Medšolsko projektno delo z večparametrskimi odločitvenimi modeli in izmenjavo podatkov XML

Maja Azarov Domajnko[1], Vladislav Rajkovič[2], Marko Bohanec[3,4]

[1] Srednja šola za elektrotehniko in računalništvo, Vegova 4, 1000 Ljubljana, Slovenija
[2] Univerza v Mariboru, Fakulteta za organizacijske vede, Kidričeva cesta 55a, 4000 Kranj, Slovenija
[3] Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenija
[4] Univerza v Ljubljani, Visoka upravna šola, Gosarjeva 5, 1000 Ljubljana, Slovenija

Metoda projektnega in skupinskega dela je pri pouku zelo uspešna. Ker se po vsebini projekti razlikujejo, je zaradi množice kriterijev težavna tako izbira primernih tem kot ocenjevanje končnih izdelkov. S sistemom za podporo pri odločanju, ki smo ga uporabili pri medšolskem projektnem delu, lahko uspešno rešujemo oba zgoraj navedena problema. Zgradili smo dva odločitvena modela. Uporabili smo program za podporo pri odločanju DEXi, ki uporablja za shranjevanje podatkov zapis XML. Raziskali smo možnosti za vnos variant v naš odločitveni sistem z oddaljenih lokacij v obliki XML. Izdelali smo dokumente DTD in sheme za ugotavljanje veljavnosti podatkov (validacijo), ki jih vnašamo v odločitveni model z oddaljenih lokacij.

Ključne besede: projektno delo, odločitveni model, sistem za podporo pri odločanju, DEXi, XML, shema, DTD

1. Uvod

V prispevku opisujemo uporabo odločitvenih modelov za izbiro in ocenjevanje projektnih nalog. Izdelajo jih dijaki, ki so lahko z različnih šol, povezani v delovne skupine. Vsaka skupina se mora dogovoriti o vsebini svojega projektnega dela, ki mora ustrezati nekim vnaprej znanim kriterijem. Pri tem ponavadi prihaja do težav, saj dijaki običajno ne upoštevajo vseh parametrov, ki naj bi vplivali na odločitev pri izbiri. Drugi problem nastopi, ko je treba oceniti projektno delo. Ker gre v splošnem za zahteven izdelek, vpliva na končno oceno veliko kriterijev, ki jih moramo upoštevati.

S pomočjo sistemov za podporo pri odločanju lahko uspešno rešujemo oba navedena problema. Prvi korak je izdelava večkriterijskih odločitvenih modelov. Uporabili smo program za podporo pri odločanju DEXi, ki je bil razvit na Inštitutu Jožef Stefan v Ljubljani v sodelovanju s Fakulteto za organizacijske vede v Kranju (Krapež in dr., 2000; Gams, Bohanec, 2001).

S pomočjo odločitvenih modelov in programa DEXi se lahko dijaki vsak zase odločijo med različnimi možnimi projektnimi temami in vsak zase ocenijo ostale projektne naloge. Pri tem se tudi praktično seznanijo s pristopi in metodami sistemov za podporo pri odločanju.

Ker naj bi dijaki delali v skupinah, ki bi bile po možnosti sestavljene iz dijakov z oddaljenih šol, bi bilo zelo uporabno, če bi lahko vsi dijaki v skupini vpisovali v en sistem za podporo pri odločanju in tako izbrali skupno temo projektne naloge. Prav tako bi lahko vsi dijaki sodelovali pri ocenjevanju projektnih nalog, če bi lahko vsi vpisovali svoje ocene v isti sistem. Problemom odločanja se pridruži še problem prenosa podatkov med dijaki in oddaljenim sistemom za podporo pri odločanju. Za prenos podatkov se v zadnjem času uspešno uveljavlja standard XML (Armstrong, 2000; IBM, 2001).

Dijaki se lahko na preprost in poučen način spoznajo s standardom XML tako, da podatke, ki jih v program DEXi vnašajo z oddaljenih lokacij, prenašajo v obliki XML. Rešiti pa je potrebno probleme, ki jih dosedanja realizacija sistema ne podpira, kot so varnost, preverjanje pravilnosti podatkov in skupinsko odločanje.

