2004 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XII

UPORABNA INFORMATIKA

Testni centri ECDL

ECDL (European Computer Driving License), ki ga v Sloveniji imenujemo evropsko računalniško spričevalo, je standardni program usposabljanja uporabnikov, ki da zaposlenim potrebno znanje za delo s standardnimi računalniškimi programi na informatiziranem delovnem mestu, delodajalcem pa pomeni dokazilo o usposobljenosti. V Evropi je za uvajanje, usposabljanje in nadzor izvajanja ECDL pooblaščena ustanova ECDL Foundation, v Sloveniji pa je kot član CEPIS (Council of European Professional Informatics Societies) to pravico pridobilo Slovensko društvo INFORMATIKA. V državah Evropske unije so pri uvajanju ECDL močno angažirane srednje in visoke šole, aktivni pa so tudi različni vladni resorji. Posebej pomembno je, da velja spričevalo v več kot osemdesetih državah, ki so vključene v program ECDL. Doslej je bilo v svetu izdanih že več kot tri milijone indeksov, v Sloveniji okoli 1700 in podeljenih okoli tisoč spričeval. Za testne centre ECDL so se v Sloveniji usposobile organizacije, katerih logotipi so natisnjeni na tej strani.

ISER

AGORA
ustvarjalne komunikacije

IZOBRAŽEVANJE
INFORMACIJSKE STORITVE

ELES - ICES

e-trade
Elit trade Zagreb

EUROCOM
Simply logical

UNIVERZA V LJUBLJANI
Fakulteta za pomorstvo in promet

ISA
INFORMACIJSKE
TEHNOLOGIJE

IVANUŠA
Contact um

KOPA

LJUDSKA UNIVERZA KOPER
UNIVERSITÀ POPOLARE CAPODISTRIA

Micro Team

Much

spin

UPI LJUDSKA UNIVERZA ŽALEC
šola prijaznih ljudi

VLS
computers

UPORABNA
INFORMATIKA
2004 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XII ISSN 1318-1882

Uvodnik

Razprave

Rešitve

Poročila

Dogodki in odmevi

Koledar prireditev

ISSN 1318-1882

Ustanovitelj in izdajatelj:
Slovensko društvo INFORMATIKA
Vožarski pot 12
1000 Ljubljana

Predstavnik
Niko Schlamberger

Odgovorni urednik:
Andrej Kovačič

Tehnična urednica
Mira Turk Škraba

Oblikovanje
Bons

Prelom
Dušan Weiss, Ada Poklač

Tisk
Prograf

Naklada
700 izvodov

Naslov uredništva
Slovensko društvo INFORMATIKA
Uredništvo revije Uporabna informatika
Vožarski pot 12, 1000 Ljubljana
www.drustvo-informatika.si/posta

Revija izhaja četrtletno. Cena posamezne številke je 4.500 SIT. Letna naročnina za podjetja 17.800 SIT, za vsak nadaljnji izvod 11.900 SIT, za posameznike 5.900 SIT, za študente 2.800 SIT.

Revijo sofinancira Ministrstvo za šolstvo, znanost in šport RS.

Revija Uporabna informatika je od številke 4/VII vključena v mednarodno bazo INSPEC.

Revija Uporabna informatika je pod zaporedno številko 666 vpisana v razvid medijev, ki ga vodi Ministrstvo za kulturo RS.

Navodila avtorjem

Revija Uporabna informatika objavlja izvirne prispevke domačih in tujih avtorjev na znanstveni, strokovni in informativni ravni. Namenjena je najširši strokovni javnosti, zato je zaželeno, da so tudi znanstveni prispevki napisani čim bolj poljudno.

Članke objavljamo praviloma v slovenščini, prispevke tujih avtorjev v angleščini.

Prispevki so obojestransko anonimno recenzirani. Vsak članek za rubriko Razprave mora za objavo prejeti dve pozitivni recenziji. O objavi samostojno odloča uredniški odbor.

Prispevki naj bodo lektorirani, v uredništvu opravljamo samo korekturo. Po presoji se bomo posvetovali z avtorjem in članek tudi lektorirali. Prispevki za rubriko Razprave naj imajo dolžino do 40.000, prispevki za rubrike Rešitve, Poročila do 30.000, Obvestila pa do 8.000 znakov.

Naslovu prispevka naj sledi ime in priimek avtorja, ustanova, kjer je zaposlen in elektronski naslov. Članek naj ima v začetku do 10 vrstic dolg izvleček v slovenščini in angleščini, v katerem avtor opiše vsebino prispevka, dosežene rezultate raziskave. Abstract se začne s prevodom naslova v angleščino. Članku dodajte kratek avtorjev življenjepis (do 8 vrstic), v katerem poudarite predvsem delovne dosežke.

Pišite v razmaku ene vrstice, brez posebnih ali poudarjenih črk, za ločilom na koncu stavka napravite samo en prazen prostor, ne uporabljajte zamika pri odstavkih.

Revijo tiskamo v črno-beli tehniki s folije, zato barvne slike ali fotografije kot originali niso primerne. Objavljali tudi ne bomo slik zaslonov, razen če niso nujno potrebne za razumevanje besedila. Slike, grafikoni, organizacijske sheme ipd. naj imajo belo podlago. Po možnosti jih pošiljajte posebej, ne v datoteki z besedilom članka. Disketi z besedom priložite izpis na papirju.

Prispevke pošiljajte po elektronski ali navadni pošti na naslov uredništva revije: ui@drustvo-informatika.si, Slovensko društvo INFORMATIKA, Vožarski pot 12, 1000 Ljubljana. Za dodatne informacije se obračajte na tehnično urednico Miro Turk Škraba.

Po odločitvi uredniškega odbora o objavi članka bo avtor prejel pogodbo, s katero bo prenesel vse materialne avtorske pravice na Slovensko društvo INFORMATIKA. Po izidu revije pa bo prejel nakazilo avtorskega honorarja po veljavnem ceniku ali po predlogu odgovornega urednika.

Spoštovane bralke in bralci,

v enem od uvodnikov revije Uporabna informatika, ko se je "zgodila informatika" v NLB, sem ugotavljal premalo pomembno in odmevno vlogo informatikov ter ignoranco menedžerjev do uporabe poslovne informatike v slovenskih organizacijah. Tudi potek dogodkov in kasnejšega revidiranja informacijske rešitve v NLB je potrdil mojo slutnjo in trditev, da bi bilo bolj smotrno revidirati "lastnike", menedžment in vodstvo področja informatike.

Zaradi splošne razsežnosti tega problema smo njegovi širši osvetlitvi in poskusu reševanja namenili letošnje posvetovanje Dnevi slovenske informatike v Portorožu. Po končanem posvetovanju Menedžment in informatika ugotavljamo dobro odzivnost direktorjev in vodij informatike, ki so se ga udeležili v znatno večjem številu kot navadno. Na drugi strani pa žal ni bilo čutiti povečanega odziva krovnega menedžmenta naših organizacij. Kot da se ne zavedajo problema in pomena tesnega in neposrednega sodelovanja z informatiki na takšnih projektih.

Posvetovanje je potrdilo ugotovitve, da ostaja prenova poslovanja in neprestano prilagajanje programskih rešitev edina stalnica v hitro se spreminjajočem poslovnem svetu oz. poslovnem okolju. Korenite in stalne spremembe ne vplivajo le na potrebo po prenavljanju poslovanja, temveč tudi na prilagajanje informacijske podpore poslovanju. Nove razmere zahtevajo prenovo poslovnega modela ter učinkovito upravljanje s poslovnimi procesi in programskimi rešitvami. Pri tem je bila poudarjena problematika miselne vrzeli med menedžerji in informatiki, možnosti in priložnosti informacijske tehnologije ter vplivnosti informacijske podpore na poslovno uspešnost in konkurenčnost organizacije.

Tudi okrogla miza Partnerstvo menedžmenta in informatike je ugotovila, da informacijska tehnologija, orodja in rešitve niso čarobna paličica, s katero bi reševali vse probleme. Nasprotno – podobno kot pri zdravilih –, neuporaba, neprimerna ali neustrezna uporaba lahko tudi v procesu informatizacije povzročijo pogubne posledice. Pri nas menedžerji navadno niso dovolj aktivni, odločitve prepuščajo informatikom, pogosto pa kar ponudnikom informacijske tehnologije. Na drugi strani pa nekatere organizacije preveč stavijo na tehnologijo, na orodja za modeliranje poslovnih procesov. Pri tem se pretirano ubadajo z analizo obstoječih procesov, zmanjka pa jim časa za njihovo uspešno poslovno prenovo. V teh primerih, po takšni analizi-paralizi si menedžerji upravičeno zastavljajo vprašanja o koristnosti modeliranja procesov, o potrebnosti podrobnega analiziranja stanja informatike in procesov ... in na koncu o vzrokih za neuspeh projektov informatizacije.

Ne glede na gornje dileme ugotavljamo, da so večje možnosti uspešne prenove in informatizacije poslovanja pri tistih projektih, pri katerih je stalno prisotna vodilna in usmerjevalna vloga menedžmenta oz. zagotovljen poslovni vidik in pristop k informatizaciji poslovanja. Izhodišče takšnega pristopa je za oba partnerja v procesu (menedžerja in informatika) jasen in nedvoumen poslovni model in iz njega izhajajoči modeli poslovnih procesov. Menedžment se mora odločati o prioritetah in intenzivnosti informatizacije predvsem na podlagi njenega vpliva na poslovanje. Le z aktivno vlogo na projektu lahko menedžment premosti ali odpravi tradicionalni prepad med »poslovanjem« in »informatiko«.

Na povetovanju smo na omenjeno problematiko poskusili odgovoriti v nekaj tematskih sklopih, vezanih na odnos med menedžmentom in informatiko, odprli pa smo tudi več novih področij, ki jim bomo v prihodnosti namenili prostor v naši reviji.

Odgovorni urednik
Andrej Kovačič

Vabilo k sodelovanju pri tematski številki revije Uporabna informatika

Teorija in praksa elektronskega poslovanja

Leta 2000 je Slovensko društvo INFORMATIKA izdalo svojo vizijo o možnem načinu prehoda Slovenije iz industrijske v informacijsko družbo in jo objavilo v posebni številki Uporabne informatike z naslovom *Slovenija kot informacijska družba*. Znana je kot *Modra knjiga*, ki je postala del političnega programa slovenske vlade. V Modri knjigi smo opredelili bistvene elemente, ki jih mora vsebovati tak dokument, definirali informacijsko družbo in pokazali, kako je mogoče iz tedanje razvojne stopnje preiti v informacijsko družbo. Kot soustanovitelj slovenskega foruma za informacijsko družbo smo izkazali svoje prepričanje, da samo programski dokument ni dovolj. Predlagali smo slovenske Bangemannove aplikacije, ki bi podobno kakor evropske pripomogle k zavedanju o nujnosti razvojne usmeritve in olajšale prehod vsem. Delo, tj. program prihodnjega razvoja Slovenije v Evropski uniji, ne more biti opravljeno v enem zamahu. Načrtovali smo separate k Modri knjigi, ki bi obravnavali aktualna področja in jih podrobneje obdelali s teoretičnih in praktičnih vidikov. S tem smo želeli ponuditi tudi praktične napotke raziskovalcem, gospodarskim družbam, državi in nevladnim organizacijam. Delo bomo nadaljevali in pričujoči prispevek je povabilo k sodelovanju pri nadaljevanju Modre knjige.

