Primerjava pristopov k razvoju ontologij

Boštjan Grašič, Vili Podgorelec, Marjan Heričko
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Smetanova ul. 17, 2000 Maribor
bostjan.grasic@uni-mb.si, vili.podgorelec@uni-mb.si, marjan.hericko@uni-mb.si

Izvleček

Vse večjo prisotnost in pomembnost ontologij pri razvoju sodobnih informacijskih rešitev potrjuje tako število raziskovalnih projektov, ki temeljijo na ontologijah, kot vse večje število komercialnih produkcijskih rešitev, ki omogočajo delo z ontologijami in semantičnimi tehnologijami. Kljub precejšni razširjenosti izraza ontologija, ostajajo bistvo in namen v precejšni meri napačno razumljeno. Prvi del prispevka predstavlja ozadje termina ontologija ter izpostavlja pomen ontološkega inženirstva – discipline za razvoj ontologij. V drugem delu predstavimo aktualne pristope k razvoju ontologij in rezultate primerjave različnih pristopov k razvoju ontologij.

Abstract

COMPARISON OF ONTOLOGY DEVELOPEMENT APPROACHES

Ontologies are becoming an important part of contemporary information systems. The importance of the ontologies is proved by the number of research projects dealing with ontologies, as well as by the number of proprietary solutions that are supporting ontologies in production environments. However, the term ontology is still often misunderstood. The first part of this paper describes the essence and the background of ontologies, it compares ontologies with well known terms from ontological engineering and highlights the importance of ontological engineering. The second part of the paper describes the most important approaches to ontology development, and presents the results of the comparative analysis of ontology development approaches.

1 UVOD

Izraz ontologija se je pojavil že v antični filozofiji, ko so z ontologijami skušali izraziti bistvo stvari. Osnovna naloga ontologij je bila odgovoriti na vprašanja, kot so: Kaj so stvari oz. kaj je njihovo bistvo? Ali ostane bistvo stvari nespremenjeno tudi, ko se spreminjajo? Kako lahko klasificiramo stvari/entitete, ki se pojavljajo na svetu? Ali obstajajo koncepti stvari (knjiga, drevo, miza ipd.) tudi zunaj našega razuma ali je to le naš konstrukt? (Gómez-Perez, Corcho, & Fernández-López, 2004)

V sklopu računalništva in informatike so se ontologije začele pojavljati v devetdesetih letih 20. stoletja. Nastopale so v podobni vlogi kot v antiki, vendar je bil njihov cilj namesto človeka računalnik. Osrednje vprašanje je postalo, kako stroju predstaviti bistvo stvari, da bo lahko s procesom sklepanja prišel do novih spoznanj in koristnih informacij. Najprej so se začele ontologije uporabljati na področju umetne inteligence v sklopu predstavitve znanja (angl. knowledge representation).

Namen predstavitve znanja je izraziti znanje o svetu tako, da bodo lahko računalniki obdelovali to znanje in nad njim izvajali proces sklepanja (Davis, Shrobe, & Szolovits, 1993). Osnova podlaga za to je formalna predstavitev znanja, sveta oz. okolja. S procesom sklepanja na podlagi formul, funkcij ali pravil detereministično pridobivamo nove izjave o svetu oz. okolju – novo znanje. Vizija predstavitve znanja je omogočiti strojem, tj. računalnikom, sklepanje o svetu, ki naj bo kar se da podobno sklepanju, ki ga opravljajo človeški možgani (Davis et al., 1993).

Za formalno predstavitev znanja uporabljamo več tehnik. V zametkih področja predstavitve znanja so uporabljali predvsem semantične mreže, pravila in logiko (Davis et al., 1993). Zadnje čase v ospredje prehajajo ontologije, ki temeljijo na različnih formalizmih logike (opisna logika, logika okvirjev ipd.) (Gómez-Perez et al., 2004). Poseben razcvet in prepoznavnost so ontologije doživele z razvojem semantičnega spleta, saj so eden njegovih ključnih elementov.

Izzive razvoja ontologij in zagotavljanja interoperabilnosti naslavlja področje ontološkega inženirstva (angl. ontological engineering), ki skuša razvoj ontologij spremeniti iz spretnosti v inženirstvo. Ontološko inženirstvo po Goméz (Gómez-Perez et al., 2004) je »množica aktivnosti, ki zadevajo razvojni proces

ontologij, življenjski cikel ontologij ter metodologije, orodja in jezike za razvoj ontologij«.

Članek predstavlja in analizira najsodobnejše pristope k razvoju ontologij. Drugi razdelek opisuje področja uporabe ontologij, v tretjem je zgoščeno predstavljeno področje predstavitve znanja, medtem ko četrti podaja definicijo ontologij, jih klasificira in navede jezike za implementacijo. Peti razdelek podaja primerjavo med ontologijami in obstoječimi koncepti iz programskega inženirstva, šesti predstavlja pomembnejše metode in metodologije za razvoj ontologij, v sedmem so podani rezultati primerjave pristopov k razvoju ontologij.

2 VLOGA ONTOLOGIJ V INFORMACIJSKIH SISTEMIH

Ontologije se uporabljajo predvsem v aplikacijah in informacijskih sistemih, pri katerih je poudarek na znanju. Najdemo jih lahko v tako imenovanih sistemih znanja, ekspertnih sistemih, portalih, pri katerih je pomembna informacijska vrednost podatkov in njihova soodvisnost, ter vedno pogosteje tudi pri dinamični integraciji podatkov. Ontologije niso primerne za opravila, pri katerih je v ospredju procesna obdelava podatkov, ali kot nadomestilo podatkovnih skladišč.

Uporabne so v informacijskih sistemih, v katerih je ključnega pomena in vrednosti znanje, ki je zajeto v samem informacijskem sistemu. Prednost ontologij pred klasičnimi programskimi jeziki je, da so namenjene in narejene izključno za predstavitev znanja. Zaradi tega lahko z njimi bolje in bolj preprosto predstavimo znanje, ki je potrebno v informacijskem sistemu. Omogočajo boljši pregled nad znanjem in preprostejše upravljanje ter vzdrževanje znanja v informacijskem sistemu. Pri uporabi klasičnih programskih jezikov je znanje pogosto prepleteno z aplikacijsko logiko. Prepletenost znanja in aplikacijske logike otežuje proces upravljanja z znanjem ter hkrati zmanjšuje organizacijsko agilnost. Namen ontologij ni nadomestitev konvencionalnih programskih jezikov (kot so npr. C++, Java, C#, PHP), temveč njihova dopolnitev. Ontologije služijo kot medij za predstavitev in izmenjavo znanja, konvencionalni programski jeziki pa za implementacijo aplikacijske logike ter za procesiranje podatkov (procesiranje, ki ne temelji na procesu sklepanja).

Informacijski sistemi, ki temeljijo na ontologijah, že prehajajo v produkcijska okolja. V nadaljevanju na kratko povzemamo dva primera uporabe ontologij v uspešnih in priznanih evropskih podjetjih. V priznanem proizvajalcu avtomobilov so vpeljali inteligentni sistem, ki pomaga pri testiranju elektronike za nove avtomobile. Elektronski sestavi v modernih avtomobilih so zelo kompleksni in prepleteni med seboj. Pregled nad celotnim delovanjem takšnega elektronskega sestava je skoraj nemogoč. Ker je v sodobnih vozilih elektronski sestav tako rekoč osrednja točka avtomobila, je zanesljivo delovanje avtomobila odvisno od zanesljivosti elektronskega sestava. Z uporabo ontologij so uspeli eksplicitno definirati predvideno in zahtevano obnašanje komponent ter njihovo povezavo z drugimi komponentami. Celoten sistem znatno olajša proces testiranja ter izboljša predvidljivost obnašanja elektronskih sestavov (Syldatke, Chen, Angele, Nierlich, & Ullrich, 2007).

Naslednji primer uporabe ontologij v realnih okoljih najdemo pri priznanem proizvajalcu industrijskih robotov. S pomočjo ontologij so modelirali obnašanje in napake robotov ter njihovih sestavnih delov. Sistem je namenjen identifikaciji in odpravi mehanskih ter programskih okvar industrijskih robotov. Sistem služi za zajem in izmenjavo ekspertnega znanja o okvarah robotov (Nierlich 2007).

3 PREDSTAVITEV ZNANJA

Glavni namen, s katerim se ukvarja področje predstavitve znanja, je snovanje računalniških sistemov, ki so sposobni sklepati po strojno berljivi predstavitvi sveta, in sicer čim bolj podobno človeškemu sklepanju. Davis opisuje predstavitev znanja kot nadomestek realnosti, znotraj katerega opišemo stvari oz. podamo izjave o svetu in omogočimo strojno sklepanje o svetu (Grimm, Hitzler, & Abecker, 2007). Tehnologije, ki to omogočajo in se pogosto uporabljajo za namene predstavitve znanja, so semantične mreže, pravila in logika.