2. Odločitveni modeli

Za potrebe našega projektnega dela smo razvili dva večparametrska odločitvena modela. Prvi služi dijakom kot pomoč pri odločanju o izbiri kraja, ki ga bodo predlagali za skupni izlet in zanj izdelali predstavitveni projekt. Drugi model pa obsega kriterije za ocenjevanje projektne naloge in bo v pomoč učitelju pri ocenjevanju projektov in dijakom pri izdelavi projektnih nalog.

2.1 Odločitveni model za izbiro kraja

Pri našem medšolskem projektu se morajo dijaki odločiti o kraju za skupni izlet. Pri tem morajo upoštevati različne parametre kot so: oddaljenost, cena, zanimivosti ipd. Pogosto se zgodi, da dijaki upoštevajo le enega ali nekaj parametrov, na ostale pa pozabijo. Rezultat so popolnoma neživljenjski predlogi. Zato smo izdelali odločitveni model, v katerega bodo dijaki lahko vnesli svoje predloge in se odločali na podlagi vseh potrebnih parametrov.

V modelu smo opredelili kriterije (atribute), ki so pomembni za pravilen izbor kraja. S tem smo odločitveni

problem razgradili na več enostavnih odločitev. Za te atribute smo vnesli vrednosti, ki jih lahko zavzamejo (zaloga vrednosti) in med katerimi se bodo dijaki odločali.

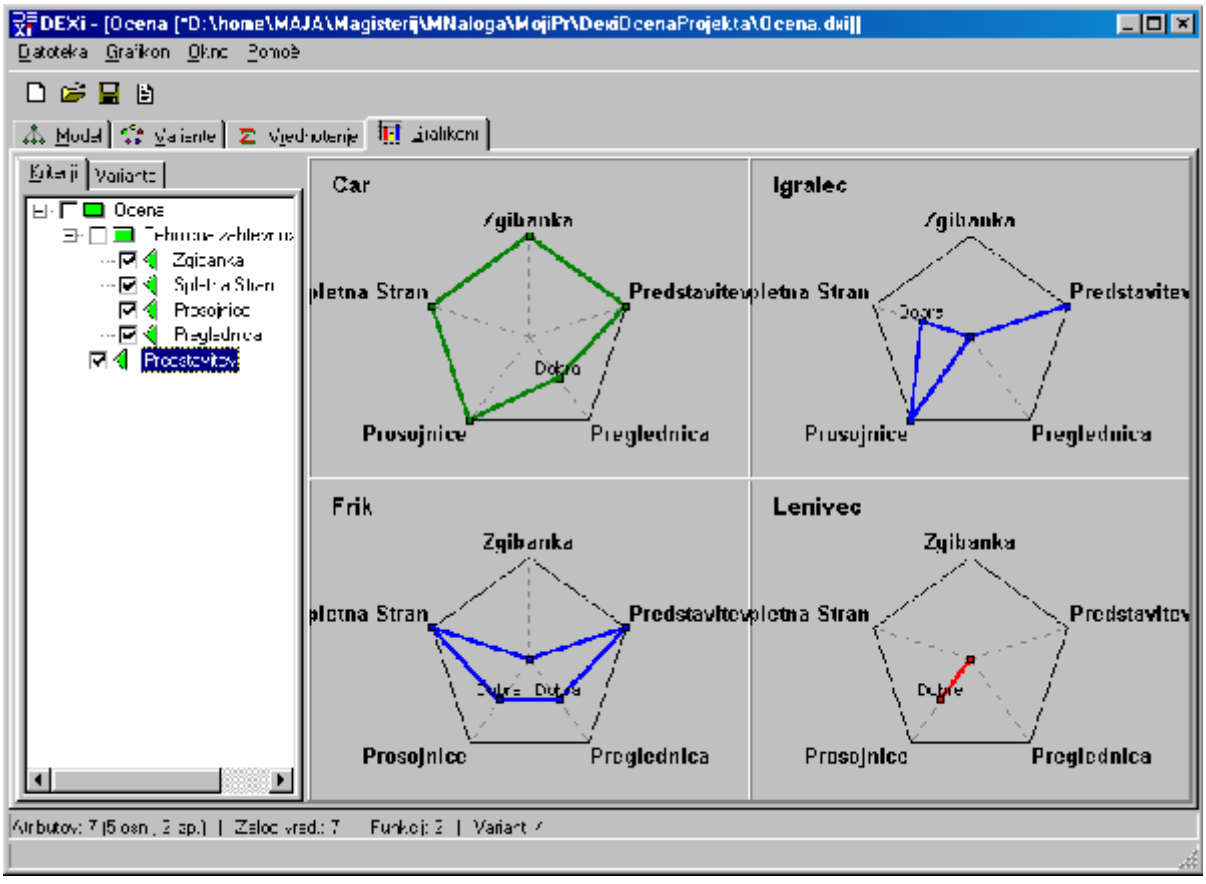
Nato smo določili pravila odločanja (funkcije koristnosti), na podlagi katerih se osnovni atributi združujejo v smiselno povezane skupine in končno v skupno vrednost, na podlagi katere razvrstimo variante po ustreznosti.