Prvi separat, ki ga namerava SDI izdati kot nadaljevanje začetega dela, je tematska številka Uporabne informatike, ki bo izšla letos jeseni pod naslovom *Teorija in praksa elektronskega poslovanja*. Odločitev za to področje ni naključna; Slovenija ima vsaj dve desetletji tradicije razvoja področja, ki je bilo sprva omejeno na računalniško izmenjavanje podatkov, za katerega se je izkazalo, da je le tehnika, nujno potrebna za izvajanje poslovnih in upravnih funkcij, ki jih s skupnim imenom označujemo kot elektronsko poslovanje. Informacijska revolucija ni toliko posledica uporabe računalnika ali nastanka interneta, temveč novega načina delovanja, ki sta ga omogočila – elektronskega poslovanja. Le-to je tisto, ki radikalno spreminja gospodarstvo, politiko in družbo nasploh. Samo pomislimo na e-Bay, Amazon, .com družbe, delo na daljavo, globalizacijo poslovanja, konvergenco tržišč, e-demokracijo in sorodne pojave, ki so znanilci novih paradigem na vseh področjih življenja in dela. Po ocenah se bo do leta 2006 okrog štirideset odstotkov ameriškega poslovanja odvijalo prek interneta.

Vabilo k sodelovanju SDI naslavlja na vse, ki bi želeli s svojimi prispevki sodelovati pri nastajanju separata k Modri knjigi. Prispevki naj obravnavajo teoretične ali praktične, tehnološke ali uporabniške vidike iz gospodarstva, uprave, izobraževanja in znanosti, pri čemer so dobrodošli tudi pogledi državljanov in nevladnih organizacij. Separat naj bi obravnaval elektronsko poslovanje s čim več različnih vidikov, da bi mogli vsi, ki jih to zanima ali ki imajo v tem pogledu celo obveznosti, ugotoviti, kaj je v tem za Slovenijo, kako ravnati in seveda tudi, kako se usposobiti za nove oblike delovanja.

Prispevke z oznako Tema2004 pošljite na naslov info@drustvo-informatika.si do 1. septembra 2004.

Dr. Andrej Kovačič	Aljoša Domijan	Niko Schlamberger
odgovorni urednik	gostujoči urednik	gostujoči urednik

Prenova poslovnih procesov in uspešnost slovenskih podjetij

Fabris Peruško
Halcom Informatika, d. o. o. Ljubljana
fabris.perusko@halcom.si

Povzetek

Prenova poslovnih procesov je v zadnjih dvajsetih letih postala popularno menedžersko orodje. Cilj članka je osvetliti stanje prenove poslovnih procesov v slovenskih podjetjih in presoditi o vplivu izvajanja prenove poslovnih procesov na uspešnost poslovanja podjetja. Kakor se je izkazalo doslej, je glavna težava pri tovrstnih raziskavah opredeljevanje kazalnikov uspešnosti podjetja. V naši raziskavi je uporabljen pristop za ocenjevanje uspešnosti podjetja, ki temelji na modelu uravnoteženih kazalnikov (angl. Balanced Scorecard – BSC). Uspešnost podjetja je tako ocenjena s štirih vidikov: s finančnega vidika, z vidika kupcev, z vidika notranjih poslovnih procesov in z vidika učenja in rasti, kar nadalje opredeljujejo štiri skupine kazalnikov. Raziskava je pokazala statistično značilno povezanost med uspešnostjo prenove poslovnih procesov in uspešnostjo poslovanja podjetij z vseh vidikov razen s finančnega vidika.

Abstract

Reformance of Slovenian Companies in the Light of Reengineered Bussines

Business process reengineering (BPR) has become a popular managerial tool in the last two decades. The goal of this article is to highlight the state of business process reengineering Processes in Slovenian companies and to judge the influence of BPR on the performance of those companies. As we have often seen, the main obstacle during this kind of research is defining the indicators of company's performance. In our research, we have used the approach to assess company's performance, which is based on the Balanced Scorecard model. Company's performance is thus evaluated from four perspectives: financial perspective, customer perspective, internal-business-process perspective and learning and growth perspective. Those four perspectives are further defined by four groups of indicators.

The research has showed a statistically significant correlation between the success of business process reengineering and the performance of companies from all the above mentioned perspectives except the financial perspective.

1 Uvod

Sredi osemdesetih let prejšnjega stoletja so idejo o prenovi poslovnih procesov napovedala pomembna svetovalna podjetja, kot so Peat Marwick in McKinsey. Podjetje Index Group in Michael Hammer sta raziskovala številna podjetja, vključno z Mutual Benefit Life in Ford. Ta podjetja so uporabljala številne elemente prenove poslovnega sistema, še zlasti idejo o uporabi informacijske tehnologije (IT) za doseganje radikalnih sprememb v medfunkcijskih procesih (Grover, Malhotra, str. 196, 1997).

V teh letih je prišlo tudi do nekaj velikih sprememb v poslovnem okolju podjetij. Ena od njih je zagotovo izjemen tehnološki napredek, predvsem razvoj informacijskih tehnologij in telekomunikacij. Nadalje, to je obdobje začetka globalizacije in liberalizacije razmer v svetovnem gospodarstvu. Omenjene spremembe so nas pripeljale do okolja, ki ni predvidljivo, saj ni mogoče natančno predvideti ne rasti trga in ne povpraševanja ali življenjskega cikla proizvoda.

Tri gonilne sile, ločeno ali v sodelovanju, danes vodijo podjetja globlje in globlje v področje, v katerem se menedžment počuti prestrašeno in neprijetno. Na kratko jih imenujemo »3C«: kupec (angl. *customer*), konkurenčnost (angl. *competition*) in spremembe (angl. *change*). Povzemamo jih po delu Hammerja in Champya (Hammer, Champy 1993, str. 17–30):

- kupci – današnji se precej razlikujejo od nekdanjih; so segmentirani in pričakujejo nasvet;
- konkurenčnost – narašča konkurenčnost, katere cilj je zadostiti potrebam kupcev;
- spremembe – so postale vseobsegajoče, nenehne, hitrejše in nujno potrebne.

Poleg že omenjenih sprememb v poslovnem okolju je konec osemdesetih in začetek devetdesetih let v svetu zaznamovala občutna gospodarska recesija. Ta je od podjetij, na začetku predvsem v ZDA, zahtevala drastično zmanjšanje stroškov, novo okolje pa nov

način poslovanja. Recesija je spodbudila menedžerje, da razmišljajo o novem načinu zmanjševanja stroškov. Rastoča globalna konkurenca je stiskala dobiček in vodila do reaktivnega pristopa ter do programov zmanjšanja stroškov in tako imenovanega »downsizinga«. Cilj teh programov je med drugim bil tudi povečanje sposobnosti odzivanja in fleksibilnosti podjetij (Grover, Malhotra, str. 196, 1997). To je bil temeljni razlog, da so se podjetja lotila prenove poslovnih procesov.

Fenomen prenove poslovnih procesov sta utemeljila dva članka na to temo, katerih avtorji so Davenport in Short (1990) ter Hammer (1990). Hammer in Champy sta prenovo poslovnega procesa opredelila kot na novo premišljen in na novo načrtovan poslovni proces, namenjen doseganju dramatičnega napredka v ključnih, sodobnih merilih učinkovitosti, kot so cena, kakovost storitev in hitrost (Hammer in Champy, 1993). Prenova poslovnih procesov je uveljavila procesni pristop oziroma procesno naravnanost organizacije namesto funkcijske. Ta procesni pristop se v osnovi povsem razlikuje od pristopa Adama Smitha, ki je skušal razbiti proces na manjše in ponavljajoče se naloge. Pri izvajanju projekta prenove poslovnega procesa sam projekt ne zahteva samo temeljite spremembe poslovnega procesa, ampak tudi vse povezane poslovne elemente, kot so organizacijska struktura, nadzorni mehanizmi, sistem nagrajevanja. Pri svojem izvajanju porablja znatne organizacijske vire in potrebuje podporo ključnih članov organizacije.

Podjetja, ki se odločajo za prenovo poslovnih procesov, se nemalokrat srečujejo s kopico težav. Najbolj pogoste so težave z upravljanjem sprememb (*angl. change management*), kratkoročno gledanje vodilnega menedžmenta, okorela organizacijska struktura, pomanjkljivi ali neprilagojeni človeški in finančni viri, omejene zmožnosti informacijskih tehnologij in strokovnjakov v podjetju, pomanjkanje podpore članov organizacije za prenovo, pomanjkanje zagovornikov prenove, težave v medfunkcijskem sodelovanju in pri prepoznavanju »pravega« procesa ter številne druge (Ranganathan, Dhaliwal, str. 132, 2001). Pomembnost vsake omenjene težave je odvisna od kulturnih, organizacijskih, sociooloških in drugih dejavnikov notranje organizacije in od okolja, v katerem podjetje deluje.

Zgoraj omenjene težave vplivajo na uspešnost prenove poslovnega procesa. Izkušnje kažejo, da programi prenove poslovnega procesa propadejo v 70 odstotkih primerov (O'Neill, Sohal, 1999, str. 573). Ta odstotek se spreminja od države do države, znotraj le-teh pa se uspešnost spreminja glede na lastniško strukturo podjetja – ali gre za državna, lokalna ali multinacionalna podjetja (Ranganathan, Dhaliwal, 2001, str. 133).

Ko v luči prenove poslovnih procesov govorimo o slovenskih podjetjih, moramo upoštevati še nekaj dodatnih dejavnikov okolja. Domača podjetja so se v začetku devetdesetih let prejšnjega stoletja spopadala s kopico težav, kot so izguba relativno velikega jugoslovanskega tržišča, prehod iz planskega socialističnega gospodarstva v sodobno kapitalistično družbo, trenutno pa se srečujejo še z dodatnim izzivom – priključitvijo v polnopravno članstvo Evropske unije. Za slovenska podjetja imajo takšne okoliščine dvojni pomen; po eni strani izgubljajo zaščito na domačem trgu, po drugi pa dobivajo dostop do izjemno konkurenčnega in ogromnega trga Unije. Vse skupaj pa nas pripelje do ugotovitve, da je potreba slovenskih podjetij po prenovi poslovnih procesov in s tem po doseganju konkurenčnosti na globalnem tržišču še bistveno večja, kot je pri podjetjih razvitih zahodnih gospodarstev.

2 Raziskava Prenova poslovnih procesov in uspešnost slovenskih podjetij

Raziskava Prenova poslovnih procesov in uspešnost slovenskih podjetij je bila izpeljana v okviru Inštituta za poslovno informatiko Ekonomske fakultete Univerze v Ljubljani. Cilj raziskave je bil osvetliti stanje prenove poslovnih procesov v slovenskih podjetjih in presoditi o vplivu izvajanja prenove poslovnih procesov v podjetjih na uspešnost poslovanja podjetja. Potreba po ugotavljanju povezav med projekti prenove poslovnih procesov in uspešnostjo poslovanja podjetja postaja še bolj nazorna pri pregledu obstoječih raziskav s tega področja v slovenskem gospodarstvu.

Obstoječe raziskave namreč zgolj deskriptivno ugotavljajo stanje, vendar ne ugotavljajo vpliva teh procesov na uspešnost poslovanja podjetij. Tudi v številnih raziskavah in literaturi v tujini ne najdemo veliko takšnih, ki skušajo ugotoviti vpliv prenove poslovnih procesov na uspešnost poslovanja podjetij. V strokovnih krogih je tako že bila prepoznana potreba po raziskovanju povezave med uporabo orodij in tehnik prenove poslovnega procesa ter uspešnostjo poslovanja (O'Neill, Sohal, 1999, str. 579).