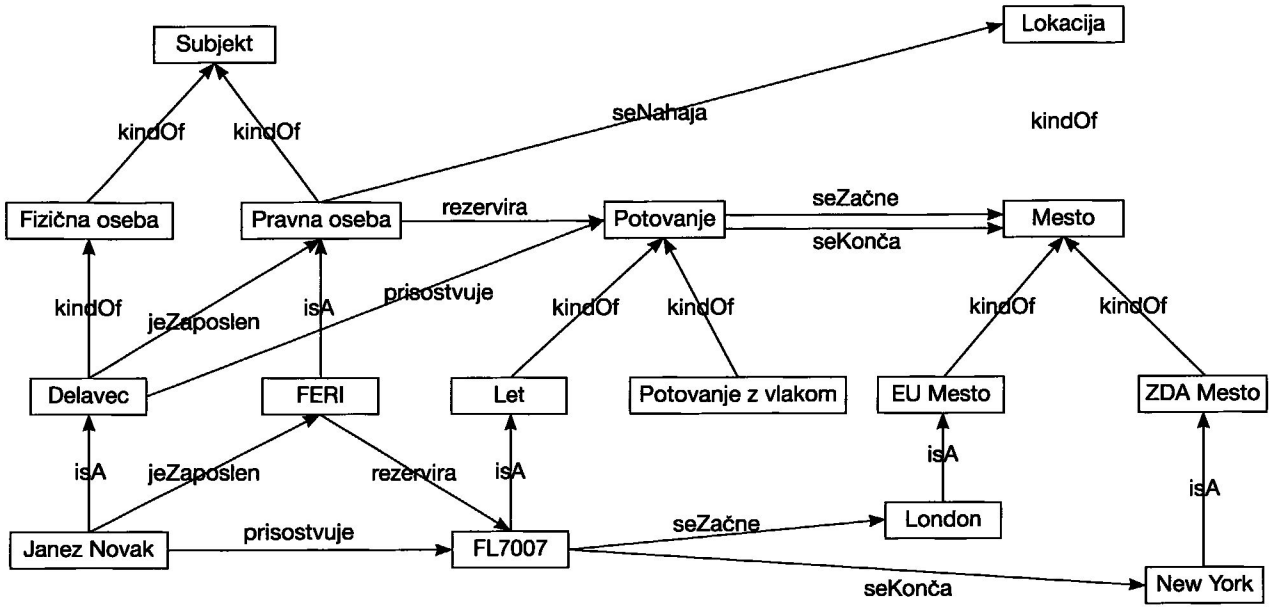
3.1 Semantične mreže

Semantične mreže se osredinjajo na izražanje taksonomskih struktur kategorij objektov in relacij med njimi. Semantična mreža je usmerjen graf, katerega vozlišča predstavljajo koncepte, povezave pa relacije med njimi. Omogočajo strukturno predstavitev izjav znotraj neke domene. Semantična mreža na sliki 1 predstavlja znanje iz domene poslovnih potovanj, ki je zajeto v naslednjem prostem besedilu (Grimm et al., 2007):

Primer 1: **Znanje, predstavljeno s prostim besedilom**

Delavci, ki so zaposleni pri pravnih osebah, so fizične osebe. Pravne in fizične osebe so subjekti. Pravne osebe rezervirajo potovanja za svoje zaposlene. Potovanja so lahko leti ali potovanja z vlakom. Vsako potovanje se začne ali konča v mestu, ki je bodisi v EU ali ZDA. Pravne osebe se nahajajo na poljubni lokaciji.

Pravna oseba FERI je rezervirala let FL7007 iz Londona v New York za Janeza Novaka.



Slika 1: **Primer semantične mreže, ki vsebuje znanje iz primera 1**

3.2 Pravila

Druga možnost strukturirane predstavitve znanja so pravila. Pravila izražajo vzročnost in imajo obliko konstruktov IF-THEN. Pravila, predstavljena v primeru 2, so zapisana v neformalni obliki in jih ni mogoče procesirati strojno. Za formalno predstavitev pravil uporabljamo logiko, sama konverzija je precej preprosta. Za zgornji primer bi bila predstavitev znanja v obliki pravil takale:

Primer 2: **Predstavitev znanja iz primera 1 z uporabo pravil**

1. IF nekaj je let THEN je tudi potovanje.
2. IF neka fizična oseba prisostvuje na potovanju, ki ga je rezervirala neka pravna oseba THEN; ta fizična oseba je zaposlena pri tej pravni osebi.
3. FACT[1] oseba JanezNovak prisostvuje na letu, ki ga je rezervirala pravna oseba FERI.
4. IF začetna in končna lokacija potovanja sta blizu THEN potovanje je z vlakom.

Semantične mreže so uporabne za namene taksonomizacije konceptov in relacij med njimi. Njihova šibkost je pri izražanju natančnejše specifikacije konceptov in relacij med njimi. S pravili je mogoče predstaviti znanje iz semantičnih mrež. Obratna relacija ni mogoča. Tako recimo semantične mreže ne omogočajo predstavitve znanja, ki je izraženo z zadnjim pravilom iz primera 2.

[1] FACT označuje dejstvo, to je znanje, ki je znano že pred izvajanjem pravil (eksplicitno znanje).

3.3 Logika

Semantične mreže in pravila lahko formalno predstavimo z uporabo logike. Brez natančne formalizacije je predstavitev znanja *neopredeljiva* in *dvoumna* kar onemogoča *strojno procesiranje*. Logika je sredstvo, s katerim lahko predstavimo tako semantične mreže kot pravila na formalen način in s tem omogočimo strojno procesiranje. Ni nujno, da se pri predstavitvi znanja omejimo samo na konstrukte semantičnih mrež ali pravil, uporabimo oz. definiramo lahko poljubne logične izraze. Najpogostejše vrste logike, ki se uporabljajo za predstavitev znanja, so logika prvega reda, opisna logika, predikatna logika, F-logika in Hornova pravila (Grimm et al., 2007).

4 ONTOLOGIJE

Ontologije so eden od načinov predstavitve znanja. Za formalen zapis pogosto uporabljamo opisno logiko, F-logika, opisno logiko v kombinaciji s Hornovimi pravili ali logično programiranje. V akademskem svetu še ni bil dosežen konsenz o splošni definiciji ontologij. Najbolj sprejeta in tudi citirana definicija je Gruberjeva iz leta 1993: »Ontologija je eksplicitna specifikacija konceptualizacije.«[2] (Gruber, 1993)

Na definicijo je bilo podanih precej komentarjev, predvsem glede izrazov specifikacija in konceptualizacija, saj naj ne bi bila dovolj pomensko polna. Borst je leta 1997 dopolnil Gruberjevo definicijo ontologije: »Ontologije so definirane kot formalna specifikacija skupne konceptualizacije.«[3] (Borst, 1997)

Obe definiciji so združili in razložili Studer idr. (Studer, Benjamins, & Fensel, 1998): »Ontologija je formalna in eksplicitna specifikacija skupne konceptualizacije«, pri čemer posamezne termine interpretirajo takole:

- **konceptualizacija** se nanaša na abstraktni model nekega pojava v svetu na podlagi identificiranih konceptov tega pojava;
- **eksplicitna** pomeni, da so tipi uporabljenih konceptov in omejitve pri njihovi uporabi eksplicitno definirani (omejitev je npr., da se koncepta Let in PotovanjeZVlakom izključujeta);
- **formalna** se nanaša na dejstvo, naj bi bila ontologija strojno berljiva, kar izključuje naravni jezik;
- **skupna** odraža stališče, naj ontologija zajema splošno oz. skupno znanje, to je znanje, ki ni lastno posamezniku, temveč ga je sprejela določena skupina ljudi.

Ontologija ni samo računalniška predstavitev znanja neke domene, temveč izraža neko stopnjo konsenza o znanju iz te domene (Studer et al.., 1998). Gomez idr. poudarjajo, da neki domenski model ni nujno ontologija zaradi tega, ker je predstavljen v ontološkem jeziku (npr. Ontolingua ali Web Ontology Language (OWL)), iz enakih razlogov, kot ni neki program sistem znanja samo zaradi tega, ker je napisan v prologu (Gómez-Perez et al., 2004).

Za naš primer iz domene poslovnih potovanj lahko rečemo, da je ta model ontologija takrat, ko ga bomo formalno zapisali in ko bo med uporabniki tega modela dosežen konsenz o znanju, ki je zajeto v ontologiji. Studer (Studer et al., 1998) navaja, da naj bo konsenz odvisen od konteksta; če npr. razvijamo ontologijo o boleznih za bolnišnico, mora biti dosežen konsenz med vsemi zdravniki te bolnišnice, če pa razvijamo nacionalni bibliografski sistem, naj bo dosežen konsenz na nacionalni ravni – vsak uporabnik naj bi sprejel ontologijo kot veljavno.

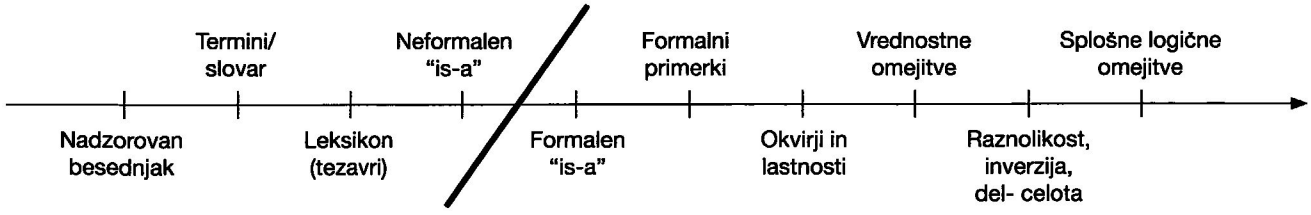
4.1 Vrste ontologij

Lassila in McGuiness sta se zaradi različne, pogosto tudi napačne uporabe izraza ontologija odločila kategorizirati vse vrste njegovih pojavitev (Lasilla & McGuinness, 2001). Ontologije sta klasificirala kot linearen spekter glede na bogatost in izraznost njihove interne strukture. Slika 2 prikazuje, da bolj kot se pomikamo po premici v desno, bolj bogata je ontologija. Točke na premici predstavljajo pogoste specifikacije ontologij, na katere so naleteli avtorji. Prečna črta razmejuje umestnost uporabe izraza ontologija.

Najbolj osnovna oblika, ki so jo avtorji zasledili pri uporabi izraza ontologija, je nadzorovano besedišče, ki predstavlja samo končni niz izrazov. Pomensko polnejši je slovar, ki predstavlja seznam terminov in njihovih pomenov. Takšna predstavitev daje več informacij, ki pa pogostno niso enoumne (en termin lahko označuje dve pomensko različni stvari).

Leksikoni oz. tezavri predstavljajo dodatno semantiko v obliki relacij med termini. Tezavri ponujajo informacije o sopomenkah in niso tipično organizirani v hierarhijo.

[2] An ontology is an explicit specification of a conceptualization.
[3] Ontologies are defined as a formal specification of a shared conceptualisation.