2.2 Odločitveni model za oceno projekta

Pri ocenjevanju projektov smo že v prejšnjih letih poskušali vključiti dijake. Izkazalo se je, da ocenjevanje projektov ni prav enostavno opravilo, in da si moramo tudi učitelji vnaprej precej natančno postaviti kriterije, ki vplivajo na končno oceno. Tudi takrat, ko so se dijaki trudili, da bi ocenjevali čim bolj realno, so ponavadi upoštevali le nekatere kriterije. Kot pomoč smo uporabljali ocenjevalne obrazce, ki smo jih pozneje postavili na spletno stran na enega od šolskih strežnikov. Pri določanju skupne ocene smo se omejili na izračun skupne srednje ocene. Razpršenost ocen je bila majhna, v glavnem so se ocene vrtele med "dobro" in "prav dobro".

Ob pripravi modela smo analizirali potek dosedanjih predstavitev projektov in kriterije, ki smo jih uporabili pri ocenjevanju. Ker je kriterijev veliko, bi bilo ročno vnašanje odločitvenih pravil zamudno in nedosledno. Odločili smo se za uporabo uteži, in sicer smo dali večjo težo tistim kriterijem, ki so bili zahtevnejši za izvedbo (npr. izvedbi spletne strani). Seveda pa v modelu nismo mogli upoštevati bolj subjektivnih kriterijev, ki ponavadi tudi vplivajo na oceno, kot so: osebni napredek dijaka, trud, ki ga je vložil glede na svoje sposobnosti in predznanje, sposobnost javnega nastopanja, priljubljenost v skupini, ipd.

Program DEXi nam ustreznost variant, v našem primeru je to uspešnost posameznega dijaka, prikaže tudi grafično. Na sliki 1 vidimo grafični prikaz štirih izmišljenih variant. Prikazane so vrednosti osnovnih atributov. Čim višje so vrednosti atributov, tem večjo površino zavzame omejeni lik na grafu, kar pomeni boljšo izvedbo projektne naloge.



Slika 1: Grafična predstavitev variant

3. Podatki v zapisu XML

Dokument XML sestavljajo posamezni elementi, ki jih omejujeta začetna in končna oznaka (angl. tag). Oznaka vsebuje ime elementa.

Primer elementa:
<NAME>Ljubljana</NAME>

Število oznak v jeziku XML ni omejeno, namenjene pa so opisu pomena in strukture podatkov. Prednost takega zapisa je, da ga lahko razume tako človek (saj vsebuje le