Glavna težava pri tovrstnih raziskavah je opredeljevanje kazalnikov uspešnosti podjetja. V zadnjem desetletju so namreč vse bolj glasni zagovorniki nefi-

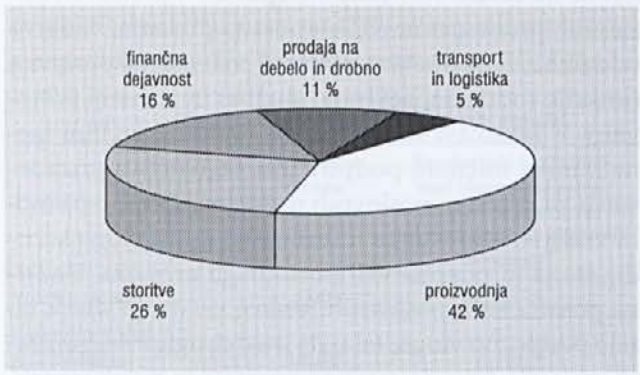
nančnih informacij, nekateri celo zagovarjajo potrebo po nadomestitvi finančnih kazalnikov z nefinančnimi. Večina pa poudarja komplementarnost obeh in prav iz tega razloga je v raziskavi uporabljan pristop za ocenjevanje uspešnosti podjetja, ki temelji na modelu uravnoteženih kazalnikov (angl. *The Balanced Scorecard – BSC*). Uspešnost podjetja je tako ocenjena s štirih vidikov: s finančnega vidika, z vidika kupcev, z vidika notranjih poslovnih procesov in z vidika učenja in rasti, kar nadalje opredeljujejo štiri skupine kazalnikov. Tako bomo na eni strani imeli finančni vidik kot rezultat, ki opredeljuje interese lastnikov in kaže trenutno uspešnost, in na drugi strani tri skupine nefinančnih dejavnikov, ki so dejavniki uspešnosti in vplivajo na prihodnje finančne tokove.

2.1 Metodologija in vzorec

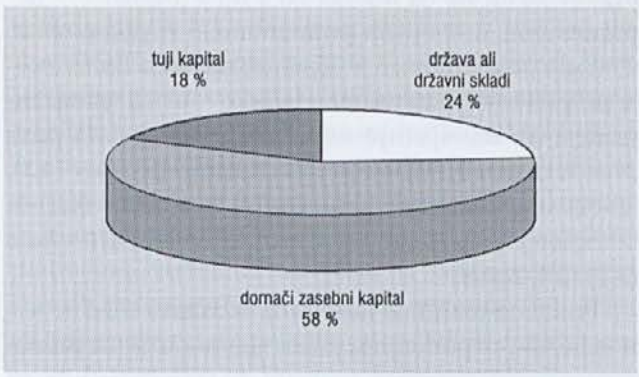
Raziskava je bila izpeljana v zadnjem četrtletju leta 2002. Podatki so zbrani na osnovi vprašalnika o različnih vidikih prenove poslovnih procesov. Vprašanja so pokrila naslednja področja: (i) splošne podatke o organizaciji; (ii) položaj organizacije med tradicionalno in projektno organizirano organizacijo; (iii) vpliv informacijske tehnologije na uspešnost delovanja organizacije; (iv) uspešnost organizacije; (v) motive za prenovo poslovnih procesov; (vi) vlogo posameznih skupin udeležencev pri projektih prenove poslovnih procesov ter (vii) oceno stopnje prispevka projektov prenove na uspešnost organizacije.

Vprašalnik je bil pripravljen na podlagi predhodnih raziskav Inštituta za poslovno informatiko, podobnih raziskav v tujini in dostopne literature. Pred pošiljanjem so vprašalnik pregledali trije slovenski menedžerji in predavatelj z Ekonomske fakultete.

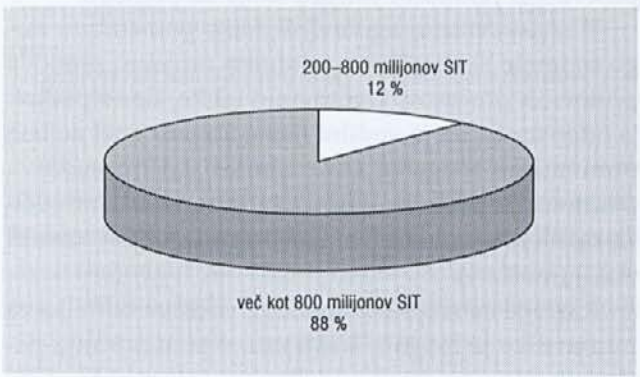
Vprašalnik je bil poslan dvestotim podjetjem v Sloveniji (30 največjih podjetij po prihodku, 30 največjih podjetij po dobičku in 140 podjetji, ki so sodelovala pri raziskavi Inštituta za poslovno informatiko »Poslovna informatika 2001«). V raziskavi je pridobljenih 19 uporabnih izpolnjenih vprašalnikov ali 9,5 odstotka odgovorov. Slike 1, 2, 3 in 4 prikazujejo osnovne značilnosti vzorca podjetij, ki so sodelovala v raziskavi.



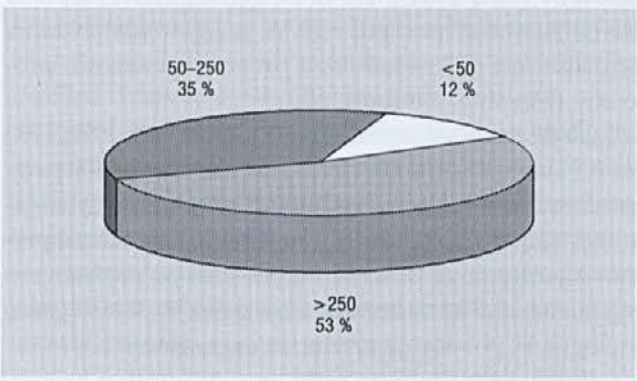
Slika 1: **Glavni vir prihodkov podjetij**



Slika 2: **Prevladujoči lastnik**



Slika 3: **Letni prihodek podjetij**



Slika 4: **Število zaposlenih v podjetjih**

V raziskavi smo preverjali naslednje postavljene hipoteze:

- Hipoteza 1: Večja vloga posameznih skupin igralcev v projektih prenove poslovnih procesov zagotavlja večji uspeh teh projektov.
- Hipoteza 2: Uporaba informacijske tehnologije v projektih prenove zagotavlja večji uspeh takšnih projektov.
- Hipoteza 3: Naravnanost k procesni organizaciji omogoča boljše rezultate prenove poslovnih procesov.
- Hipoteza 4: Rezultati prenove poslovnih procesov vplivajo na uspešnost organizacije.
- Hipoteza 5: Naravnanost k procesni organizaciji vpliva na uspešnost organizacije.

3 Prenova poslovnih procesov v slovenskih podjetjih

3.1 Motivi za prenovo poslovnih procesov

Motivi za prenovo poslovnih procesov so lahko različni. Raziskava Poslovna informatika 2001, ki se redno izvaja v okviru Inštituta za poslovno informatiko, ugotavlja, da sta na lestvici od 1 do 5 (ocene: 1 = nepomembno, 2 = včasih pomembno, 3 = pomembno, 4 = zelo pomembno, 5 = ključno) največja motivatorja pri procesih prenove dvig učinkovitosti in skrajšanje časa ter izboljšanje uspešnosti (oba imata povprečno oceno 4,3). Sledijo jim znižanje stroškov (4,1), izboljšanje kakovosti proizvodov in storitev (4,1) in izboljšanje prijaznosti ali razpoložljivosti partnerjem (3,9) (IPI, 2002).

Podobne rezultate kaže tudi raziskava med severnoameriškimi podjetji, ki je kot glavni motiv za prenovo poslovnih procesov prepoznala povečanje hitrosti izvajanja poslovnega procesa in takoj za tem znižanje stroškov (CSC Index, 1994). Druga raziskava, izpeljana na vzorcu 80 podjetij v ZDA, je ugotovila znižanje stroškov kot glavni motiv za prenovo poslovnih procesov (Maglitta, 1995, str. 20).

Glede na izsledke raziskav, ki jih je v letih 1992, 1993 in 1994 izvajalo ameriško svetovalno podjetje Gateway med vodilnimi menedžerji ameriških podjetij (Manganelli in Klein, 1994, str. 12), so motive za začetek prenove poslovnega procesa razporedili po naslednjem vrstnem redu: konkurenca, tržni delež in dobiček, tehnologija in povečanje vrednosti organizacije.

Po enakem vrstnem redu smo ponudili v ocenjevanje (ocene: 1 = popolnoma nevpliven, 2–3 = delno vpliven, 4 = vpliven, 5–6 = zelo vpliven, 7 = ključnega pomena) tudi motive v naši raziskavi, vendar so podjetja, ki so delala prenovo poslovnih procesov, ocenila tehnologijo z oceno 5,60 kot najvplivnejši motiv, zaradi katerega se lotijo prenove poslovnih procesov. Vrstni red preostalih omenjenih motivov je ostal enak kot v primerljivi raziskavi med ameriškimi podjetji: konkurenca (5,40), tržni delež in dobiček (5,29) in povečanje vrednosti organizacije (4,00). Seveda je treba na primerjavo teh rezultatov gledati z zadržkom, saj je primerljiva ameriška raziskava stara že 9 let.

Iz rezultatov torej lahko sklepamo, da se podjetja za prenovo poslovnih procesov odločajo zaradi pritiska možnosti, ki jih ponuja tehnologija, in ne toliko zaradi neposrednega pritiska konkurence.

3.2 Vloge posameznih skupin v projektih prenove poslovnih procesov

Ko govorimo o sodelovanju in pomenu različnih skupin v projektih prenove poslovnih procesov, se vprašamo, kakšen pomen naj bi le-ti imeli.

Ranganathan in Dhaliwal (2001, str. 128) sta v raziskavi med singapurskimi podjetji povzela iz strokovne literature štiri skupine igralcev v prenovah poslovnih procesov: najvišje vodstvo podjetja, vodstvo oddelka za informatiko, vodstvo posamezne poslovne funkcije (oddelka, sektorja, službe) in zunanji svetovalci. V raziskavi sta ugotovila, da mora vodilni menedžment inicirati, podpirati in zagovarjati prizadevanja za prenovo poslovnih procesov, sočasno pa mora vodstvo oddelka za informatiko igrati vlogo koordinatorja in pospeševalca celotnega projekta. Vodstva posameznih poslovnih funkcij po drugi strani igrajo podporno vlogo, tako da med drugim v organizaciji objavijo prizadevanja za prenovo in potek projekta. Zunanji svetovalci pa, če so najeti, pomagajo pospeševati in podpirati prizadevanja za prenovo.

V raziskavi smo ugotavljali, kako pomembno vlogo so imele te skupine v celotnem projektu prenove poslovnih procesov. Ugotovimo lahko, da so podjetja odgovorila, da je vodilni menedžment imel najbolj pomembno vlogo (6,46), za njim sledijo vodstva posameznih poslovnih funkcij (5,62), vodstvo oddelka za informatiko (4,85) in na koncu zunanji svetovalci (4,23) (tabela 1).

Razloge za nižjo oceno vloge vodstva oddelka za informatiko je mogoče iskati tudi v pomanjkanju poslovnih znanj vodstev oddelkov za informatiko. Takšno tezo ugotavljajo nekateri raziskovalci, podpirajo

pa jo tudi rezultati naše raziskave. Boljše poslovno znanje v teh projektih daje prednost in postavlja v bolj pomembno vlogo vodstva posameznih poslovnih funkcij (Bates, 1995, str. 134 in Maglitta, 1995, str. 20). Prav zaradi tega nekateri avtorji poudarjajo potrebo po sodelovanju med vodstvi oddelkov za informatiko in posameznih poslovnih funkcij (Ray, 1995, str. 135).

