD datoteke 88
Slika 48: Primer klica shranjene procedure »dxxShredXML« 89
Slika 49: Primer klica shranjene procedure »dxxInsertXML« 89

9.7. SEZNAM TABEL

Tabela 1: Primerjava značilnosti DTD datoteke in XML sheme 9
Tabela 2: Prednosti SAX razčlenjevalnika 10
Tabela 3: Slabosti SAX razčlenjevalnika 11
Tabela 4: Prednosti DOM razčlenjevalnika 11
Tabela 5: Slabosti DOM razčlenjevalnika 12
Tabela 6: Lastnosti spletnih storitev 16
Tabela 7: Elementi SOAP ovojnice 18
Tabela 8: Elementi WSDL dokumenta 18
Tabela 9: Primerjava funkcionalnosti 20
Tabela 10: Primerjava podatkovnih elementov 27
Tabela 11: Primerjava kontrol podatkov 27
Tabela 12: Prednosti in slabosti podatkovnih baz z delno podporo tehnologiji XML 32
Tabela 13: Prednosti in slabosti podatkovnih baz s popolno podporo tehnologiji XML 34
Tabela 14: Možni pristopi k shranjevanju XML podatkov 35
Tabela 15: Primerjava poizvedb in indeksiranja 44
Tabela 16: Glavni ukazi za izbiro vozlišč v Xpath 46
Tabela 17: Primeri XPath izrazov 46
Tabela 18: Primeri XPath ukazov 46
Tabela 19: Deleži sredstev, potrošenih za shranjevanje podatkov glede na arhitekturo 50
Tabela 20: Tržni deleži podatkovnih baz v odstotkih 50
Tabela 21: Dodatne tabele ob pripravi podatkovne baze na delo z XML dokumenti 89