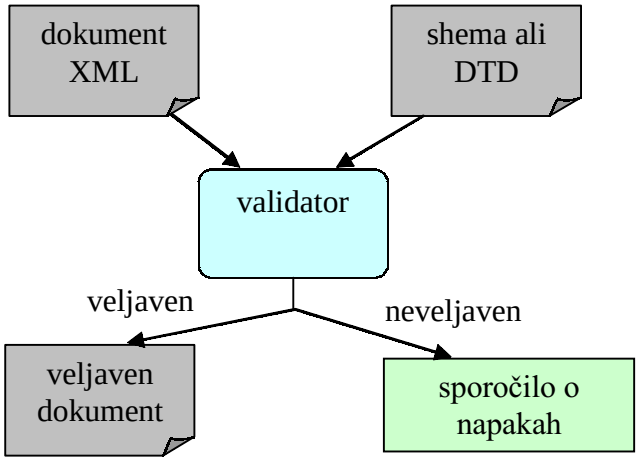
navadno besedilo), kot tudi računalniški program (podatke iz dokumenta lahko uporabi v nadaljnjih obdelavah ali jih pripravi na izpis). Različne binarne formate pa lahko preberemo le s pripadajočimi programi, ki se uporabljajo le v določenem obdobju in okolju.

3.1 Slovnična pravilnost in veljavnost dokumentov XML

Dokumenti XML morajo biti napisani po določenih slovničnih pravilih. Slovnično pravilnost dokumenta preverja razčlenjevalnik (parser). Če dokument vsebuje slovnične napake, nam razčlenjevalnik javi sporočilo o napakah. Dokumenti, ki ne upoštevajo teh pravil, niso slovnično pravilni (well formed) in zato tudi niso veljavni (valid) (Breskvar, 2001).

Ko je dokument slovnično pravilen, je treba ugotoviti še, ali je tudi veljaven. Veljaven dokument XML ima zahtevano strukturo in vnesene vse zahtevane podatke (podobno kot pravilno izpolnjen obrazec). Osnovni način preverjanja veljavnosti dokumenta je s pomočjo dokumenta DTD (Document Type Definition) (W3C Recommendation, 2001). Pozneje so bile namesto dokumentov DTD vpeljane sheme (W3C Recommendation, 2001), ki odpravljajo nekatere pomanjkljivosti dokumenta DTD.

Ko napišemo dokument XML in dokument, ki določa njegovo veljavnost (DTD ali shema), lahko s pomočjo orodij za validacijo (preverjanje veljavnosti) preverimo, če je dokument XML veljaven. V tem koraku bomo npr. odkrili, če so v dokumentu XML uporabljeni napačni tipi podatkov, napačne vrednosti podatkov ali napačna struktura dokumenta. Postopek prikazuje slika 2.



Slika 2: Preverjanje veljavnosti dokumenta XML

3.2 DTD

Dokument DTD (Document Type Definition) določa pravila, po katerih so strukturirani podatki v dokumentu XML. Določiti je treba, katere elemente vsebuje dokument XML, v kakšnem vrstnem redu si sledijo, katere atribute vsebujejo in kakšna je dovoljena vsebina elementov in atributov.

3.3 Sheme

XML shema ima podobno funkcijo kot DTD, vendar pa odpravlja nekatere pomanjkljivosti specifikacije DTD. Prva od njih je, da dokument DTD ni tudi sam napisan v jeziku XML. Dokument DTD ne podpira različnih podatkovnih tipov. Vsebina elementa, ki jo določa DTD, tako ne more biti nič drugega, kot ugnezdeni element ali pa niz znakov.

Sheme podpirajo različne podatkovne tipe:

1. standardne enostavne tipe, npr. nize (string), cela (integer) in realna števila (float), datum (date), logične vrednosti (boolean) in podobno;
2. nestandardne enostavne tipe, ki jih definiramo z elementom “simpleType”;
3. kompleksne tipe, ki jih definiramo z elementom “complexType”.

Shemo lahko v grobem razdelimo na dva dela. En del predstavljajo deklaracije elementov, ki nastopajo v dokumentu XML. Deklaracija elementa poveže element s pripadajočim podatkovnim tipom. Drugi del sheme predstavljajo deklaracije nestandardnih enostavnih in kompleksnih podatkovnih tipov.