Nekatere študije priporočajo, naj zaposleni z oddelkov za prenovo za izboljšanje rezultatov prenove razvijejo veščine analiziranja organizacije in nadgradijo znanja o strategiji organizacije (Teng, Fiedler, Grover, 1998, str. 696).

Če ta spoznanja uporabimo pri analizi rezultatov raziskave, vidimo, da so najvišja vodstva in vodstva posameznih poslovnih funkcij v organizaciji igrala najbolj pomembne vloge pri projektih prenove v slovenskih podjetjih (tabela 1). Vendar so korelacijski koeficienti med vlogo posamezne funkcije in uspešnostjo projekta prenove precej raznoliki. Vloga vodstva oddelka za informatiko kakor tudi vloga vodstva posameznih poslovnih funkcij kažeta visok korelacijski faktor z oceno uspeha prenove poslovnih procesov in tudi statistično značilnost (tabela 1). Ta rezultat nas opozarja, da je treba posebno pozornost pri takšnih projektih usmeriti prav na ti dve skupini, saj je njun prispevek k uspehu projekta nedvomno ključen.

Tabela 1: **Vloge posameznih skupin v projektih prenove**

	Povprečne ocene	Korelacijski koeficienti med vlogo posamezne skupine in ocenjenim uspehom prenove poslovnih procesov[1]
Najvišje vodstvo	6,46	0,4186
Vodstvo oddelka za informatiko	4,85	0,7363**
Vodstvo posamezne poslovne funkcije (oddelka, sektorja, službe)	5,62	0,6074*
Zunanji svetovalci	4,23	0,1740

Ocene:
1 = popolnoma nevpliven, 2–3 = delno vpliven, 4 = vpliven, 5–6 = zelo vpliven, 7 = ključnega pomena

Hipoteza 1 je tako statistično potrjena za vlogo dveh skupin igralcev, in sicer vodstva oddelka za informatiko in vodstva posameznih poslovnih funkcij. Vloga teh dveh skupin namreč zagotavlja večjo uspešnost projektov prenove, kaže visok korelacijski koeficient in je statistično značilen z uspehom prenove.

Pomen vodstva podjetja je dobil visoko povprečno oceno za vlogo v tem projektu, vendar njegova vloga ne kaže visokega korelacijskega koeficienta in ni statistično značilna z uspehom prenove. Vloga zunanjih svetovalcev pri procesih prenove je bila najmanjša in z uspehom teh procesov kaže majhen korelacijski koeficient in ni statistično značilna.

Med razlogi za manjšo vlogo zunanjih svetovalcev pri prenovi poslovnih procesov je zagotovo tudi na splošno nerazvita kultura najemanja zunanjih svetovalcev v slovenskem gospodarstvu.

3.3 Informacijska tehnologija in prenova poslovnih procesov

Raziskave po svetu kažejo na tesno povezanost prenove poslovnih procesov in informacijske tehnologije. Ranganathan in Dhaliwal (2001, str. 130) sta v že omenjeni raziskavi med singapurskimi podjetji ugotovila, da se od informacijske tehnologije največ uporabljajo podatkovne baze in z njimi povezane tehnologije, sledijo mreže in komunikacije, celovite rešitve (ERP), internet in WEB tehnologije, elektronska izmenjava podatkov in podobno.

V raziskavi med slovenskimi podjetji nismo ugotavljali, katere tehnologije uporabljajo v prenovi poslovnih procesov, pač pa smo skušali ugotoviti, kakšen je vpliv informacijske tehnologije na poslovanje podjetja in kakšno vlogo ima informacijska tehnologija v prenovi poslovnih procesov. Kot je razvidno iz rezultatov raziskave (tabela 2), informacijska tehnologija, ki se uporablja v podjetjih, zelo vpliva na uspešnost poslovanja, vendar nima vpliva na uspeh prenove poslovnih procesov. To pomeni, da trenutno uporabljana informacijska tehnologija ne vpliva na rezultat prenove. Nasprotno pa informacijska tehnologija, uporabljena v projektih prenove, igra pomembno vlogo pri uspešnosti prenove poslovnih procesov. *Vpliv informacijske tehnologije na uspešnost projekta prenove kaže visok korelacijski koeficient in ta povezava je statistično značilna. S tem smo potrdili hipotezo 2.*

Podoben rezultat vpliva informacijske tehnologije na splošno uspešnost poslovanja in njen vpliv v projektih prenove kaže na to, da podjetja s terminom prenova poslovnih procesov ne razumejo zgolj preproste informatizacije svojega poslovanja, ampak da verjetno iščejo priložnosti za napredek tudi v sami organizaciji procesa.

[1] * = *Raven statistične značilnosti 0,05*
** = *Raven statistične značilnosti 0,01*

Tabela 2: **Vloga informacijske tehnologije**

	Povprečne ocene	Korelacijski koeficienti med merjenimi vplivi in ocenjenim uspehom prenove poslovnih procesov[2]
Vpliv informacijske tehnologije, ki jo uporablja organizacija, na uspešnost delovanja organizacije	5,81	0,3926
Pomembnost in vpliv informacijske tehnologije na prenovo poslovnih procesov	5,46	0,7125**

Ocene:
1 = popolnoma nevpliven, 2–3 = delno vpliven, 4 = vpliven, 5–6 = zelo vpliven, 7 = ključnega pomena

3.4 Prenova poslovnih procesov in procesna organizacija

Hammer (2002) navaja, še posebej v svojih novejših člankih, potrebo po prehodu organizacije iz tradicionalnega podjetja v procesno podjetje. Prav tako Hammer in Champy v svojih zgodnjih člankih in knjigah o prenovi poslovnih procesov (1993 in 2001) nove organizacije ne imenujeta procesno podjetje, vendar naštejeta podobne značilnosti, ki sledijo organizaciji po prenovi poslovnih procesov.

Tabela 3: **Tradicionalno in procesno podjetje**

	Tradicionalno podjetje	Procesno podjetje
Centralna os	funkcija	proces
Delovna enota	oddelek	skupina
Opis dela	določen	širok
Merilo	ozko	od začetka do konca
Osredinjen na	nadrejenega	stranko
Nadomestilo temelji na	aktivnosti	rezultatih
Menedžersko pravilo	nadzor	mentor
Ključna osebnost	funkcijski izvršitelj	lastnik procesa
Kultura	konfliktno naravnana	sodelovanje

Vir: Hammer, 2002, str. 28

Podjetja, ki izpeljejo prenovo poslovnih procesov, najbolj pogosto doživijo eno ali več sprememb, ki jih povzema tabela 3, in sicer: delovne enote se spremenijo iz funkcijskih oddelkov v procesne skupine; dela se spremenijo iz preprostih nalog v vseobsegajoča dela; vloge ljudi se zamenjajo iz nadzornih v mentorske; priprava na delo se spremeni iz urjenja v izobraževanje; osredotočenost merjenja uspešnosti poslovanja in nagrajevanja se preusmeri od dejavnosti k rezultatom; spremenijo se merila za napredovanje, in sicer od učinka k sposobnostim; nadalje, vrednote se spremenijo od zaščitnih k produktivnim (ne dela se več za nadrejenega, temveč za kupca – kupec »plačuje« za plačo, ne nadrejeni); menedžerji se spremenijo iz nadzornikov v mentorje; organizacijska struktura se spremeni iz hierarhične v enakopravno, izvršni delavci pa se spremenijo iz zapisnikarjev v vodje (Hammer, Champy, str. 69–86, 2001).

Da bi raziskali povezavo med procesno organizacijo in prenovo poslovnih procesov, smo merili dve spremenljivki. Prva je ocena prispevka prenove poslovnih procesov k uspešnosti organizacije. Menedžment sodelujočih podjetij je ocenjeval, v kolikšni meri je prenova prispevala k uspešnosti organizacije, z ocenami od 1 do 7 (1 = ni prispevka; 7 = ključne izboljšave). Druga spremenljivka, ki smo jo merili, je bila ocena trenutnega položaja organizacije med tradicionalno in procesno usmerjeno organizacijo, z ocenami od 1 do 7 (1 = tradicionalna organizacija; 7 = procesna organizacija). Kot smo že omenili, v svojih novejših člankih Hammer (2002) pogosto navaja potrebo po prehodu organizacije iz tradicionalnega v procesno podjetje. Prav zato smo tudi v naši raziskavi vprašali podjetja, kako bi se opredelila v tej razdelitvi, in naredili povezavo uspešnosti podjetij in njihovega položaja na lestvici. Povprečna ocena prispevka dosedanjih projektov prenove poslovnega procesa na uspešnost poslovanja organizacije v naši raziskavi je bila 4,85. Ocena položaja podjetja med procesno naravnano (ocena 7) in tradicionalno organizacijo (ocena 1) sodelujočih v raziskavi je bila 4,35.

Sung in Gibson (1998) sta pri merjenju prispevka prenove poslovnih procesov na uspešnost poslovanja korejskih podjetij pri enako postavljeni lestvici dobila nekoliko višjo povprečno oceno 5,49.

Pri testu hipoteze 3, da je uspeh prenove poslovnih procesov odvisen od stopnje naravnanosti k procesno usmerjeni organizaciji, smo dobili korelacijski koeficient med rezultati teh dveh meritev 0.8716 z ravnjo statistične značilnosti pod 0,01. S tem je hipoteza 3 potrjena.

Pri tem rezultatu obstajata še odprti vprašanji, ali je trenutna raven procesne naravnanosti posledica prenove poslovnih procesov in ali procesno naravnana organizacija odpira večje možnosti za uspeh projektov prenove. Po naši oceni se oba vpliva medsebojno prepletata. Da bi iz tradicionalnih postala procesno naravnana, podjetja namreč potrebujejo prenovo

[2] * = *Raven statistične značilnosti 0,05*
** = *Raven statistične značilnosti 0,01*

poslovnih procesov. Po drugi strani pa, čim bolj je organizacija procesno naravnana, tem boljši rezultat lahko zagotovi prenova takega procesa.

3.5 Prenova poslovnih procesov, procesna organizacija in njena uspešnost

Glede na teorijo in prakso prenove poslovnih procesov je temeljni razlog za začetek projektov prenove povečanje uspešnosti organizacije. Zato je bil eden od poglavitnih ciljev raziskave ugotoviti povezavo uspeha prenove poslovnih rezultatov in uspešnosti organizacije.

Uspešnost organizacije je preveč kompleksna kategorija, da bi jo lahko ocenili samo z eno mero ali eno številko. Prav zaradi tega smo pristop ocenjevanja uspešnosti organizacije temeljili na modelu uravnoteženih kazalnikov (angl. *The Balanced Scorecard – BSC*). Ocenili smo uspešnost organizacije s štirih vidikov oziroma s štirimi skupinami kazalcev in s skupaj 27 kazalci (Kaplan, Norton, 1996). Za posamezne kazalce smo se odločili po priporočilih modela uravnoteženih kazalnikov in po usklajevanju s sodelujočimi pri sestavljanju vprašalnika in tako dobili nabor 27 kazalcev. Da so bili kazalci pravilno izbrani, potrjuje tudi visoka ocena, ki so jih kazalci dobili pri ocenjevanju njihove pomembnosti za doseganje uspeha organizacije.