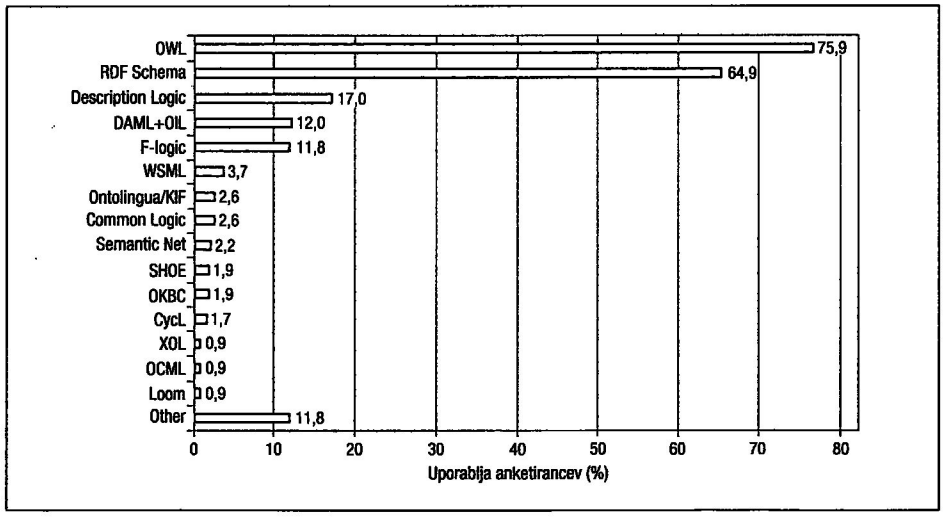
Slika 2: **Kategorizacija ontologij po Lasilla in McGuiness**

Neformalna hierarhija »is-a« predstavlja neformalno generalizacijo, kar pomeni, da ni nujno, da je pripadnik specifičnega razreda hkrati tudi pripadnik generičnega (nadrejenega) razreda. Formalne hierarhije »is-a« vsebujejo striktne relacije »podrazred«, v katerih mora biti pripadnik specializiranega koncepta obvezno tudi pripadnik generalnega koncepta.

Formalni primerki nadgrajujejo striktne hierarhije »is-a« tako, da dosledno ločijo med koncepti in njihovimi primerki (razredi in objekti). Okvirji in lastnosti gredo korak naprej s specifikacijo lastnosti, ki jih imajo lahko koncepti in objekti. Kompleksnejše vrste ontologij še natančneje definirajo koncepte, primerke in njihove soodvisnosti (v obliki logičnih omejitev, relacij ipd.).

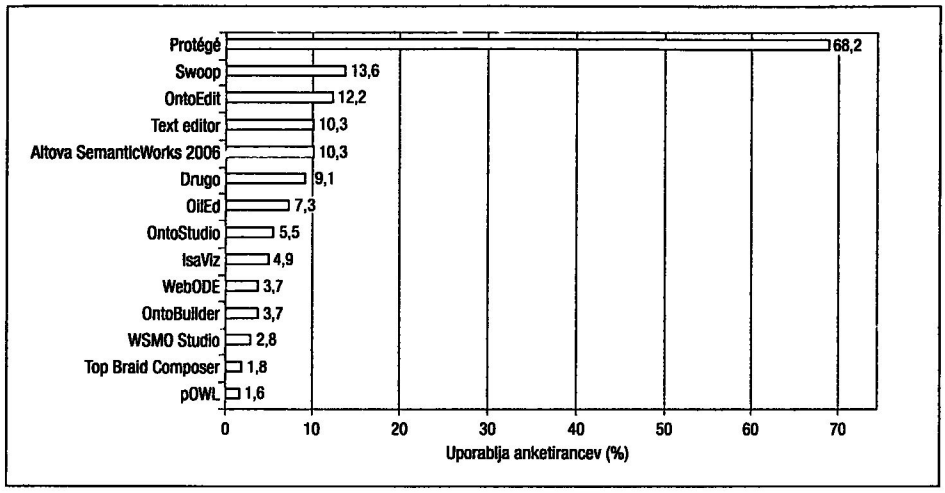
4.2 Jeziki za implementacijo ontologij

Jezike za implementacijo ontologij lahko delimo na tradicionalne in označevalne. Večina tradicionalnih in začetnih označevalnih jezikov ni pogosto uporabljanih. Najsodobnejši in trenutno najbolj razširjeni ontološki jeziki so Resource Description Framework (RDF) (Manola & Miller, 2004) in RDF Schema (RDFS), OWL (McGuinness, 2004) ter Web Service Modeling Language (WSML). Cordoso je leta 2007 izvedel anketo o uporabljenosti ontoloških jezikov in orodij med 700 raziskovalci ter razvijalci na področju semantičnega spleta. Rezultati raziskave so prikazani na slikah 3 in 4; slika 3 prikazuje razširjenost uporabe posameznih ontoloških jezikov, slika 4 pa razširjenost uporabe orodij za razvoj ontologij. Podrobnejša analiza je na voljo v Gómez-Perez et al. (2004).



Slika 3: **Uporaba jezikov za razvoj ontologij (Cardoso, 2007)**[4]

[4] OWL je naslednik jezika DAML+OIL; Description Logic ni jezik, temveč je vrsta logike, ki jo med drugimi uporablja tudi OWL.



Slika 4: **Uporaba orodij za razvoj ontologij (Cardoso, 2007)**

5 ONTOLOGIJE IN KLASIČNI KONCEPTI PROGRAMSKEGA INŽENIRSTVA

V nadaljevanju bomo podali primerjavo med ontologijami in poznanimi koncepti s področja programskega inženirstva (metamodel, podatkovni model, objektni model).

5.1 Ontologije in metamodeli

Model je abstrakcija stvarnosti, metamodel predstavlja dodatno abstrakcijo, ki označuje lastnosti modela samega (Söderström, Andersson, Johannesson, Perjons, & Wangler, 2002). Medtem ko nekateri avtorji (Söderström et al., 2002) primerjajo metamodele z ontologijami, jih drugi uvrščajo med ontologije (OMG, 2005). Po našem mnenju je primernejša umestitev ontologij med modele, ontološke jezike pa lahko smatramo kot ontološke metamodele. Izhajamo iz osnovne definicije predstavitve znanja, ki pravi, da je predstavitev znanja nadomestek realnosti (torej model), ki omogoča izvajanje procesa sklepanja nad dejstvi predstavljenimi znotraj njega (Davis et al., 1993). Ontologije kot tehnika oz. metoda predstavitve znanja nastopajo v enaki vlogi.

Ker so ontologije zelo fleksibilne, lahko nekatere nastopajo tudi v vlogi metamodelov (npr. ontologije za izdelavo ontologij). Object Management Group (OMG) je izdelal metamodele najbolj razširjenih ontoloških jezikov (OWL, RDF, RDF-Schema) (OMG, 2005) ter jih povezal z drugimi metamodeli (UML, objektni metamodel, entitetnorelacijski metamodel).

5.2 Ontologije in podatkovni modeli

Podatkovni modeli (npr. entitetnorelacijski model) vsebujejo z vidika razvijalca samo za namen aplikacije omejen pogled na svet. Ko se zahteve aplikacije spremenijo, se spremenita tudi vidik in podatkovni model. V splošnem morajo imeti ontologije lastnost deljenosti (glej definicijo v poglavju 4) – biti morajo splošne in vsebovati več pogledov na realnost. Zaradi tega ob spremembi zahtev ni treba spreminjati ontologije, če se svet oz. stvarnost, ki jo opisuje, ni spremenila (Tran, Lewen, Haase, 2007).

Pomembna razlika med ontologijami in podatkovnimi modeli je tudi ta, da ontologije omogočajo proces sklepanja in pridobivanja novih dejstev, medtem ko so podatkovni modeli statični in vsebujejo zgolj podatke, ki smo jih vstavili eksplicitno (Tran et al., 2007). Namen ontologij ni nadomestiti podatkovne modele oz. baze. Nasprotno, namen ontologij je dopolniti njihovo funkcionalnost. Zadnje čase je vse bolj razširjena preslikava med podatkovnimi modeli in ontologijami (npr. Bizer & Cyganiak (2006)), ki omogoča dostop do relacijskih podatkov s semantičnimi tehnologijami.

5.3 Ontologije in objektni modeli

Ontologije in objektni modeli imajo podobne koncepte in mehanizme. Objektni modeli imajo razrede, katerih ekvivalent pri ontologijah so koncepti. Ekvivalent atributom objektov so lastnosti konceptov. Pomembna razlika med ontologijami in objektnimi

modeli je, da objektni modeli definirajo obnašanje objekta, medtem ko ontologije sploh ne obravnavajo obnašanja razredov. Oba pristopa vsebujeta dedovanje, vendar je v primeru objektnih modelov bolj napredno (polimorfizem, dinamično povezovanje). V primeru objektnih modelov se pri dedovanju podedujejo vse lastnosti razreda vključno z obnašanjem. Pri dedovanju v okviru ontologij koncepti podedujejo samo lastnosti starševskega koncepta (brez logičnih omejitev in dodatnih specifikacij) (Tran et al., 2007).

Sklenemo lahko, da oba pristopa predstavljata neko domeno oz. del stvarnosti, vendar z drugačnim namenom. Objektni model ima bolj napredne koncepte za specifikacijo obnašanja, kar omogoča procesiranje podatkov, ki jih zahteva aplikacija. Ontologije se osredinjajo na modeliranje splošnega znanja o svetu (stvarnosti). Ontološki jeziki (npr. OWL) ponujajo bolj izrazne konstrukte za doseganje tega cilja (npr. operacije množic za kompleksne izraze nad razredi, karakteristike lastnosti, več vrst omejitev ipd.) (Tran et al., 2007).

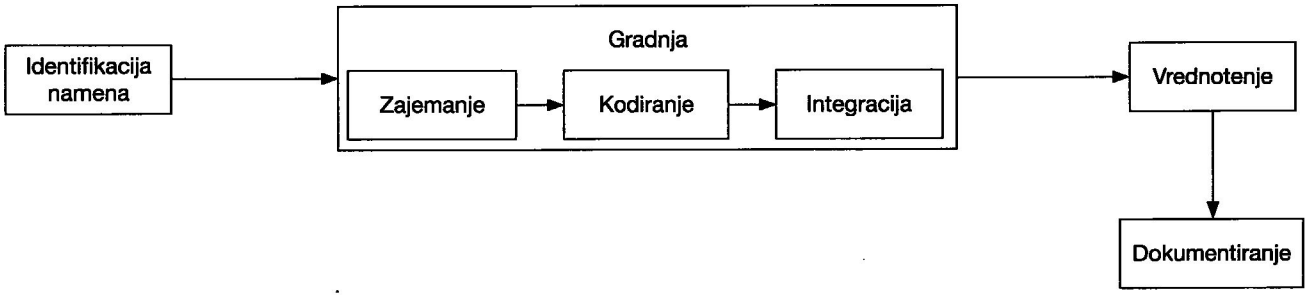
Ontologije in objektni modeli imajo različne – komplementarne namene. Kadar je potrebno napredno procesiranje podatkov, so primernejši objektni modeli, medtem ko so ontologije primernejše za bolj izrazno modeliranje znanja (Tran et al., 2007).