4. XML in program za večparametrsko odločanje DEXi

Ker pri projektu lahko sodelujejo dijaki različnih šol, je potrebno poiskati načine, kako bodo tudi oddaljeni dijaki lahko uporabljali skupni sistem za podporo pri odločanju. Sodobna oblika za izmenjavo podatkov med oddaljenimi stranmi je standard XML. DEXi sam že shranjuje podatke o odločitvenem modelu in podatke o vnesenih variantah v obliki XML. V nadaljevanju so predstavljeni načini, na katere bi podatke z oddaljenih računalnikov lahko vnašali v sistem za podporo pri odločanju.

Primer 1 prikazuje tisti del kode XML, ki opisuje vnesene podatke o variantah in se doda v dokument XML, ki hrani podatke o modelu, in sicer med oznaki <OPTION> in </OPTION>.

Primer1:

```
<OPTION>
<NAME>Ljubljana</NAME>
<OPTIONVALUE>
  <ATTRIBUTE>Oddaljenost kraja</ATTRIBUTE>
  <VALUE>pod 50 km</VALUE>
</OPTIONVALUE>
</OPTION>
```

S pravilno definirano kodo XML lahko vnašamo podatke v program DEXi, npr. različne variante z različnimi vrednostmi atributov. Tak način vnosa podatkov je uporaben takrat, kadar vnaša variante več udeležencev odločitvenega procesa, ki so med seboj oddaljeni in delajo v različnih okoljih in na različnih sistemih.

Poglejmo primer skupnega projektnega dela dijakov z dveh oddaljenih šol. Program in model se nahajata na eni od obeh šol, predloge (variante) pa želijo ovrednotiti dijaki obeh šol. Dijaki, ki nimajo nameščenega programa DEXi, variant ne morejo niti vnašati niti vrednotiti. Za prenos podatkov v obliki, neodvisni od programske in strojne opreme, uporabimo jezik XML. Dijaki z oddaljene šole lahko svoje variante pošljejo v partnersko šolo v tej obliki. Pred tem jih morajo seveda na nek način vnesti.

V nadaljevanju opisujemo dva načina za vnos variant z oddaljene lokacije:

- Ročni vpis variant v obliki XML. Variante lahko dodamo neposredno v kodo XML skupnega odločitvenega modela z navadnim urejevalnikom besedil ASCII (vi, emacs, Notepad, edit,...). Paziti moramo na uporabo velikih in malih črk pri oznakah.
- Dijaki v oddaljeni šoli uporabijo kot orodje, ki generira XML, kar program DEXi.

4.1 Ročni vpis variant

Prvi pogoj za uspešen vnos podatkov je, da morajo biti vsi podatki slovnično pravilni. To najenostavneje preverimo kar s prikazom kode v spletnem pregledovalniku, ki podpira prikaz kode XML, npr. Internet Explorer 6 (MSDN Online, 2001). Naslednji korak pa je preverjanje veljavnosti podatkov. Uporabniki (dijaki) lahko vnesejo napačne elemente ali napačne vsebine elementov, ki jih DEXijev odločitveni model ne pozna. Zato je poleg slovnične pravilnosti potrebno tudi preverjanje veljavnosti dokumenta XML. Prav lahko bi se zgodilo, da bi dijak navedel vrednost lastnosti neke variante, ki je ni v zalogi vrednosti, definirani v odločitvenem modelu. Npr. za oddaljenost kraja bi vpisal vrednost "nad 200 km".

Drugo vrsto napake pa predstavlja uporaba neveljavnih elementov. Če bi dijak npr. uporabil namesto elementa <NAME> element <IME>, bi bil dokument ravno tako neveljaven in ga program DEXi ne bi sprejel. Veljavnost XML kode lahko preverjamo na dva načina, z DTD (Document Type Definition) ali s shemo. Shema ali DTD se lahko nahajata na skupnem strežniku, od koder jo lahko naložimo z vseh oddaljenih lokacij, s katerih poteka vnos variant.