Uporabljeni in merjeni vidiki in kategorije za določanje uspešnosti organizacije so naslednji:

- Finančni vidik, ki odraža uspešnost z vidika lastnika podjetja. Ocenili smo ga z naslednjimi kazalci:
  - rast prihodkov in
  - rast dobička pred obdavčitvijo med fiskalnima letoma 2001 in 2000.
- Vidik kupca, s katerim menedžment spremlja, kako poslovanje podjetja vrednoti kupec. Ocenili smo ga z naslednjimi kazalci:
  - splošno zadovoljstvo kupcev,
  - uspešnost pri pridobivanju novih kupcev,
  - velik delež stalnih kupcev,
  - visoka profitabilnost na kupca,
  - kakovost izdelka/storitev,
  - vrednost, ki jo pridobi kupec glede na ceno izdelka/storitev,
  - hitrost dostave izdelka/storitev,
  - sposobnost prepoznavanja potreb kupcev,
  - dostopnost izdelka/storitev (npr. delovni čas, geografska dostopnost),
  - hitrost odziva na zahteve in potrebe kupcev ter
  - ugled in sloves organizacije.
- Vidik notranjih poslovnih procesov, ki vključuje kazalnike za notranje procese. To so procesi, kjer se mora podjetje najbolj odlikovati, če želi zadovoljiti kupce in lastnike. Ta vidik smo ocenili z naslednjimi kazalci:
  - sposobnost razvijanja novih izdelkov,
  - hitrost razvijanja novih izdelkov,
  - uspešnost novih izdelkov na trgu,
  - stroški proizvodnje,
  - kakovost dela (odpad, neučinkovito delo, vrnjeni izdelki),
  - čas, potreben za proizvodnjo/storitev, ki se trenutno ponuja,
  - poprodajna podpora kupcu ter
  - čas med prodajo in sprejetim plačilom.
- Vidik učenja in rasti prek izbranih kazalnikov odraža sposobnost zaposlenih, kakovost sistemov in organizacijskih postopkov v podjetju, ki so osnova za organizacijsko učenje in rast. Vidik učenja in rasti smo ocenili z naslednjimi kazalci:
  - zadovoljstvo zaposlenih,
  - pogostost odhoda in spreminjanja zaposlenih,
  - produktivnost zaposlenih,
  - zmožnosti uporabljenega informacijskega sistema,
  - motiviranost zaposlenih ter
  - predlogi in uvajanje rešitev, ki jih predlagajo zaposleni.

Tako bomo po eni strani merili finančni vidik kot rezultat, ki opredeljuje interese lastnikov in kaže na trenutno uspešnost podjetja, po drugi strani pa tri skupine nefinančnih dejavnikov, ki so dejavniki uspešnosti in ki vplivajo na prihodnje finančne tokove.

To nam je omogočilo tudi, da smo uspešnost določili z uporabo tako subjektivnih kot objektivnih mer uspešnosti. Finančni vidik je značilna objektivna mera uspešnosti, ki ima po eni strani prednost, saj jo lahko natančno merimo, vendar ima tudi najmanj tri pomanjkljivosti, gledano s stališča ocenjevanja vpliva prenove poslovnih procesov na uspešnost organizacije. Prva pomanjkljivost je, da prenova ni edini dejavnik, ki lahko vpliva na finančne rezultate. Druga pomanjkljivost je, da imajo efekti prenove časovni zaostanek in verjetno ne bodo takoj vidni na finančnih rezultatih. Tretja pomanjkljivost je v obstoju verjetnosti, da prenova ne bo vplivala neposredno na finančni uspeh organizacije, temveč bo predvsem vplivala na poslovne vrednote, prepričanja, procese in infrastrukturo, kar pa je težje meriti (Sung, Gibson, 1998, str. 304).

Prav zaradi omenjenih pomanjkljivosti objektivnega dela ocene uspešnosti smo se poslužili tudi subjektivnih

mer. Pri subjektivnih merah uspešnosti je menedžment ocenil uspešnost svoje organizacije z ocenami od 1 do 7 (1 = precej slabše od konkurence; 4 = enako kot pri konkurenci; 7 = precej boljše od konkurence).

Zaradi velikega števila kazalnikov uspešnosti smo merili v dveh dimenzijah, in sicer:

- kako je organizacija uspešna pri določenem vidiku in kazalcu glede na konkurenco ter
- koliko so posamezni vidiki in kazalci pomembni pri doseganju ciljev organizacije.

Pomen posameznega kazalca je ocenjen z oceno od 1 do 7 (1 = popolnoma nepomembno, 2–3 = delno pomembno, 4 = pomembno, 5–6 = zelo pomembno, 7 = ključnega pomena). Tako smo dobili dva podatka o uspešnosti organizacije v določenem vidiku in utež tega vidika za doseganje cilja organizacije. Z drugim podatkom smo izračunali uspešnost, ki smo jo poimenovali ponderirana uspešnost. Določili smo jo takole:[3]

ponderirana uspešnost =
= (ocena glede na konkurenco – 4) x ocena pomembnosti

Rezultati o uspešnosti podjetij, ki so sodelovala v raziskavi, so predstavljeni v tabeli 4.

Tabela 4: **Ocene uspešnosti po posameznih vidikih**

		Ocena[4]	Ocena[5]	Ponder[6]
		Uspešnost glede na konkurenco	Pomen pri doseganju ciljev organizacije	Ponderirana uspešnost
FINANČNI VIDIK				
1.	Rast prihodkov od prodaje	1,13	6,06	6,80
2.	Rast dobička pred obdavčitvijo	1,77	5,53	8,86
VIDIK KUPCA				
3.	Splošno zadovoljstvo kupcev	5,18	6,47	8,00
4.	Uspešnost pri pridobivanju novih kupcev	4,71	6,18	5,00
5.	Velik delež stalnih kupcev	5,88	6,18	11,94
6.	Visoka profitabilnost na kupca	4,24	5,35	1,74
7.	Kakovost izdelka/storitev	5,41	6,47	9,59
8.	Vrednost, ki jo pridobi kupec glede na ceno izdelka/storitev	5,47	6,29	9,76
9.	Hitrost dostave izdelka/storitev	5,24	5,88	7,59
10.	Sposobnost prepoznavanja potreb kupcev	5,00	6,29	6,94
11.	Dostopnost izdelka/storitev (npr. del. čas, geogr. dostopnost)	4,88	6,06	5,88
12.	Hitrost odziva na zahteve in potrebe kupcev	5,18	6,12	8,12
13.	Ugled in sloves organizacije	5,65	6,18	10,94
NOTRANJI PROCESI				
14.	Sposobnost razvijanja novih izdelkov	4,88	5,59	5,65
15.	Hitrost razvijanja novih izdelkov	4,88	5,71	5,88
16.	Uspešnost novih izdelkov na trgu	4,59	6,00	4,35
17.	Stroški proizvodnje	4,56	5,75	3,44
18.	Kakovost dela (odpad, neučinkovito delo, vrnjeni izdelki)	5,13	6,13	7,13
19.	Čas, potreben za proizvodnjo/storitev, ki se trenutno ponuja	5,13	6,07	6,73
20.	Poprodajna podpora kupcu	5,44	6,00	9,19
21.	Čas med prodajo in sprejetim plačilom	4,88	5,44	5,25
VIDIK UČENJA IN RASTI				
22.	Zadovoljstvo zaposlenih	4,65	5,88	5,12
23.	Pogostost odhoda in spreminjanja zaposlenih	5,00	5,35	5,65
24.	Produktivnost zaposlenih	5,06	6,12	6,76
25.	Zmožnosti uporabljenega informacijskega sistema	4,59	6,47	4,18
26.	Motiviranost zaposlenih	4,94	6,12	6,41
27.	Predlogi in uvajanje rešitev, ki jih predlagajo zaposleni	4,71	5,76	5,35

[3] *Finančne ponderirane kazalce določa naslednja formula = (fiskalno leto 2001/2000) x ocena pomembnosti*

[4] *Ocene: 1 = precej slabše od konkurence; 4 = enako kot pri konkurenci; 7 = precej boljše od konkurence.*

[5] *Ocene: 1 = popolnoma nepomembno, 2–3: delno pomembno, 4 = pomembno, 5–6 = zelo pomembno, 7 = ključnega pomena*

[6] *Ponderirana vrednost = (Ocena glede na konkurenco – 4) x ocena pomembnosti. Finančne ponderirane kazalce določa naslednja formula = (fiskalno leto 2001/2000) x ocena pomembnosti*

Oceno uspešnosti posameznih vidikov glede na konkurenco smo ocenili tako, da smo izračunali srednjo aritmetično vrednost vseh kazalnikov pri posameznem vidiku uspešnosti. Rezultati so povzeti v tabeli 5.

Tabela 5: **Uspešnost po posameznih vidikih**

Vidiki uspešnosti	Uspešnost glede na konkurenco	Pomen pri doseganju ciljev	Ponderirana uspešnost
Finančni vidik	1,41[7]	5,79	7,63
Vidik kupca	5,17	6,13	7,78
Vidik notranjih procesov	4,84	5,79	5,49
Vidik učenja in rasti	4,82	5,95	5,58

Če iz tabel 6 in 7 analiziramo vpliv prenove poslovnih procesov na uspešnost organizacije, merjeno glede na konkurenco, kakor tudi ponderirano uspešnost po različnih vidikih, vidimo, da je prenova poslovnih procesov vplivala z visokim korelacijskim koeficientom na notranje procese (0,8744 in 0,7701), na vidik kupca (0,7987 in 0,8251) in na vidik učenja in rasti (0,6731 in 0,6963) in da je s temi tremi elementi statistično značilno povezana. Ponderirani kazalniki uspešnosti nam pri vseh vidikih uspešnosti prikazujejo celo najvišjo stopnjo statistične značilnosti. Finančni vidik uspešnosti ne kaže statistično značilne povezave in nima visokega korelacijskega koeficienta. Glede na to,

Tabela 6: **Analiza uspešnosti poslovanja glede na konkurenco[8]**

		Prispevek BPR k uspešnosti organizacije – korelacijski koeficient	*Procesna organizacija – korelacijski koeficient*
FINANČNI VIDIK		0,0409	0,1422
1.	Rast prihodkov od prodaje	0,0009(-)	0,0478(-)
2.	Rast dobička pred obdavčitvijo	0,0411	0,1461
VIDIK KUPCA		0,7987**	0,6300**
3.	Splošno zadovoljstvo vaših kupcev	0,5432	0,3380
4.	Uspešnost pri pridobivanju novih kupcev	0,6807*	0,4562
5.	Velik delež stalnih kupcev	0,4847	0,3153
6.	Visoka profitabilnost na kupca	0,6818*	0,6546**
7.	Kakovost izdelka/storitev	0,4241	0,3153
8.	Vrednost, ki jo pridobi kupec glede na ceno izdelka/storitev	0,6439*	0,5131*
9.	Hitrost dostave izdelka/storitev	0,6662*	0,5057*
10.	Sposobnost prepoznavanja potreb kupcev	0,7421**	0,5417*
11.	Dostopnost izdelka/storitev (npr. del. čas, geogr. dostopnost)	0,1828	0,2798
12.	Hitrost odziva na zahteve in potrebe kupcev	0,7363**	0,7635**
13.	Ugled in sloves organizacije	0,6988**	0,5808*
NOTRANJI PROCESI		0,8744**	0,6787**
14.	Sposobnost razvijanja novih izdelkov	0,7123**	0,7001**
15.	Hitrost razvijanja novih izdelkov	0,7248**	0,5392*
16.	Uspešnost novih izdelkov na trgu	0,7073**	0,6190**
17.	Stroški proizvodnje	0,5851*	0,2710
18.	Kakovost dela (odpad, neučinkovito delo, vrnjeni izdelki)	0,5751	0,5026
19.	Čas, potreben za proizvodnjo/storitev, ki se trenutno ponuja	0,3617	0,1955
20.	Poprodajna podpora kupcu	0,5202	0,3100
21.	Čas med prodajo in sprejetim plačilom	0,7131**	0,3834
VIDIK UČENJA IN RASTI		0,6731*	0,5220*
22.	Zadovoljstvo zaposlenih	0,6508*	0,5026*
23.	Pogostost odhoda in spreminjanja zaposlenih	0,0270	0,0000
24.	Produktivnost zaposlenih	0,5458	0,5468*
25.	Zmožnosti uporabljenega informacijskega sistema	0,6420*	0,6790**
26.	Motiviranost zaposlenih	0,6379*	0,4481
27.	Predlogi in uvajanje rešitev, ki jih predlagajo zaposleni	0,6764*	0,4572