6 PRISTOPI K RAZVOJU ONTOLOGIJ

V nadaljevanju opisujemo pristope za razvoj ontologij kronološko glede na njihov nastanek. Pri vsakem pristopu bomo opisali glavne korake ter poudarili novosti, ki jih je uvedel posamezen pristop. Zaradi prevelike razlike v ravneh specifikacije pristopov bomo pri vsakem opisali samo korake, procese, faze ali opravila.

6.1 Uschold-Kingova metoda

Ushold in King sta svoj pristop označila kot metodo, kljub temu jo več virov navaja kot metodologijo (Cardoso (2007), Corcho, Fernández-López, Gómez-Pérez (2003), Gómez-Perez et al. (2004)). Pristop predstavlja prvo metodologijo za razvoj ontologij. Slika 5 prikazuje procese metodologije.



Slika 5: **Procesi Uschold-Kingove metode razvoja ontologij**

Proces identifikacije namena

Glavni cilj procesa je razjasniti, zakaj razvijamo ontologijo, kakšna je predvidena uporaba ontologije in kateri so glavni pojmi domene ontologije. V okviru poslovnih potovanj bi to bili pojmi **Potovanje, Lokacija, Pravna oseba, Fizična oseba**.

Proces gradnje

Proces gradnje je sestavljen iz treh aktivnosti.

1. Zajemanje ontologije – aktivnost vsebuje naslednja opravila: (i) identifikacija glavnih konceptov in relacij domene, (ii) izdelava natančnih in enoumnih tekstovnih definicij konceptov in relacij, (iii) identifikacija pogojev, ki se nanašajo na identificirane koncepte in relacije, in (iv) doseganje sporazuma o rezultatih prejšnjih opravil.

Za identifikacijo konceptov v ontologiji avtorji predlagajo tri pristope (Gómez-Perez et al., 2004):

- od spodaj navzgor – identifikacija najbolj specifičnih elementov (npr. Let, EUMesto), ki jih nato generaliziramo v bolj abstraktne koncepte,
- od zgoraj navzdol – identifikacija najbolj abstraktnih konceptov (npr. Subjekt, Lokacija), ki jih nato specializiramo v bolj specifične koncepte,
- iz sredine navzven – identifikacija jedra osnovnih terminov (npr. PravnaOseba, Let, Lokacija), ki jih nato po potrebi specializiramo in generaliziramo. Ta pristop prestavlja ravnovesje med prej omenjenima pristopoma in je najpogosteje uporabljeni pristop.

2. Kodiranje – dejanska implementacija ontologije v ontološkem jeziku. Metodologija ne predvideva ali predlaga uporabe specifičnega jezika,
3. Integracija – integracija ontologije z drugimi ontologijami.

Proces vrednotenja

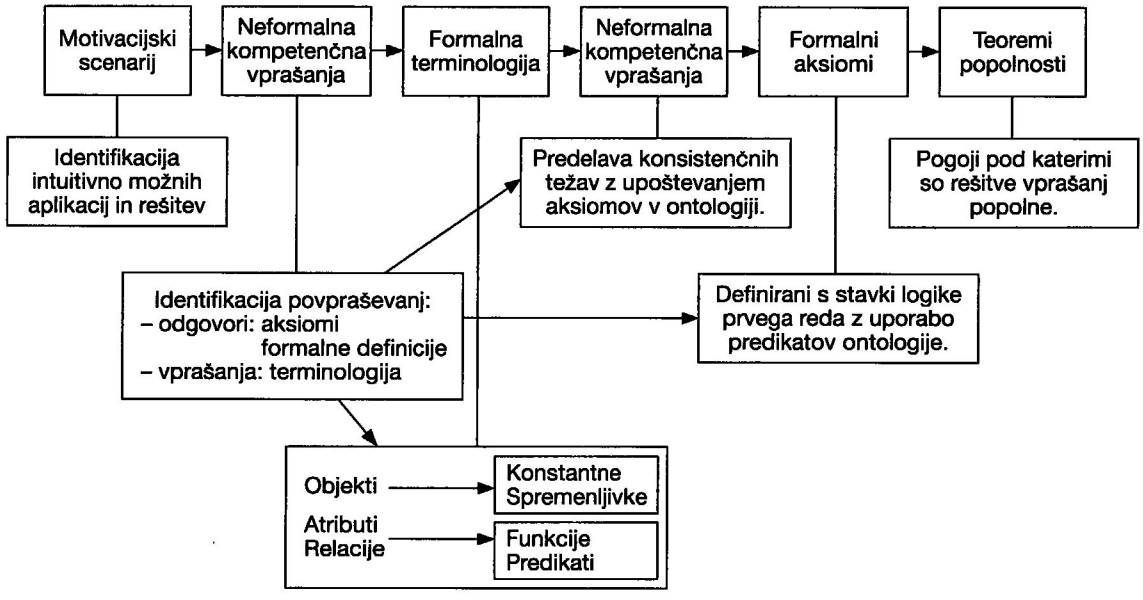
Vrednotenje ontologij se osredinja na ocenjevanje ontologij. Če želimo uporabiti neko ontologijo – bodisi lastno ali od drugih avtorjev –, je koristno preveriti kakovost vsebine, torej definicij konceptov, taksonomije in aksiomov. Metodologija ne specificira koraka podrobno, temveč ga samo omenja.

Proces dokumentiranja

Proces podaja priporočila za dokumentiranje ontologij, ki so različna glede na namen ontologije. Tudi tega procesa metodologija ne specificira podrobno, temveč ga samo navaja.

6.2 Grüninger-Foxova metodologija

Grüninger-Foxova metodologija izhaja iz izkušenj, pridobljenih pri razvoju sistemov znanja, ki uporabljajo logiko prvega reda. Razvijalcem ontologij predlagajo, da začnejo z intuitivno identifikacijo glavnih scenarijev, ki so mogoči za uporabo ontologije. Uporabljajo t. i. kompetenčna vprašanja (KV)[5], ki določajo obseg ontologije. Na podlagi kompetenčnih vprašanj in odgovorov nanja poteka ekstrakcija glavnih konceptov, njihovih lastnosti, relacij in formalnih aksiomov ontologije. Kompetenčna vprašanja služijo kot smernice pri razvoju ontologije. Procesi metodologije so prikazani na sliki 6.



Slika 6: **Procesi Grüninger-Foxovove metodologije**

Proces identifikacije motivacijskih scenarijev

Razvoj ontologij poteka na podlagi scenarijev, ki so povezani z načinom uporabe ontologij v aplikacijah. Ti scenariji vsebujejo množico ontoloških zahtev, ki morajo biti izpolnjene po formalni implementaciji. Motivacijski scenariji vključujejo množico intuitivno možnih rešitev k problemom, ki jih naslavljajo scenariji. Te rešitve neformalno nakazujejo semantiko objektov in relacij, ki bodo kasneje zajeti v ontologiji.

Proces definicije neformalnih kompetenčnih vprašanj

Na podlagi motivacijskih scenarijev poteka definicija neformalnih kompetenčnih vprašanj. Neformalna kompetenčna vprašanja so izražena v naravnem jeziku in predstavljajo vprašanja, na katera mora ontologija ponuditi odgovore, ko bo izražena v formalnem jeziku. Kompetenčna vprašanja imajo vlogo specifikacije zahtev, na podlagi katerih lahko kasneje vrednotimo ontologijo. Uporabna so za pridobivanje predpostavk, omejitev, potrebnih vhodnih podatkov ipd.

[5] Tehniko kompetenčnih vprašanj povzemajo tudi sodobne metodologije.

Proces specifikacije formalne terminologije

Razvijalec ontologij lahko uporabi neformalna kompetenčna vprašanja za ekstrakcijo vsebine ontologije; identificira koncepte, atribute, relacije in jih predstavi v logiko prvega reda. Rezultat procesa so formalno definirani koncepti, njihovi atributi in relacije ter taksonomija konceptov.

Proces oblikovanja formalnih kompetenčnih vprašanj

Metodologija predvideva pretvorbo neformalnih kompetenčnih vprašanj v formalna. Za formalno predstavitev se uporabi logika prvega reda in koncepti ter relacije, definirane v prejšnjem procesu. Tako definirana formalna kompetenčna vprašanja se lahko neposredno uporabijo pri vrednotenju implementirane ontologije v obliki testov.

Proces specifikacije formalnih aksiomov

V tem procesu definiramo aksiome, ki specificirajo definicijo terminov v ontologiji in omejitve njihove interpretacije. Metodologija predpisuje uporabo logike prvega reda, tipični aksiomi so aksiomi generalizacije (je_podrazred), lahko pa podamo kakršne koli druge omejitve (npr. lete lahko rezervira podjetje samo za svoje zaposlene).

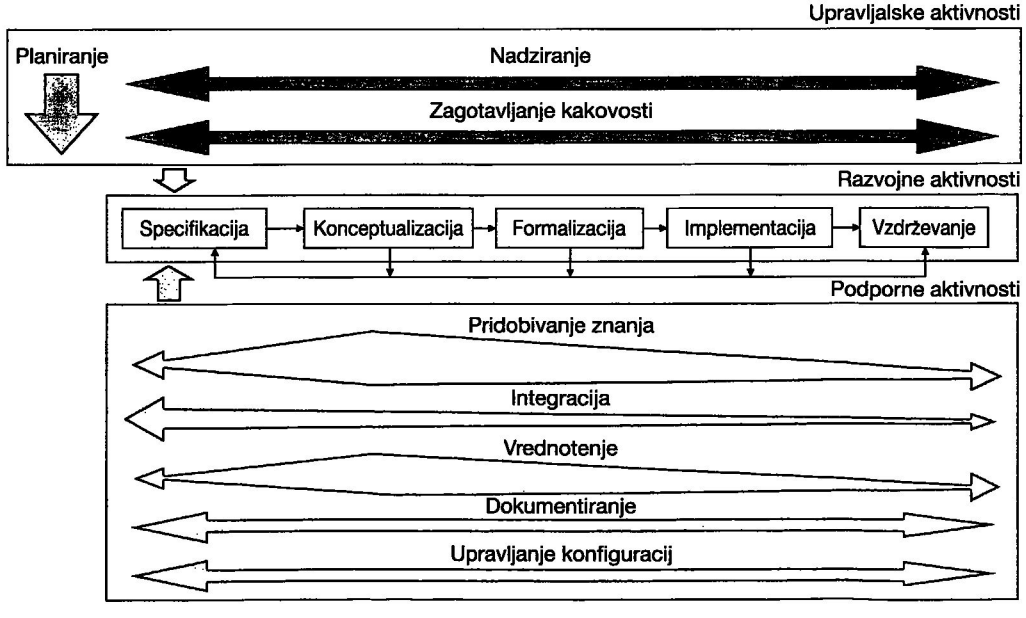
Proces specifikacije teoremov popolnosti

Ko so bila kompetenčna vprašanja formalno definirana, je treba definirati pogoje, pod katerimi so te rešitve popolne. Te definicije služijo kot osnova za specifikacijo teoremov popolnosti.