4.2 Preverjanje veljavnosti z DTD

Preverjali bomo le veljavnost tistega dela kode XML, ki opisuje variante, ki jih dijaki vnašajo v sistem. Oglejmo si postopek preverjanja veljavnosti za primer 1. V dokumentu DTD določimo, naj korenski element vsebuje elementa NAME in OPTIONVALUE, element NAME lahko vsebuje le besedilo, element OPTIONVALUE pa elementa ATTRIBUTE in VALUE, ki oba vsebujeta le besedilo. Elementov OPTIONVALUE je lahko več:

```
<!ELEMENT OPTION (NAME,OPTIONVALUE*)>
<!ELEMENT NAME (#PCDATA)>
<!ELEMENT OPTIONVALUE (ATTRIBUTE,VALUE)>
<!ELEMENT ATTRIBUTE (#PCDATA)>
<!ELEMENT VALUE (#PCDATA)>
```

Še vedno pa z DTD ne moremo nadzorovati vsebine besedila, ki je vneseno v elemente ATTRIBUTE in VALUE. Tako lahko vnesemo npr. kraj, ki ga v odločitvenem modelu nismo predvideli in ga program DEXi seveda ne bo sprejel.

4.3 Preverjanje veljavnosti s shemo

Shemo začnemo z elementom, ki označuje začetek vsakega dokumenta XML, saj je tudi shema dokument XML.

```
<?xml version="1.0"?>
```

Definicijo sheme začnemo s stavkom, ki definira element "schema" in njegov imenski prostor, npr. "f", ki preprečuje, da bi se zaradi enakih imen elementov različni elementi XML interpretirali enako.

```
<schema xmlns:f='myschem1.xsd'
targetNamespace='myschem1.xsd'>
```

Sledi definicija tipov, ki jih bomo potrebovali v shemi.

Element OPTION ima dva vgnezdena elementa, enako element OPTIONVALUE.

```
<complexType name="typeOPTION">
  <sequence>
   <element ref="f:NAME"/>
   <element ref="f:OPTIONVALUE"/>
  </sequence>
</complexType>
<complexType name="typeOPTIONVALUE">
  <sequence>
   <element ref="f:ATTRIBUTE"/>
   <element ref="f:VALUE"/>
  </sequence>
</complexType>
```

Sledi seznam elementov:

```
<element name="NAME" type="string"/>
<element name="ATTRIBUTE" type="string"/>
<element name="VALUE" type="string"/>
<element name="OPTIONVALUE"
type="f:typeOPTIONVALUE"/>
 <element name="OPTION" type="f:typeOPTION"/>
```

Elementi so tipa »string«, kar ne omejuje njihove vsebine.

Poglejmo na naslednjem primeru še uporabo omejitev, ki omogoča, da omejimo vrednost poljubnega niza na končno število naštetih nizov.

```
<simpleType name="typeNAME">
    <restriction base="string">
        <enumeration value="Ljubljana" />
        <enumeration value="Trbovlje" />
    </restriction>
</simpleType>
```

Shemo zaključimo z zaključitvenim elementom za shemo:

```
</schema>
```

Kot vidimo, je ročni vnos variant je sicer poučen in tranparenten, za vnos večjih količin podatkov, še posebej, če

niso v numerični obliki, pa je zamuden in ga težko vnesemo brez napak.

4.4 Vnos variant s pomočjo programa DEXi z oddaljene lokacije

Za vnos variant z oddaljenih lokacij pa lahko uporabimo kar sam program DEXi na oddaljeni strani. Na oddaljene lokacije naložimo program DEXi in datoteko z odločitvenim modelom. Dijaki vpišejo svoje variante in pošljejo samo tisti del datoteke, ki vsebuje podatke o variantah prek ene od internetnih storitev (FTP, mail) na skupni strežnik. Tu datoteke združimo v skupno datoteko z vsemi variantami in odločitvenim modelom. Po potrebi se izvede tudi združevanje enakih variant. Preverjanje veljavnosti ni potrebno, saj DEXi ustvari veljavno kodo XML.