[7] *Fiskalno leto 2001/2000*

[8] * = *Raven statistične značilnosti 0,05;* ** = *Raven statistične značilnosti 0,01*

Tabela 7: **Analiza ponderirane uspešnosti poslovanja**[9]

	Prispevek BPR k uspešnosti organizacije – korelacijski koeficient	Procesna organizacija – korelacijski koeficient
FINANČNI VIDIK	0,2270	0,2146
1. Rast prihodkov od prodaje	0,0517	0,0034
2. Rast dobička pred obdavčitvijo	0,2991	0,2542
VIDIK KUPCA	0,7701**	0,5949*
3. Splošno zadovoljstvo kupcev	0,5802*	0,3590
4. Uspešnost pri pridobivanju novih kupcev	0,6818*	0,4401
5. Velik delež stalnih kupcev	0,4446	0,3087
6. Visoka profitabilnost na kupca	0,6753*	0,6506*
7. Kakovost izdelka/storitev	0,4249	0,3120
8. Vrednost, ki jo pridobi kupec glede na ceno izdelka/storitev	0,6774*	0,5364*
9. Hitrost dostave izdelka/storitev	0,6810*	0,4950*
10. Sposobnost prepoznavanja potreb kupcev	0,7248**	0,5280*
11. Dostopnost izdelka/storitev (npr. del. čas, geogr. dostopnost)	0,0665	0,1977
12. Hitrost odziva na zahteve in potrebe kupcev	0,7174**	0,7259**
13. Ugled in sloves organizacije	0,6974**	0,5346*
NOTRANJI PROCESI	0,8251**	0,5939*
14. Sposobnost razvijanja novih izdelkov	0,6736*	0,6134**
15. Hitrost razvijanja novih izdelkov	0,6884**	0,4614
16. Uspešnost novih izdelkov na trgu	0,6617*	0,5442*
17. Stroški proizvodnje	0,5729*	0,4354
18. Kakovost dela (odpad, neučinkovito delo, vrnjeni izdelki)	0,5639	0,4356
19. Čas, potreben za proizvodnjo/storitev, ki se trenutno ponuja	0,2452	0,1372
20. Poprodajna podpora kupcu	0,5188	0,2585
21. Čas med prodajo in sprejetim plačilom	0,7215**	0,3516
VIDIK UČENJA IN RASTI	0,6963**	0,5066*
22. Zadovoljstvo zaposlenih	0,6293*	0,4281
23. Pogostost odhoda in spreminjanja zaposlenih	0,0528	0,0342
24. Produktivnost zaposlenih	0,5548*	0,5180*
25. Zmožnosti uporabljenega informacijskega sistema	0,6434*	0,6724**
26. Motiviranost zaposlenih	0,6884**	0,4483
27. Predlogi in uvajanje rešitev, ki jih predlagajo zaposleni	0,6585*	0,3772

da obstajajo statistično značilne povezave med ostalimi vidiki delovanja organizacije in uspešnostjo prenove, lahko sklepamo, da bo rezultat na finančnem vidiku verjetno viden v prihodnosti. *S tem smo hipotezo 4 potrdili, kar pomeni, da prenova poslovnih procesov vpliva na uspešnost delovanja podjetja, merjeno z vidika kupca, z vidika notranjih procesov in z vidika učenja in rasti.*

Iz istih tabel je razvidno tudi, da procesna naravnanost organizacije podjetja vpliva na uspešnost organizacije, merjeno glede na konkurenco kakor tudi na ponderirano uspešnost, po različnih vidikih in z visokim korelacijskim koeficientom – na vidik kupca (0,6300 in 0,5949), na notranje procese (0,6787 in 0,5939) in na vidik učenja in rasti (0,5220 in 0,5066). Vpliv na vidik kupca in na vidik notranjih procesov kaže najvišjo stopnjo statistične značilnosti. Finančni vidik uspešnosti ne kaže statistično značilne povezave, vendar pa kaže nekoliko večji korelacijski koeficient pri ponderirani uspešnosti poslovanja. Glede na to, da obstaja statistična povezava med ostalimi vidiki delovanja organizacije in uspešnostjo prenove, lahko sklepamo, da bo rezultat na finančnem vidiku verjetno viden v prihodnosti. *S tem smo hipotezo 5 potrdili za vidik kupca, za vidik notranjih procesov in za vidik učenja in rasti, kar pomeni, da procesno naravnana organizacija vpliva na uspešnost delovanja organizacij.*

4 SKLEP

Čeprav prenova poslovnih procesov počasi izginja kot najbolj udaren termin v strokovni literaturi, bo zagotovo

[9] * = *Raven statistične značilnosti 0,05;* ** = *Raven statistične značilnosti 0,01*

še naprej ostala priljubljeno menedžersko orodje. Dosedanja uporaba tega programa sprememb je bila v praksi kompleksna in večplastna, zato je tudi o njegovi prihodnosti treba govoriti z več vidikov. Tako lahko prepoznamo nekaj trendov, ki so opazni že danes, kot so denimo: prenačrtovanje notranjih procesov se vse bolj nadomešča s prenovo medorganizacijskih procesov (Sandberg, 2001; Champy, 2002); namesto načina za doseganje znižanja stroškov, postaja prenova poslovnih procesov v praksi vse bolj program za iskanje novih priložnosti za rast (Sandberg, str. 3, 2001); v nasprotju z dosedanjimi izkušnjami, ko je bila prenova izpeljana v »back office« (v tovarnah in v skladiščih), bo v prihodnosti vse bolj uporabljana v tako imenovanem »front office« in na dohodkovno-proizvodni strani, torej pri razvoju izdelka, pri prodaji in trženju (Hammer, Champy, 2001, str. 5); in na koncu omenimo še trend združevanja prenove z uporabo informacijske tehnologije v nov koncept, ki ga imenujemo poslovni inženiring (angl. *business engineering* – BE) (Bosilj Vukšić, Kovačič, 2002).

Kakor so pokazali rezultati opravljene raziskave, so slovenske organizacije relativno dobro seznanjene s prenovo poslovnih procesov in v veliki večini verjamejo, da je to način za izboljšanje uspešnosti podjetja. Podjetja se prenove v glavnem lotevajo zaradi povečanja učinkovitosti, skrajšanja proizvodnega cikla in izboljšanja uspešnosti (dobičkonosnosti). Razlogi, ki so jih spodbudili k prenovi, so v prvi vrsti zmožnosti tehnologije in šele po tem pritisk konkurence in želja po povečanju tržnega deleža.

Pri projektih prenove v slovenskih organizacijah igrajo najpomembnejšo vlogo vodstva podjetij, medtem ko so vodstva oddelkov za informatiko in vodstva posameznih poslovnih funkcij zelo povezana z uspešnostjo projektov. Najpogosteje pa se prenavljajo temeljni poslovni procesi, kot so prodaja, nabava in proizvodnja.

Opazimo lahko, da je uspeh prenove poslovnih procesov odvisen od stopnje naravnanosti k procesno usmerjeni organizaciji. Uspešnost prenove poslovnih procesov v slovenskih organizacijah ni pokazala statistično značilne povezave s finančnimi rezultati, pokazala pa je tesno povezanost z uspešnostjo organizacije, in sicer z vidika kupca, notranjih procesov in z vidika učenja in rasti. Upravičeno lahko domnevamo, da bo rezultat pri finančnem vidiku uspešnosti organizacije viden v prihodnosti.

Za potrebe nadaljnjega raziskovanja vpliva prenove poslovnih procesov na uspešnost poslovanja podjetja bi bilo treba: (1) ponoviti raziskavo na večjem vzorcu podjetij; (2) povezati časovno komponento konca projekta prenove in vpliva na uspešnost poslovanja ter (3) premisliti o vpeljavi drugačnih načinov merjenja uspešnosti podjetja.

5 LITERATURA

1. Bates S. E.: *What is IS role in reenginering? Business leaders must lead*, Computer 129 (39), 1995, str. 134.
2. Bosilj Vukšić, Kovačič: Uporabna informatika, Ljubljana, 2002.
3. Champy J.: *X-Engineering the Corporation: Reinventing Your Business in the Digital Age*. Warner Books, New York, 2002, str. 30.
4. CSC/Index: *State of reenginering Report*, CSC Index, 1994.
5. Davenport T. H., Short J. E.: *The New Industrial Engineering: Information Technology And Business Process Redesign*. Sloan Management Review, 1990.
6. Grover V., Malhotra M. K.: *Business Process Reengineering: A Tutorial on the Concept, Evolution, Method, Technology And Application*. Journal of Operation Management, 15, 1997, str. 193–213.
7. Hammer M., Champy J.: *Re-Engineering the Corporation: A Manifesto for Business Revolution*. Harper Business, 1993.
8. Hammer M., Champy J.: *Re-Engineering the Corporation: A Manifesto for Business Revolution*. Harper Business, 2001.
9. Hammer M.: *Process Management and the Future of Six Sigma*. MIT Sloan Management Review. Winter 2002, str. 30–42.
10. Hammer M.: *Reengineering Work: Don't Automate, Obliterate*. Harvard Business Review. 1993.
11. Kaplan R. S., Norton D. P.: *The Balance Scorecard*. Harvard Business School Press, Boston, 1996, str. 9–54.
12. IPI – Inštitut za poslovno informatiko, Ekonomska fakulteta: Rezultati ankete Poslovna informatika 2001. Interni material. Ljubljana, 2002, str. 1–19.
13. Maglitta J.: *IS seen as reenginering blockade*, Computerworld 29 (24), 1995, str. 20.
14. Manganelli R. L. , Klein M. M.: *The Reengineering Handbook: A Step-by-Step Guide to Business Transformation*. Amacom, New York, 1994, str. 5–310.
15. O'Neill P., Sohal A. S.: *Business Process Reengineering: A Review of Recent Literature*. Technovation, 19, 1999, str. 571–581.
16. Ranganathan J., Dhaliwal J. S.: *A Survey of Business Process Reengineering Practices In Singapore. Information & Management*, 39, 2001, str. 125–134.
17. Ray J.: *Wath is IS role in reenginering? IS pros shoud be treated as equals*, Computerworld 29 (39), 1995, str. 135.
18. Sandberg K.D.: Reenginering Tries a Comeback – This Time for Growth, Not Just for Cost Savings. Harvard Managament, Boston, 2001, str. 1–4.
19. Sung T. K., Gibson V. D.: *Critical Success Factors for Business Reengineering and Corporate Performance: The Case of Korean Corporations. Technological Forecasting and Social Change*, 58, 1998, str. 297–311.
20. Teng J., Fiedler K., Grover V.: *An exploratory study of the influence of the IS function and organizational context on business process reeninerin project initiatives*, 1998, Vol. 26, str. 679–698.