6.3 Metodologija methontology

Metodologija methontology izvira iz standarda IEEE za razvoj procesov življenjskih ciklov programske opreme. Standard opisuje aktivnosti, ki jih je koristno izvajati pri razvoju življenjskih ciklov programske opreme (osnovne aktivnosti standarda so zajete v tabeli 1 v poglavju 7) (IEEE, 2006). Metodologija definira aktivnosti razvoja ontologij na podlagi navedene specifikacije. Metodologija methontology ne podaja strogega vrstnega izvajanja aktivnosti, vendar jih namesto tega združuje v skupine in dopušča njihovo vzporedno izvajanje.

Aktivnosti so združene v tri skupine: (i) upravljalske aktivnosti, (ii) razvojne aktivnosti in (iii) podporne aktivnosti. Upravljalske in podporne aktivnosti tečejo vzporedno z razvojnimi aktivnostmi – intraodvisnost. Življenjski cikel razvoja ontologij po tej metodologiji temelji na principu razvijajočih prototipov. To pomeni, da vsaka nova verzija dodaja ali odvzema termine oz. koncepte. Metodologija dopušča izvajanje aktivnosti razvoja ontologije sočasno z aktivnostmi razvoja druge ontologije, od katere je odvisen razvoj prve ontologije, t. i. interodvisnost. Avtorji kot primer ponujajo prepleten razvoj ontologij, pri čemer lahko ena skupina razvija ontologijo za neko domensko področje (npr. letalski poleti), ki jo



Slika 7: **Procesi in aktivnosti metodologije methontology**

uporabijo v ontologiji druge domene (npr. poslovna potovanja).

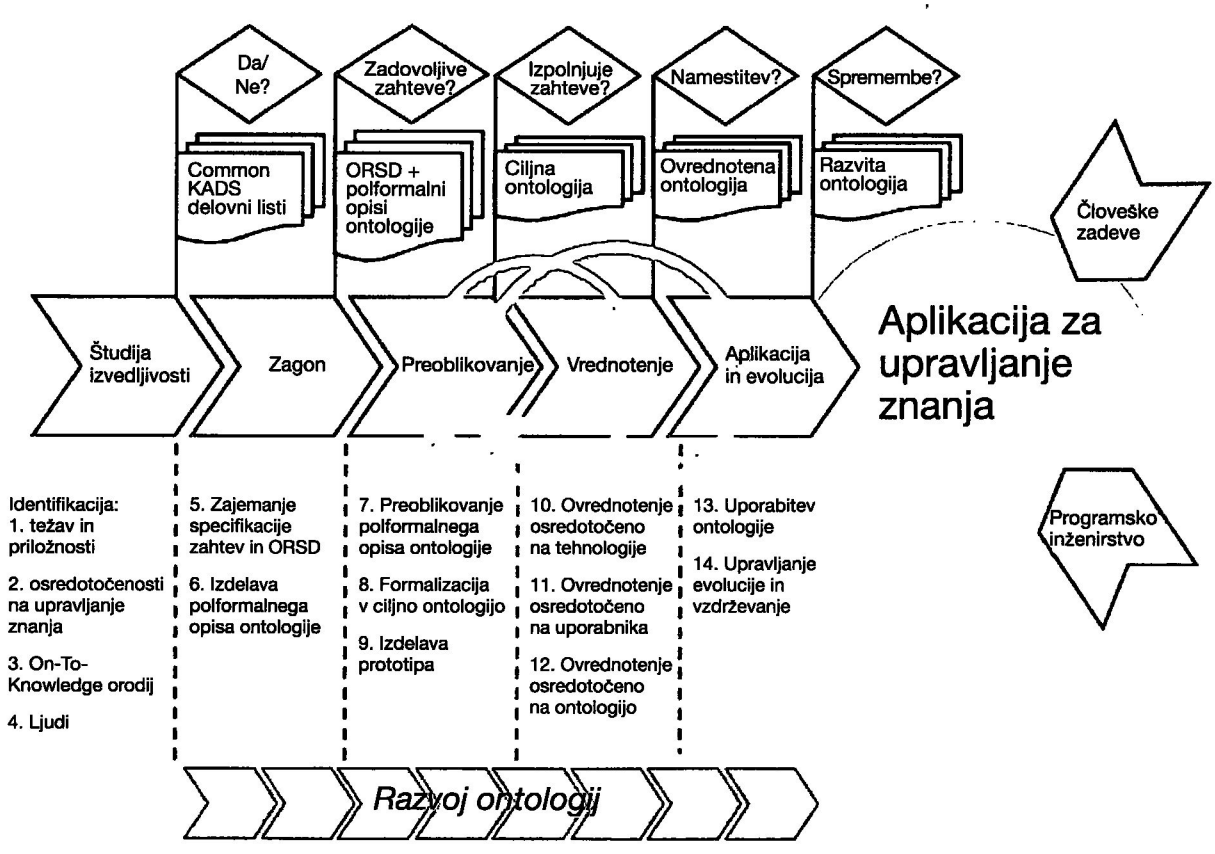
Proces razvoja ontologij in njegove aktivnosti so prikazane na sliki 7. Razvojni proces se začne s planiranjem – naredimo načrt izvajanja aktivnosti. Upravljalske aktivnosti vsebujeta dve aktivnosti, ki tečeta vzporedno z razvojem ontologije, to sta (i) nadziranje izvajanja načrtovanih aktivnosti ter (ii) preverjanje kakovosti produktov, ki so rezultat posamezne aktivnosti.

Razvojni proces predvideva linearno izvajanje aktivnosti, ki se ponavljajo z vsako iteracijo. Ni nujno, da se v vsaki iteraciji izvedejo vse aktivnosti, lahko se izvedejo samo tiste, ki so pomembne za posamezno iteracijo. Aktivnost specifikacije odgovarja na vprašanja, zakaj razvijamo ontologijo, kdo so njeni uporabniki, kakšna je predvidena uporaba ontologije (uporabijo se lahko kompetenčna vprašanja). Posebnost metodologije methontology je proces konceptualizacije.

Metodologija ne priporoča neposredne predstavitve v konkretnem formalizmu ali ontološkem jeziku, temveč vnaša t. i. vmesno predstavitev v obliki konceptualnega modela, ki naj bi olajšal težave pri napakah v konceptualni zasnovi in olajšal spremembe. Določene nejasnosti, ki se pojavljajo v konceptualnem modelu, lahko odpravimo z opcijsko aktivnostjo formalizacije. Tako nastali formalni konceptualni model lahko v procesu implementacije preslikamo v konkretno ontologijo, ki je zapisana v enem od ontoloških jezikov.

6.4 Metodologija On-To-Knowledge

Metodologija On-To-Knowledge (OTKM) je nastala v okviru projekta On-To-Knowledge,[6] katerega cilj je bil aplikacija ontologij k elektronsko dostopnim informacijam za izboljšanje kakovosti upravljanja znanja v velikih in distribuiranih organizacijah (Sure, Staab, & Studer, 2004). Metodologija je sestavljena iz petih glavnih korakov, ki so prikazani na sliki 8 in podrobneje opisani v nadaljevanju.



Slika 8: **Procesi in aktivnosti metodologije On-To-Knowledge (prirejeno po Sure et al. (2004))**

[6] www.ontoknowledge.org

Faza 1: Študija izvedljivosti

Metodologija On-To-Knowledge povzema študijo izvedljivosti po metodologiji CommonKADS, ki je bila narejena za sisteme za upravljanje z znanjem in se osredinja na celotno aplikacijo in ne samo na ontologijo. Služi kot osnova za proces zagona.

Faza 2: Zagon

V fazi zagona se začne dejanski razvoj ontologije. Razvoj se začne z dokumentom specifikacije zahtev ontologije (ontology requirements specification document – ORSD). ORSD opisuje, kaj naj podpira ontologija, skicira predvideno aplikacijo ontologije, smernice razvoja, vire znanja in potencialne uporabnike. Pri specifikaciji lahko uporabimo tudi kompetenčna vprašanja. ORSD naj služi kot vodnik razvijalcu ontologije, kot pripomoček za identifikacijo konceptov, relacij ter hierarhične strukture.

Ta faza predvideva izdelavo polformalne ontologije v sodelovanju z domenskimi eksperti. Ko so zahteve zadovoljivo specificirane, metodologija dopušča prehod v naslednjo fazo. Zadovoljivo se smatra takrat, kadar ni več potrebe po dodatnem pridobivanju ali analiziranju znanja. Seveda lahko zmeraj v kakšni kasnejši fazi nastopi potreba po dodatnem pridobivanju znanja. Za takšne primere metodologija omogoča povratne zanke.

Faza 3: Preoblikovanje

V fazi preoblikovanja formaliziramo polformalno ontologijo. Metodologija predvideva izgradnjo taksonomije. V kombinaciji s polformalno ontologijo lahko uporabimo tudi pristop generiranja ontologije, ki pospeši gradnjo. Ko je definirana taksonomija, metodologija predlaga dodajanje relacij k taksonomiji. V tej fazi se pogosto pojavi pojavijo odprta vprašanja, zaradi tega metodologija predvideva konzultacije z domenskimi eksperti.