5. Zaključek

Pri praktični izvedbi medšolskega projektnega dela smo uspešno uporabili oba odločitvena modela. Raziskani so bili načini za vnos podatkov v obliki XML. Prav je, da se dijaki čim prej seznanijo s tako pomembnim standardom. Žal pa je standard XML v osnovi res preprost, ko pa ga želimo praktično uporabiti, se moramo takoj spopasti z dodatnimi standardi, kot so npr. sheme. Zato je realno načrtovati, da bo sheme pripravil učitelj. Seveda bo moral pripraviti tudi vzorčni primer dokumenta XML. Dijaki lahko svoje variante pripravijo v obliki XML ročno ali z enostavnim orodjem, kot je npr. XML Notepad, preverijo slovnično pravilnost in veljavnost dokumenta in ga pošljejo na oddaljeni računalnik. To pa je izvedljivo le, če bi uporabili dovolj enostavne primere, za katere ne bi porabili preveč časa za iskanje napak. S tem bi tudi uporaba sistema za podporo pri odločanju postala bolj namenjena spoznavanju takih sistemov kot pa resnični pomoči pri sprejemanju odločitve. Za kompleksnejše odločitve lahko uporabimo druge načine za vnos podatkov v skupni odločitveni model. Vsaka šola si lahko namesti svoj program DEXi in svoj odločitveni model. Program bo zanesljivo ustvaril veljavne podatke, ki jih bomo lahko združili v skupni odločitveni model.

Literatura

Armstrong, E.,Working with XML, The Java API for Xml Parsing (JAXP) Tutorial, 2000, http://java.sun.com/xml/docs/tutorial.

Breskvar, G.: XML: DTD in sheme, Monitor, marec, 2001.

Gams, M., Bohanec, M., Intelligent systems applications, Informatica 25(3), 2001.

IBM, XML Education - Tutorials, 2001,

http://www-106.ibm.com/developerworks/education/xmlintro/xmlintro.html.

Krapež, A., Rajkovič, V., Wechsterbach, R., Uvajanje tehnologij znanja v predmet informatika v gimnazijah: primer upravljanja z odločitvenim znanjem, Organizacija, oktober, 2000.

MSDN Online, XML Developer Center, 2001, http://msdn.microsoft.com/xml.

W3C Recommendation, maj, 2001,

sheme: http://www.w3.org/TR/xmlschema-1/, DTD: http://www.w3.org/TR/html401/.

Maja Azarov Domajnko je leta 1984 diplomirala na Fakulteti za elektrotehniko v Ljubljani. Zaposlila se je v Iskri v razvojnem oddelku, kjer je delala na področju razvoja modemov in pozneje procesnega vodenja. Od leta 1990 je zaposlena na Srednji šoli za elektrotehniko in računalništvo v Ljubljani, kjer poučuje strokovne predmete. Je soavtorica dveh učbenikov za srednje šole, spletnih učnih pripomočkov in mentorica mladih raziskovalcev na mednarodnih, državnih in regijskih tekmovanjih. V letu 2002 je zaključila magistrski študij na Fakulteti za organizacijske vede z nalogo Večparametrski odločitveni model pri medšolskem projektnem delu pod mentorstvom dr. Vladislava Rajkoviča in dr. Marka Bohanca.

Marko Bohanec je višji znanstveni sodelavec Instituta Jožef Stefan in docent za področje računalniških informacijskih sistemov. Predava na Visoki upravni šoli v Ljubljani in na Fakulteti za organizacijske vede v Kranju. Doktoriral je leta 1991 s področja računalniških znanosti na ljubljanski Fakulteti za računalništvo in informatiko. Njegovo ožje strokovno področje so sistemi za podporo odločanja, ki jih v svojem raziskovalnem, razvojnem in aplikativnem delu povezuje z metodami umetne inteligence, kot so ekspertni sistemi, sistemi strojnega učenja in sistemi za odkrivanje zakonitosti v podatkih.

Vladislav Rajkovič je redni profesor na Fakulteti za organizacijske vede Univerze v Mariboru in sodelavec Odseka za inteligentne sisteme Instituta »Jožef Stefan«. Njegovo področje so računalniški informacijski sistemi, s posebnim poudarkom na sistemih za pomoč pri odločanju. Je soavtor večkriterijske odločitvene metodologije, ki sloni na lupini ekspertnega sistema Dex. Je član Programskega sveta programa »Računalniško opismenjevanje« in vodi temeljni projekt Ekspertni sistemi v izobraževanju. Poleg tega je predstavnik Slovenije v »International Federatino for Infromation Processing« za področje izobraževanja.