Fabris Peruško je diplomiral leta 1998 na Fakulteti za elektrotehniko in magistriral leta 2003 na Ekonomski fakulteti Univerze v Ljubljani. Zaposlen je v podjetju Halcom Informatika, d. o. o. iz Ljubljane na področju trženja aplikacij za elektronsko poslovanje in vodi Halcomovo hčerinsko podjetje v BiH EBB Electronic Banking Bureau, d. o. o. Sarajevo.

Vloga ogrodij pri razvoju sodobnih informacijskih rešitev

Andrej Krajnc, Marjan Heričko
IZUM, Prešernova 17, 2000 Maribor, andrej.krajnc@izum.si
FERI Maribor, Smetanova 17, 2000 Maribor, marjan.hericko@uni-mb.si

Povzetek

Prispevek opisuje uporabo in vlogo ogrodij pri razvoju sodobnih informacijskih rešitev. Prva uporabna ogrodja so bila narejena v osemdesetih letih. Največ prvih ogrodij je bilo uporabnih na nivoju grafičnega vmesnika, vse bolj pa se ogrodja uveljavljajo tudi na nivoju poslovnih komponent. Čeprav koncept ogrodij zelo spominja na druge koncepte ponovne uporabe, kot so komponente, knjižnice razredov in vzorci, obstajajo med njimi določene razlike. Ker so ogrodja uporabna na številnih področjih, obstaja veliko različnih ogrodij, ki jih je možno klasificirati na več načinov. Prispevek predlaga pristop k celoviti klasifikaciji ogrodij. Opredeljena je tudi vloga organizacijskih ogrodij, ki se od ostalih ogrodij razlikujejo predvsem po obsegu in osredotočenosti. Organizacijska ogrodja so veliko bolj kompleksna in poskušajo zaobjeti več vidikov, zato je tudi izgradnja takšnih ogrodij veliko bolj zahtevna. V prispevku so analizirani različni pristopi h gradnji ogrodij, poudarek pa je na definiranju primernega pristopa k razvoju organizacijskih ogrodij.
Ključne besede: objektno-orientirana programska ogrodja, klasifikacija, vzorci, komponente, ponovna uporaba, organizacijska ogrodja

Abstract

The Role of Frameworks in Developing Modern Information Solutions

The paper describes the role and usage of frameworks in developing modern information solutions. The first usable frameworks were developed in 1980's. Initially they meant to be focused on GUI development but nowadays frameworks can be found on the level of business components. Although the concept of frameworks is similar to the other reuse techniques, such as components, class libraries and patterns, they differ from each other. Because frameworks can be applied in many domains, there are many frameworks, which can be classified in different ways. The paper proposes a complete approach how to classify frameworks. The paper also describes enterprise frameworks, which are specific due to their scale and focus. Enterprise frameworks are much more complex and try to embrace many aspects and therefore their development is hard. The paper also analyzes different approaches for building frameworks. The approach how to build enterprise frameworks is described in detail.
Keywords: object-oriented software frameworks, classification, patterns, components, reuse, enterprise frameworks

1 UVOD

V zadnjih letih je pri razvoju programske opreme zelo narasla potreba po ponovni uporabi. Podjetja si ne morejo več privoščiti, da bi informacijske sisteme vedno znova gradila od začetka, saj je to predrago in nekonkurenčno. Napredek v nivoju ponovne uporabe je predstavljal uveljavitev objektne in komponentne tehnologije. Problem knjižnic razredov in komponent pa je v tem, da v večini primerov pri ponovni uporabi uporabimo le programsko kodo, nimamo pa pravih informacij o tem, kako, zakaj in na kakšne zahteve je bila zgrajena ta komponenta ipd. Da bi še izboljšali stopnjo ponovne uporabe informacijskih sistemov, je bil uveden koncept objektno orientiranih ogrodij. Ogrodja so večji gradniki kot komponente in jih nekateri uvrščajo med (že) na pol narejene aplikacije. Ogrodja so namenjena uporabi v več aplikacijah. V splošnem velja, da lahko z ogrodji še izboljšamo ponovno uporabo pri gradnji novih aplikacij [1, 2, 3].

Kljub temu, da so ogrodja prisotna že precej časa, še vedno velikokrat niso najbolje razumljena. Prispevek poskuša narediti korak naprej pri razumevanju ogrodij, predvsem pa želi jasno opredeliti koncept ogrodja in postaviti ločnico glede na ostale tehnike ponovne uporabe. Analizirali smo obstoječe klasifikacije [3, 4, 5, 6, 7], ki se večinoma osredinjajo na posamezne vidike, naš namen pa je bil oblikovati predlog celovite klasifikacije ogrodij. Posebna pozornost v prispevku je namenjena razumevanju organizacijskih ogrodij, ki so posebna vrsta ogrodij in se od ostalih razlikujejo predvsem po obsegu in namenu.

V nadaljevanju povzamemo zgodovinski razvoj ogrodij, v drugem poglavju so predstavljeni osnovni koncepti ogrodij ter primerjava ogrodij z drugimi tehnikami ponovne uporabe, v tretjem poglavju je podan

predlog celovite klasifikacije ogrodij, v četrtem pa so predstavljena organizacijska ogrodja in pristopi k njihovi izgradnji.

1.1 Zgodovinski razvoj ogrodij

Za prvo ogrodje, ki se je začelo uporabljati na več področjih, velja *Model-View-Controller* (MVC). MVC je objektno orientirano ogrodje za razvoj uporabniškega vmesnika. Razvito je bilo v začetku osemdesetih let in je uporabljeno v jeziku Smalltalk-80. Podjetje Apple Inc. je v tistem času razvilo ogrodje MacApp, ki je prav tako ogrodje za razvoj uporabniškega vmesnika, namenjeno pa je gradnji aplikacij za računalnike Macintosh.

Med pomembnejša ogrodja, razvita v devetdesetih, spadajo CommonPoint (množica ogrodij za hitrejši razvoj aplikacij), HotDraw (ogrodje za izgradnjo grafičnih urejevalnikov, napisano v jeziku Smalltalk), ACE (*ADAPTIVE Communication Environment* – objektno orientirano ogrodje, namenjeno za komunikacijsko programsko opremo), JAWS (*Adaptive Web Server* – spletni strežnik in ogrodje za izgradnjo drugih vrst strežnikov) in verjetno eno najbolj uporabljenih ogrodij za platformo Windows MFC (*Microsoft Foundation Classes*). Ogrodje MFC je bilo ob ogrodju OWL (*Object Windows Library*) kar nekaj časa de facto industrijski standard za razvoj grafičnih aplikacij na osebnih računalnikih. V devetdesetih so se pojavila številna nova ogrodja tudi na drugih področjih, kot so multimedija, operacijski sistemi in sistemi za kontrolo procesov [8].

Eno od največjih težav pri uporabi ogrodij je predstavljalo pomanjkanje standardov na področju ogrodij. Tega so se zavedali tudi v večini najpomembnejših podjetij, kar je prispevalo k temu, da veliko novejših ogrodij ni produkt zgolj enega podjetja, temveč pri njihovi izgradnji prek različnih delovnih skupin sodeluje več podjetij.

Med najpomembnejša ogrodja nove generacije prav gotovo spadajo programske arhitekture COM/DCOM (*Component Object Model/Distributed Component Object Model*), CORBA (*Common Object Request Broker Architecture*), EJB/RMI (*Enterprise JavaBeans/Remote Method Invocation*) in Microsoft .NET.

Opazen napredek je vzpodbudil tudi nastanek in uveljavitev programskega jezika java. Večina ogrodij za javo namreč nastaja znotraj delovnih skupin v procesu JCP (*Java Community Process*) [9], ki ga sicer upravlja podjetje Sun, vendar pa v njem poleg korporacije Microsoft sodelujejo skorajda vsa pomembnejša podjetja. Med najpomembnejša ogrodja, ki so v okviru procesa JCP, sodijo poleg že omenjenih EJB in RMI še AWT (*Abstract Window Toolkit*), Swing/JFC (*Java Foundation Classes*), Java Servlet, JSP (JavaServer Pages), JSF (JavaServer Faces), Collection Framework, JMF (*Java Media Framework*), JAF (*JavaBeans Activation Framework*), še veliko več pa jih je v izgradnji. Glede na to, da java postaja eden najpomembnejših programskih jezikov za razvoj aplikacij, lahko najdemo številna ogrodja za javo tudi zunaj procesa JCP. Najpomembnejše, sedaj že precej uveljavljeno in preverjeno ogrodje je prav gotovo IBM SanFrancisco (sedaj *IBM Business Components for WebSphere Application Server*), ki poleg splošnih poslovnih objektov vsebuje še domensko specifične poslovne objekte. Vse več je tudi ogrodij, ki temeljijo na odprti kodi. Primeri takšnih ogrodij so Struts, Avalon in JCorporate Expresso.

V zadnjem času je za uporabo ogrodij zelo pomembno tudi ogrodje Microsoft .NET. S pojavom številnih jezikov, ki jih definira to ogrodje, predvsem s pojavom jezika C#, je nastalo kar nekaj novih ogrodij. Eno od najbolj znanih je ogrodje Windows Forms, ki je namenjeno za gradnjo oken v okolju Windows in predstavlja alternativo zgoraj omenjenemu Swingu. Ostala pomembnejša ogrodja v Microsoft .NET so ASP.NET, ADO.NET, .NET Web Services in BizTalk.

Številna podjetja uporabljajo opisana ogrodja in ob tem gradijo še svoja. Ta ogrodja se uporabljajo večinoma zgolj interno za razvoj drugih produktov in niso namenjena prodaji.

Že iz opisa ugotovimo, da ogrodja naslavljajo različne domene in namene. Da bi lažje identificirali primerna nam ogrodja, je smiselno opredeliti nove kriterije in klasifikacije, ki olajšajo izbiro.

2 DEFINICIJE IN KONCEPTI

2.1 Definicija ogrodja

Kaj sploh so ogrodja? V praksi lahko naletimo na številne produkte, ki so označeni kot ogrodje (*framework*). So vsi ti produkti tudi v resnici ogrodja? Zdi se, da je beseda ogrodje modna beseda in jo je koristno prilepiti k opisu produkta. Tako so nekateri produkti velikokrat ogrodja zgolj na papirju. Vsaka knjižnica razredov namreč še ni ogrodje; podobno se tudi pojmovanje ogrodij razlikuje od pojmovanja komponent in vzorcev.

Najbolj znana definicija ogrodij je iz leta 1988, avtorja pa sta Ralph Johnson in Brian Foote [4]:

"Ogrodje je množica razredov, ki vključuje abstraktni načrt rešitve za družino povezanih problemov."

Po našem mnenju so eno najustreznejših definicij objektno orientiranega ogrodja podali avtorji knjige *Design Patterns* (1995) [10]:

"Ogrodje je množica sodelujočih razredov, ki sestavljajo ponovno uporabljen načrt za specifično vrsto programske opreme. Ogrodje določa arhitekturne smernice z razdelitvijo načrta v abstraktne razrede in z definiranjem njihovih odgovornosti in sodelovanj. Razvijalec prilagaja ogrodje za posamezno aplikacijo z uporabo podrazredov in povezovanjem primerkov razredov ogrodja."

Ogrodje torej ni konkretna aplikacija, temveč predstavlja skelet za aplikacije, ki bodo zgrajene z uporabo tega ogrodja. Določene so osnovne funkcije, ki so skupne za več aplikacij, specifične funkcije za določeno aplikacijo pa dopolnijo uporabniki ogrodja sami – torej razvijalci informacijskih rešitev.