Rezultat te faze je ciljna ontologija. Ključno vprašanje, ki se pojavlja v tej fazi je, ali ciljna ontologija izpolnjuje zahteve, ki so bile zajete v fazi zagona. Prva verzija ontologije služi za razvoj prototipa aplikacije, ki jo uporabimo za vrednotenje.

Faza 4: Vrednotenje

Faza vrednotenja se osredinja na vrednotenje aplikacije s treh vidikov: (i) tehnološkega, (ii) uporabniškega in (iii) z vidika ontologije. Ovrednotenje s tehnološkega vidika vsebuje vrednotenje lastnosti generiranih ontologij (sintaksa, semantika) in vrednotenje tehnoloških lastnosti (npr. interoperabilnost, performanse, skalabilnost). Z uporabniškega vidika vrednotimo zadovoljstvo uporabnikov z aplikacijo za upravljanje z znanjem, medtem ko za vrednotenje z vidika ontologij uporabimo pristope vrednotenja ontologij.

Izhod iz te faze je ovrednotena ontologija. Glavno vprašanje ob koncu te faze je, ali razvita ontologija izpolnjuje vse kriterije, ki so relevantni za aplikacijo ontologije. Če je odgovor pritrdilen, lahko začnemo z namestitvijo ontologije v produkcijski sistem. Pogosto je treba opraviti več ciklov preoblikovanja in vrednotenja, da bi dosegli načrtovano stopnjo.

Faza 5: Aplikacija in evolucija

Zadnja faza vsebuje aplikacijo ontologije v produkcijske sisteme. Evolucija v okviru OTKM je definirana kot organizacijski proces, ki vsebuje striktna pravila za procese posodabljanja, vstavljanja in brisanja ontologij. V okviru te faze je treba določiti, kdo je odgovoren za vzdrževanje, kako se izvaja ter v kakšnih časovnih intervalih.

Glavno odločitev te faze predstavlja vprašanje, kdaj začeti naslednji evolucijski cikel ontologije. Evolucijski cikel predstavlja novo različico ontologije.

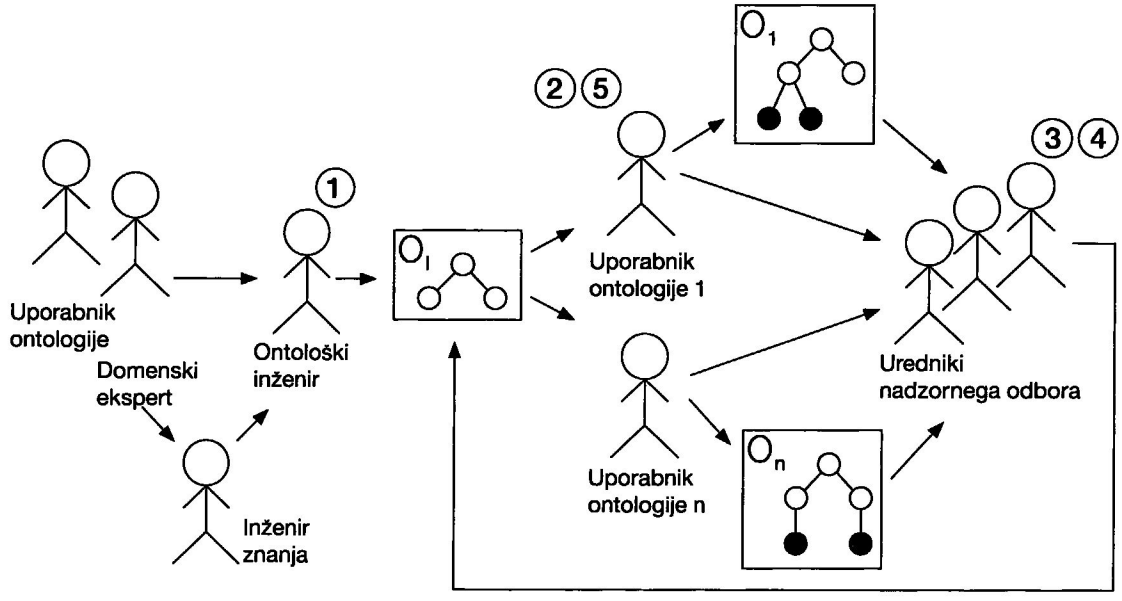
6.3.1 Diligent

Metodologija DILIGENT se precej razlikuje od do sedaj opisanih pristopov. Metodologija ima naslednje posebnosti (Pinto, Tempich, & Staab, 2004):

- Do sedaj opisani pristopi k sistemom znanja so se predvsem osredinjali na centraliziran razvoj ontologij. DILIGENT v nasprotju z obstoječimi pristopi uvaja porazdeljen razvoj ontologij, v katerega so lahko vključeni vsi deležniki ontologije (razvijalci, domenski eksperti, uporabniki).
- Obstoječe metodologije se osredinjajo predvsem na kontrolne sezname, ki nadzorujejo razvojni proces. DILIGENT uvaja dinamičen model razvoja ontologij, t. i. argumentacijski razvoj ontologij (ARO), ki ne predvideva striktnih pravil, temveč določa ogrodje, na podlagi katerega poteka argumentacija doseganja konsenza in uvajanja sprememb pri gradnji ontologij.
- Do sedaj opisane metodologije se v večini osredinjajo zgolj na začetni razvoj ontologij oz. sistemov znanja. Ko je ontologija razvita, doseže v večini modelov svojo zadnjo fazo. DILIGENT predstav-

lja sklenjen življenjski cikel, v katerem ontologija nikdar ne preide v končno fazo, temveč predvideva stalno razvijajoče se ontologije.

Izraz DILIGENT je akronim, ki karakterizira namen metodologije, in sicer: porazdeljeno, šibko nadzorovano in razvijajoče inženirstvo ontologij (angl. DIstributed, Loosely-controlled and evolvInG Engineering of oNTologies). Metodologija je sestavljena iz osnovnega procesa DILIGENT ter argumentacijskega procesa. Prvi proces opisuje gradnjo ontologij po tej metodologiji, medtem ko se drugi osredinja na učinkovito argumentiranje pri procesu razvoja ontologije, kar poveča učinkovitost samega razvoja ontologij z boljšo komunikacijo in s tem boljše doseganje konsenza. Proces razvoja po metodologiji DILIGENT je prikazan na sliki 9.



Slika 9: **Procesi razvoja po metodologiji DILIGENT (prirejeno po Pinto et al. (2004))**

Osnovni proces razvoja ontologij sestavlja pet glavnih aktivnosti: (i) gradnja, (ii) lokalna prilagoditev, (iii) analiza, (iv) revizija in (v) lokalna posodobitev. Proces razvoja se začne tako, da domenski eksperti, uporabniki, inženirji znanja in ontološki inženirji *zgradijo* začetno ontologijo. Gradnja naj bi potekala v t. i. on-line načinu razvoja, v katerem imajo omenjeni deležniki ontologije na razpolago sodelovalna orodja za razvoj ontologije. Razvoj poteka po sodelovalnem principu razvoja, in sicer na podlagi vodene argumentacije, na katero se osredinja argumentacijski proces.

Ko je razvita prvotna različica ontologije, jo lahko uporabniki začnejo uporabljati in *lokalno prilagajati* glede na svoje potrebe. Tako lahko nastane več različic ontologije. Glavni odločitveni korak je, katere funkcionalnosti bo vsebovala naslednja osrednja oz. skupna različica. Uporabniki pošljejo svoje predloge vključno z argumenti urednikom nadzornega odbora.

Nadzorni odbor spremlja razvoj skupne ontologije, prav tako preučuje argumente za spremembe in izvaja odločitve o naslednjih različicah skupne ontologije. Metodologija nalaga odboru redno *revizijo* skupne ontologije, ki zagotavlja, da se lokalne ontologije ne oddaljijo preveč od skupne verzije. Po izdaji nove verzije se pričakuje, da uporabniki posodobijo svoje lokalne ontologije tako, da so skladne s skupno različico, kar zagotavlja kompatibilnost in hkrati omogoča nadaljnjo evolucijo ontologije.

6.5 Metodologija NeOn

Metodologija NeOn nastaja v okviru projekta NeOn, ki ga financira Evropska skupnost v okviru 6. okvirnega programa (predvideni zaključek projekta je marec 2010). Cilj projekta je »... razvoj prve storit-

veno usmerjene, odprte infrastrukture in povezane metodologije za podporo razvoja življenjskih ciklov naslednje generacije semantičnih aplikacij ...«, pri čemer bodo »... aplikacije temeljile na omrežju kontekstno povezanih ontologij, ki izpostavljajo lokalno konsistentnost«.

V okviru projekta nastaja metodologija za razvoj ontologij. NeOn se ne osredinja na razvoj posameznih ontologij, temveč množice oz. omrežja medsebojno povezanih ontologij. Podobno kot DILIGENT se tudi NeOn oddaljuje od statičnih ontologij in zagovarja neprenehoma razvijajoče se ontologije na podlagi sodelovalnega razvoja.

Metodologija še ni povsem razvita, zaradi tega jo bomo po prvotni različici specifikacije, ki je izšla februarja 2008 (Suárez-Figueroa et al., 2008), ter po opisu aktualne definicije razvojnega procesa in življenjskega cikla omreženih ontologij v okviru metodologije NeOn (Suárez-Figueroa et al., 2007).

Avtorji metodologije navajajo naslednje razloge za razvoj nove metodologije (Suárez-Figueroa et al., 2008):

- Obstoječe metodologije ne obravnavajo smernic za razvoj ontologij s ponovno uporabo in reinženirstvom ontologij ter ponovno uporabo obstoječih virov znanja, ki je splošno sprejeto v neki domeni.
- Metodologijam primanjkujejo smernice za kontekstualno povezavo obstoječih ontologij in njihovo vključevanje z obstoječimi ontologijami, ki so lahko v stalni evoluciji.
- Obstoječe metodologije ne obravnavajo procesa razvoja ontologij z enakim pristopom in granularnostjo, kot to počno metodologije razvoja programske opreme.

Avtorji predvidevajo, da bodo semantične aplikacije prihodnosti vsebovale veliko število ontologij, ki so vgrajene v ontološka omrežja. Zaradi tega izpostavljajo tri glavne dimenzije projekta NeOn: (i) sodelovanje, (ii) kontekst in (i) dinamičnost. Če želimo doseči omrežje ontologij, je edini način sodelovalni pristop, ki omogoča distribuiran razvoj ontologij med heterogenimi in geografsko porazdeljenimi skupinami domenskih ekspertov, ontoloških inženirjev ter uporabnikov. Medtem ko se sodelovanja dotika DILIGENT, noben od dosedanjih pristopov ne obravnava zadovoljivo konteksta in dinamičnosti.

Osnovni koncepti v metodologiji so povzeti po že opisanih metodologijah (študija izvedljivosti, kompetenčna vprašanja, sodelovanje, argumentacija, vrednotenje). Metodologija gradi na obstoječih metodologijah, obstoječih metodah za pridobivanje znanja, na podlagi dobrih praks pri razvoju ontologij ter na preteklih izkušnjah pri razvoju ontologij v okviru različnih evropskih in nacionalnih projektov. Pri projektu sodeluje večina ključnih raziskovalnih institucij s področja semantičnega spleta in ontološkega inženirstva (med drugimi tudi Inštitut Jožef Stefan).

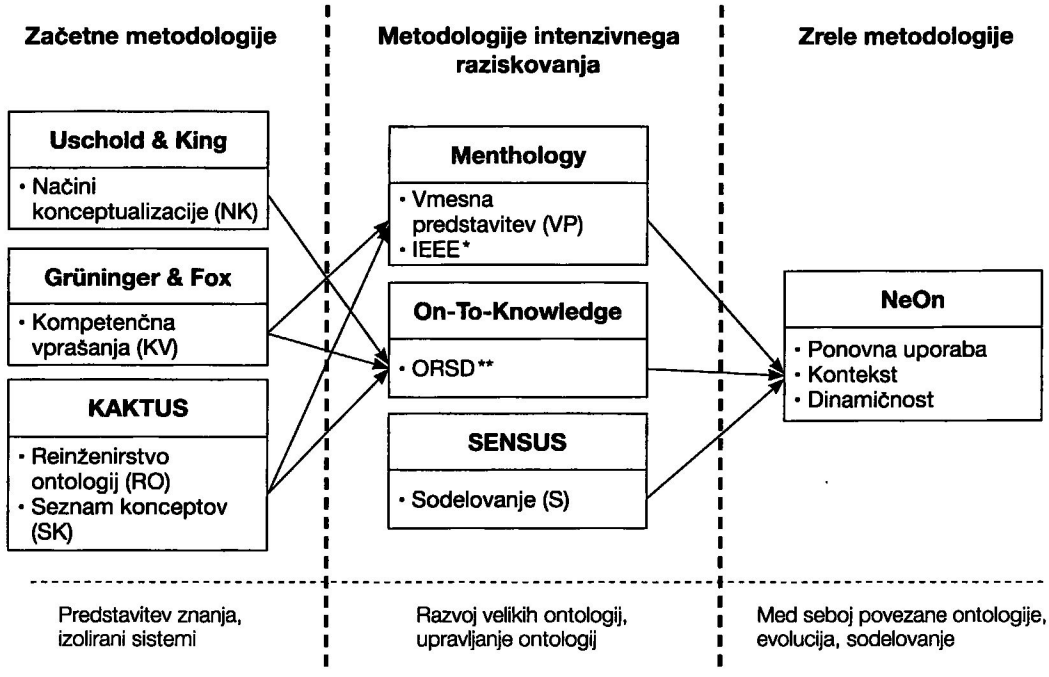
V okviru projekta nastaja tudi orodje, ki bo omogočalo hiter razvoj ontologij po metodologiji NeOn. To je novost glede na obstoječe pristope, saj nobeden od dosedanjih pristopov ni bil neposredno podprt s prirejenim razvojnim orodjem oz. okoljem. Velika šibkost projekta NeOn je velika količina in nepreglednost dokumentacije. Rezultati in koncepti projekta so porazdeljeni po večtisočstranski množici dokumentov brez dokumenta, ki bi povzemal rezultate projekta in omogočal hiter vstop.

7 PRIMERJAVA PRISTOPOV

Če primerjamo različne pristope, opazimo določen linearen trend, kako so novi pristopi povzemali od obstoječih. Na podlagi analize pristopov smo jih uvrstili v tri skupine: (i) začetne metodologije, (ii) metodologije intenzivnega raziskovanja in (iii) zrele metodologije. Začetne metodologije so se osredinjale predvsem na problematiko predstavitve znanja in temeljijo pretežno na izkušnjah pri gradnji sistemov znanja. Pri teh metodologijah je večji poudarek na razvoju izoliranih sistemov, ki temeljijo na znanju.

Metodologije intenzivnega raziskovanja izhajajo iz izkušenj začetnih metodologij ter spoznanj iz vse bolj aktivnega raziskovanja na področju ontologij. Ta vrsta metodologij se predvsem osredinja na razvoj velikih ontologij iz nič. Pristopi v tem obdobju podrobneje naslavljajo aktivnosti upravljanja razvoja ontologij. Zrele metodologije za razvoj ontologij naslavljajo razvoj med seboj povezanih ontologij (omrežene ontologije), poudarjajo sodelovalni pristop razvoja ter izpostavljajo dinamično komponento ontologij (evolucija).

Slika 10 prikazuje umestitev opisanih pristopov v predstavljene kategorije. Za vsak pristop je izpostavljena pomembna tehnika, ki jo je uvedel pristop



* IEEE standard za razvoj življenjskih ciklov programske opreme
** ORSD – dokument specifikacije zahtev ontologij

Slika 10: **Kategorizacija pristopov k razvoju ontologij**

in ki je imela vpliv pri razvoju naslednjih pristopov. Puščice med pristopi ponazarjajo povzemanje tehnik. Iz slike lahko razberemo, da je metodologija NeOn edina zrela metodologija in da glede na povzete tehnike predstavlja dejansko fuzijo dosedanjih pristopov k razvoju ontologij. Ob tem je treba dodati, da metodologija NeOn še ni dokončana. Uporaba vseh tehnik tako še ni natančno specificirana, je pa predvidena.

V tabeli 1 je prikazana primerjava metodologij glede na standard za razvoj procesov življenjskih ciklov programske opreme IEEE (IEEE, 2006). Stolpec v tabeli, ki je označen z 'lastnost', predstavlja aktivnosti, ki jih predpisuje specifikacija. Za vsako aktivnost vsake metodologije je določeno, ali metodologija predlaga aktivnost, jo opisuje, natančno določa ali sploh ne obravnava. Iz tabele je razvidno, da so začetne metodologije precej šibke na področju upravljalskih in podpornih aktivnosti. Metodologije intenzivnega raziskovanja z izjemo metodologije DILIGENT naslavljajo osrednje aktivnosti, vendar jih preveč pogosto samo predpisujejo in premalo podrobno opisujejo. Primerjava je dodelana in prirejena po Gómez-Perez et al. (2004).

Na podlagi tabele 1 je mogoče ugotoviti, da metodologije Methontology, On-To-Knowledge in NeOn bistveno bolj celovito obravnavajo razvoj ontologij kot druge tehnike. Glede na lastnosti bi bila najbolj priporočljiva uporaba metodologije NeOn. Zaradi še nezaključenega projekta in preveč obsežne ter razpršene dokumentacije bi bilo za realne projekte trenutno vredno izbrati Methontology ali On-To-Knowledge ter jo kombinirati z določenimi spoznanji projekta NeOn. Komponente NeOn, ki bi jih bilo vredno vključiti, so ponovna uporaba ter dokumenti specifikacije zahtev ontologij, kot jih definira NeOn.

Preglednica 2 podaja primerjavo metodologij s stališča metodologije NeOn (Suárez-Figueroa et al., 2008). Kot je iz preglednice razvidno, nobena od obstoječih metodologij z izjemo DILIGENT metodologije ne naslavlja aspektov, ki jih naslavlja prvotna verzija metodologije NeOn. DILIGENT sicer zelo obširno naslavlja sodelovanje, vendar je tako rekoč to tudi edino področje, ki ga podrobno opredeljuje ta metodologija. V preglednico niso vključene druge metodologije, ker nobena od njih ne naslavlja izpostavljenih aspektov.

Tabela 1: **Primerjava metodologij po specifikaciji razvoja življenjskih ciklov programske opreme IEEE**[7]

Lastnost			**Uschold & King**	**Grüninger & Fox**	**Methontology**	**On-To-Knowledge**	**Diligent**	**NeOn**
Aktivnost upravljanje	Planiranje		BO	BO	Predlaga	Opisano	BO	Opisano
	Nadzor		BO	BO	Predlaga	Opisano	Opisano	Zahteva
	Zagotovitev kakovosti		BO	BO	Predlaga	Opisano	Opisano	Zahteva
Aktivnosti razvoja	Predrazvojni proces	Študija okolja	BO	BO	BO	Predlaga	BO	Zahteva
		Študija izvedljiv.	BO	BO	BO	Opisano	BO	Zahteva
	Razvojni proces	Specifikacija	Predlaga	Podrobno opisano	Podrobno opisano	Podrobno opisano	BO	Podrobno opisano
		Konceptualizacija	BO	Podrobno opisano	Podrobno opisano	Predlaga	BO	Zahteva
		Formalizacija	BO	Podrobno opisano	Opisano	Opisano	Predlaga	Zahteva
		Implementacija	Predlaga	Podrobno opisano	Podrobno opisano	Opisano	Opisano	Zahteva
	Porazvojni proce	Vzdrževanje	BO	BO	Predlaga	Predlaga	Predlaga	Zahteva
		Uporaba	BO	BO	BO	Predlaga	BO	BO
Podporne aktivnosti	Pridobivanje znanja		Predlaga	Predlaga	Podrobno opisano	Opisano	BO	Delno opisano
	Vrednotenje		Predlaga	Podrobno opisano	Podrobno opisano	Predlaga	Predlaga	Zahteva
	Integracija		Predlaga	Predlaga	Predlaga	Predlaga	BO	Podrobno opisano
	Upravljanje konfiguracije		BO	BO	Opisano	Predlaga	Predlaga	Zahteva
	Dokumentiranje		Predlaga	Predlaga	Podrobno opisano	Opisano	Predlaga	Zahteva
	Spajanja in razvrščanje		BO	BO	BO	BO	BO	Opisano

Tabela 2: **Primerjava metodologij s perspektive metodologije NeOn**

Lastnost	**Methontology**	**On-To-Knowledge**	**Diligent**	**NeOn (ver. 1)**
		Dimenzije NeOn		
Sodelovanje	Ni obravnavano	Ni obravnavano	Obravnavano	Zgolj omenjeno
Kontekst	Ni obravnavano	Ni obravnavano	Ni obravnavano	Ni obravnavano
Dinamičnost	Zgolj omenjeno	Zgolj omenjeno	Zgolj omenjeno	Ni obravnavano
		Podrobne smernice procesov in aktivnosti		
Specifikacija	Predlagana zgolj kompetenčna vpr.	Predlagana zgolj kompetenčna vpr.	Ne predvideva te aktivnosti	Podrobno podano
Ponovna uporaba neontoloških virov	Ni omenjeno	Ni omenjeno	Ni omenjeno	Podano
Reinženirstvo neontoloških virov	Ni omenjeno	Ni omenjeno	Ni omenjeno	Podano v preliminarni obliki
Ponovna uporaba ontologij	Podan zgolj seznam aktivnosti	Podano zgolj priporočilo	Ni omenjeno	Podano
Ponovna uporaba načrtovalskih vzorcev	Ni omenjeno	Ni omenjeno	Ni omenjeno	Podano v preliminarni obliki
		Ciljne skupine		
Ciljna skupina	Ontološki inženirji in raziskovalci	Ni ciljana na ontološke inženirje in raziskovalce	Domenski eksperti in uporabniki	Ontološki inženirji in raziskovalci

[7] BO – Brez obravnave.

8 SKLEP

Razvoj ontologij je zahtevno opravilo, ki ne zahteva samo domenskega znanja, temveč tudi znanje o formalni predstavitvi znanja, poznavanje ontoloških jezikov, znanje o upravljanju razvojnega procesa ter ne nazadnje znanje o sodelovanju med ljudmi (za razvoj ontologij je treba doseči konsenz). Da bi lahko obvladovali to kompleksnost, je modro uporabiti metodologije za razvoj ontologij, ki usmerjajo razvojni proces in nas tako hitreje pripeljejo na cilj, razvite ontologije pa so bolj kakovostne.

Na podlagi spremembe metodologij in aktivnosti na področju inženirstva ontologij je mogoče zaznati trend, da se ontologije pomikajo iz večjih izoliranih ontologij v množico manjših med seboj povezanih ontologij, ki se nenehno razvijajo. Sodobni pristopi uvajajo sodelovalni princip, po katerem člani razvojne skupine na različnih lokacijah skupno razvijajo ontologijo. Te spremembe vnašajo še večjo pomembnost sistematiziranega pristopa k razvoju ontologij.

Trenutno ne obstaja celovita metodologija, ki bi naslavljala ključne aspekte razvoja ontologij. Slednje se odraža tudi v nizki stopnji uporabe metodologij pri razvoju ontologij. Najboljši približek je metodologija NeOn, ki vse te aspekte naslavlja le delno. Žal je še zmeraj v razvojnem obdobju. Po drugi strani pa metodologija NeOn že sedaj vsebuje precej kaotično in obsežno zbirko informacij. Metodologiji, ki najbolje naslavljata omenjene kriterije in se nahajata v končni različici, sta On-To-Knowledge in methontology. Omenjeni metodologiji trenutno najbolj celovito in pregledno naslavljata celotni razvojni cikel ontologij.

9 LITERATURA

[1] Bizer, C. R., Christian. (2006). *D2R Server-publishing relational databases on the Semantic Web* (poster). In 5th International Semantic Web Conference.

[2] Borst, W. N. (1997). *Construction of engineering ontologies. Unpublished doctoral dissertation*, University of Twente.

[3] Cardoso, J. (2007). *The semantic web vision: Where are we?* Intelligent Systems, IEEE, 22 (5), 84–88.

[4] Corcho, O., Fernández-López, M., & Gómez-Pérez, A. (2003). *Methodologies, tools and languages for building ontologies. where is their meeting point?* Data & Knowledge Engineering, 46 (1), 41–64.

[5] Davis, R., Shrobe, H., & Szolovits, P. (1993). *What is a knowledge representation?* AI Magazine, 14 (1), 17–33.

[6] Farquhar, A., Fikes, R., & Rice, J. (1996). *The ontolingua server: a tool for collaborative ontology construction.* In International journal of human-computer studies.

[7] Genesereth, M. R., & Fikes, R. E. (1992). *Knowledge interchange format. version 3.0. reference manual.* (Tech. Rep.). Stanford University.

[8] Gómez-Pérez, A., Corcho, O., & Fernéndez-López, M. (2004). *Ontological engineering: with examples from the areas of knowledge management, e-commerce and the semantic web.* First edition (advanced information and knowledge processing). Springer.

[9] Grimm, S., Hitzler, P., & Abecker, A. (2007). *Semantic Web Services: Concepts, technologies, and applications.* In R. Studer, S. Grimm, & A. Abecker (Eds.), (p. 51–105). Springer-Verlag New York, Inc.

[10] Gruber, T. R. (1993). *A translation approach to portable ontology specifications.* Knowledge Acquisition, 5, 199–220.

[11] IEEE (2006). *IEEE standard for developing a software project life cycle process.* (2006). IEEE Std 1074–2006 (Revision of IEEE Std 1074–1997), 1–104.

[12] Lassila, O., & McGuinness, D. L. (2001). *The role of frame-based representation on the semantic web (Tech. Rep. No. KSL-01-02).* Stanford: Stanford University.

[13] Manola, F., & Miller, E. (Eds.). (2004). *RDF Primer.* World Wide Web Consortium.

[14] McGuiness, D. L. (2004). *OWL Web Ontology Language.* W3C Recommendation.

[15] Nierlich, A. (2007). *Kuka roboter gmbh rolls out the advanced version of kuka.expert plus to customer service employees worldwide.* http://www.ontoprise.de/en/home/news/news-en/kuka/.

[16] Object Management Group (2005). *Ontology definition metamodel, third revised submission to OMG/rfp ad/2003-03-40.* www.omg.org/docs/ad/05-08-01.pdf.

[17] Pinto, H. S., Tempich, C., & Staab, S. (2004). *Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engingeering of ontologies.* In R. L. de Mantaras & L. Saitta (Eds.), Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004), pp. 393–397. Valencia, Spain: IOS Press.

[18] Studer, R., Benjamins, R., & Fensel, D. (1998). *Knowledge engineering: Principles and methods.* Data & Knowledge Engineering, 25 (1–2), 161–198.

[19] Suárez-Figueroa, M. del C., Cea, G. A. de, Buil, C., Caracciolo, C., Dzbor, M., Gómez-Pérez, A., et al. (2007). *D5.3.1 NeON development process and ontology life cycle (NeOn Project Deliverable No. D5.3.1).* Universidad Politecnica de Madrid.

[20] Suárez-Figueroa, M. del C., Cea, G. A. de, Buil, C., Dellschaft, K., Fernández-López, M., Garcia, A., et al. (2008). *D5.4.1. NeON methodology for building contextualized ontology networks (NeOn Project Deliverable No. D5.4.1).* Universidad Politecnica de Madrid.

[21] Sure, Y., Staab, S., & Studer, R. (2004). *On-to-knowledge methodology (OTK).* In S. Staab & R. Studer (Eds.), Handbook on ontologies (p. 117–132). Springer.

[22] Syldatke, T., Chen, W., Angele, J., Nierlich, A., & Ullrich, M. (2007). *How ontologies and rules help to advance automobile development.* In Advances in Rule Interchange and Applications (Vol. 4824/2007, pp. 1–6). Springer Berlin / Heidelberg.

[23] Söderström, E., Andersson, B., Johannesson, P., Perjons, E., & Wangler, B. (2002). *Towards a framework for comparing process modelling languages.* In Advanced information systems engineering (Vol. 2348/2002, pp. 600–611). Springer Berlin / Heidelberg.

[24] Tran, T., Lewen, H., & Haase, P. (2007). *On the role and application of ontologies in information systems.* In 5th IEEE International Conference on Research, Innovation and Vision for the Future (p. 14–21).

■

Boštjan Grašič je diplomiral leta 2006 na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Je študent doktorskega študijskega programa na omenjeni fakulteti in hkrati študent na znanstvenem magistrskem programu na Ekonomsko-poslovni fakulteti Univerze v Mariboru. Zaposlen je kot mladi raziskovalec na Fakulteti za elektrotehniko, računalništvo in informatiko. Njegovo raziskovalno področje obsegajo semantični splet in semantične tehnologije, semantične spletne storitve ter razvoj naprednih inteligentnih sistemov. Je avtor več znanstvenih prispevkov na domačih in tujih konferencah.

■

Vili Podgorelec je izredni profesor s področja informatike na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru, kjer predava na programih računalništvo in informatika, informatika in tehnologije komuniciranja, medijske komunikacije in bioinformatika. Raziskovalno se ukvarja predvsem s področji inteligentnih sistemov, inovativnih informacijskih rešitev, semantičnih tehnologij in teorije kompleksnosti, ki jih aplicira predvsem v programskem inženirstvu in medicinski informatiki. Je avtor mnogih člankov z omenjenih raziskovalnih področij v uglednih mednarodnih revijah, vabljeni predavatelj na več konferencah ter predsednik oz. član programskih odborov in soorganizator nekaj mednarodnih znanstvenih konferenc. Sodeloval je v več domačih in mednarodnih znanstvenoraziskovalnih projektih ter v aplikativnih projektih za industrijo.

■

Marjan Heričko je redni profesor in vodja laboratorija za informacijske sisteme na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegovo raziskovalno-razvojno delo obsega vse vidike razvoja informacijskih rešitev in storitev s poudarkom na metodologijah razvoja, ponovni uporabi, metrikah, upravljanju znanja, storitvenem inženirstvu in storitveni znanosti. Je strokovni koordinator slovenske tehnološke platforme za programsko opremo in storitve (NESSI Slovenija), predsednik konference OTS Sodobne tehnologije in storitve ter mednarodne konference CSS Collaboration, Software and Services in Information Society, vodja in koordinator mnogih domačih in mednarodnih projektov.