2.2 Primerjava ogrodij z drugimi tehnikami ponovne uporabe

Ogrodja so v današnjem času praviloma objektno orientirana tehnika ponovne uporabe. Imajo lastnosti, ki so skupne vsem tehnikam ponovne uporabe. Vse tehnike neki problem razbijejo na manjše enote, ki so bolj razumljive, obvladljive in ponovno uporabne v drugih situacijah. Cilji ponovne uporabe naj bi bili med drugim večja produktivnost, zmanjšanje stroškov in boljša kakovost.

Čeprav se ogrodja v praksi uporabljajo že nekaj časa, še posebej med objektno orientiranimi razvijalci, jih mnogo ljudi še vedno ne razume najbolje in jih zato napačno uporablja. Velikokrat je ogrodje opredeljeno kot vrsta vzorcev ali zgolj kot posebna vrsta komponent oziroma knjižnic razredov.

2.2.1 Ogrodja in komponente

Obstaja več definicij komponent. Večina avtorjev komponente obravnava kot del programske opreme z natančno določenim vmesnikom, implementacija te komponente pa je za uporabnike skrita [11]. Podobno kot za komponente velja tudi za ogrodja: ta ograjujejo podatke in funkcionalnost z natančno definirano množico vmesnikov, ki infrastrukturi ogrodja daje stabilnost in robustnost.

Z integracijo množic abstraktnih razredov in definiranjem standardnih načinov za sodelovanje med primerki teh razredov ponujajo ogrodja ponovno uporabne komponente za aplikacije. Na splošno komponente niso popolnoma samostojne, saj so običajno odvisne od funkcionalnosti, ki jih ponujajo druge komponente v ogrodju. Zbirke teh komponent tvorijo delno implementacijo, tj. skelet aplikacije. Ta skelet lahko prilagajamo z uporabo dedovanja ali z uporabo primerkov ponovno uporabnih komponent iz ogrodja.

Ker proizvajalci prodajajo ogrodja kot produkte, lahko tudi ogrodja štejemo za komponente. V aplikacijo lahko vključimo več komponent in več ogrodij. Vendar so ogrodja bolj razširljiva kot komponente, definiran pa imajo tudi bolj kompleksen vmesnik. Razvijalci morajo te vmesnike spoznati, preden začnejo uporabljati ogrodja, ker je naknadno spoznavanje novega ogrodja precej težko. Dobra lastnost ogrodij je, da so zelo zmogljiva in jih lahko uporabljamo za skoraj vse vrste aplikacij. Dobro ogrodje lahko v veliki meri zmanjša trud, ki ga je treba vložiti v izgradnjo razširljivih aplikacij [12].

Ogrodja in komponente sta torej različni, a kljub temu komplementarni in sodelujoči tehnologiji.

2.2.2 Ogrodja in knjižnice razredov

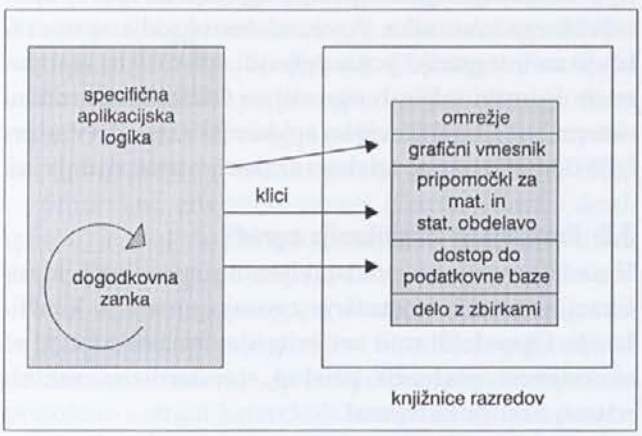
Ogrodja v praksi najpogosteje zamenjujemo s knjižnicami razredov, saj na prvi pogled med njimi ni večjih razlik. Tako kot knjižnice razredov vidijo razvijalci tudi ogrodja kot množico razredov, zato se pogosto zgodi, da uporabniki v resnici uporabljajo ogrodje, govorijo pa o "knjižnici razredov".

Ogrodja se od knjižnic razredov razlikujejo po tem, da ponujajo ponovno uporabo na višjem nivoju abstrakcije in granulacije. Pri knjižnicah razredov lahko v mnogih primerih ponovno uporabimo zgolj en razred, pri ogrodjih pa moramo najprej precej časa nameniti spoznavanju ogrodja, ponavadi spoznavanju cele množice medsebojno povezanih razredov. V večini primerov namreč nekega razreda iz ogrodja ne moremo ponovno uporabiti neodvisno od drugih. Velikokrat je knjižnica razredov ogrodje, če so med komponentami znotraj nje odvisnosti in če se programerji, ki jo spoznavajo, soočajo z veliko kompleksnostjo.

Knjižnice razredov ponujajo relativno majhno zrnatost (*granularity*) ponovne uporabe. Kot prikazuje slika 1, so npr. razredi tipično nizkonivojske, relativno neodvisne in splošne komponente, kot npr. pripomočki za matematično in statistično obdelavo, delo z

zbirkami, razredi za delo z omrežji, dostop do podatkovne baze. V nasprotju s tem komponente v ogrodju sodelujejo in tako ponujajo razširljiv arhitekturni skelet za družino povezanih aplikacij. Kot prikazuje slika 2, to zmanjšuje količino aplikacijsko specifične kode, saj je precej domensko specifičnega procesiranja že vključenega v splošne komponente ogrodja. Za razliko od knjižnic razredov za ogrodja velja, da definirajo napol gotove aplikacije, ki vključujejo domensko specifične objektne strukture in funkcionalnost. Lahko bi rekli, da imamo pri knjižnici razredov praviloma ponovno uporabo zgolj na nivoju programske kode, medtem ko pri ogrodjih ponovno uporabimo vsaj še zasnovo oz. načrt [13].

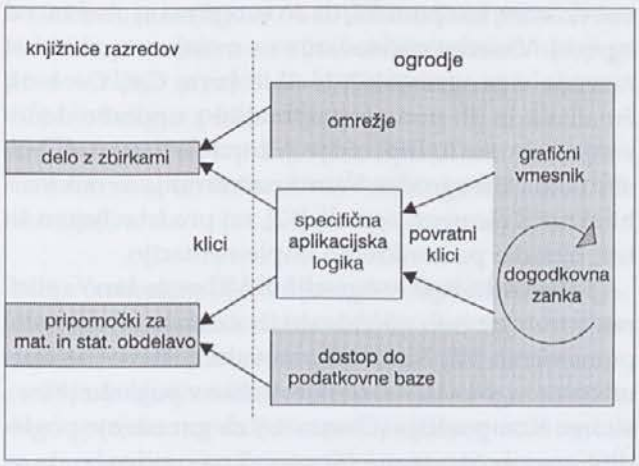
Obseg ponovne uporabe ogrodij je lahko precej večji kot pri uporabi tradicionalnih knjižnic razredov [1, 2, 3].



Slika 1: **Arhitektura knjižnice razredov**

Programiranje, ki temelji na uporabi ogrodij, zahteva nov način razmišljanja. Knjižnice razredov so večinoma "pasivne", kar pomeni, da nimajo glavne kontrole nad tokovi izvajanja aplikacije in izvajanje programa kontrolira koda določene aplikacije (slika 1). Za ogrodja pa velja, da so "aktivna", kar pomeni, da se pri izvajanju aplikacije običajno kontrola izvajanja prenese na ogrodje, ki potem po potrebi kliče programsko kodo za določeno aplikacijo. Takšno arhitekturo, ki temelji na povratnih klicih, prikazuje slika 2. Gre za obrat kontrole (*inversion of control*), ki je največkrat opisan kot "hollywoodsko načelo": ne kličite nas, mi bomo poklicali vas [14].

Tudi ogrodja in knjižice razredov sta komplementarni tehnologiji. Ogrodja pri izvajanju programske



Slika 2: **Arhitektura aplikacijskega ogrodja**

kode uporabljajo knjižnice razredov. Uporaba je lahko interna (kot del ogrodja) ali pa eksterna (prek povratnih, aplikacijsko specifičnih klicev).

2.2.3 Ogrodja in vzorci

Vzorci (*patterns*) so postali popularen način ponovne uporabe informacij o načrtovanju, kar še zlasti velja za objektno orientirano skupnost. Vzorec opisuje problem, rešitev in kontekst, v katerem rešitev deluje. Opisuje tudi tehnike uporabe, stroške in pridobitve. Razvijalci, ki si delijo množico vzorcev, imajo skupen slovar za opisovanje njihovih načrtov. Vzorci naj bi opisovali ponavljajoče se rešitve, ki so nadčasovne, torej vedno aktualne [12].

Ogrodja, ki so bila implementirana večkrat, predstavljajo neke vrste apliciranje določenega vzorca. Tako je npr. Bushmann [12] ogrodje Model-View-Controller opisal kot vzorec. Še več: aplikacije, ki uporabljajo ogrodja, se morajo prilagoditi načrtu ogrodja in modelu sodelovanja v tem ogrodju, tako da so ogrodja tudi razlog za uporabo vzorcev v aplikacijah. Vendar pa pri ogrodjih ne gre le za ideje, temveč tudi za programsko kodo. Če razvijalec razume ogrodje, mu ta koda ponuja način testiranja, primere za spoznavanje ogrodja ipd. Ponovna uporaba te kode omogoča hiter razvoj enostavne aplikacije, ki lahko, s tem ko razvijalec spoznava ogrodje, preraste v "pravo" aplikacijo.

Vzorci, opisani v knjigi *Design Patterns* [10], so z ogrodji tesno povezani na drug način. Ti vzorci so bili odkriti s preiskovanjem več ogrodij in izbrani za tipične primere ponovno uporabne objektno orientirane programske opreme. V ogrodju je običajno apliciranih

več vzorcev, kar pomeni, da so vzorci manj obsežni od ogrodij. Vzorcev načrtovanja ne moremo izraziti kot razrede v programskih jezikih java, C#, C++ ali Smalltalk in jih ponovno uporabiti z uporabo dedovanja oziroma kompozicije. Vzorci so torej tudi bolj abstraktni kot ogrodja. Vzorci načrtovanja so mikroarhitekturni elementi ogrodij [12], saj predstavljajo rešitev, ogrodja pa konkretno implementacijo.

Tako lahko npr. v ogrodju MVC zasledimo apliciranje treh glavnih načrtovalskih vzorcev in več manj pomembnih [12]. Vzorec Opazovalec (*Observer*) se uporablja za zagotovitev aktualnosti slike v pogledu (*View*), vzorec Kompozicija (*Composite*) za gnezdenje pogledov, vzorec Strategija (*Strategy*) pa omogoča, da se odgovornost za upravljanje z uporabniškimi dogodki v pogledu delegira na nadzornika (*Controller*). Podobno so številni načrtovalski vzorci iz knjige GoF [10] uporabljeni tudi v javanskih ogrodjih, ki jih je mogoče najti v razvojnem paketu JDK (*Java Development Kit*). Še zlasti veliko vzorcev lahko najdemo v ogrodju Swing.

Ogrodja so torej ena izmed tehnik ponovne uporabe. So bolj abstraktna in fleksibilna kot komponente, vendar bolj konkretna in lažja kot čisti načrt za ponovno uporabo. Čeprav bi jih lahko šteli za konkretno obliko vzorcev, pa ogrodja predstavljajo tehniko, pri kateri se hkrati ponovno uporabita načrt in programska koda, in so torej neke vrste aplikacijski generatorji oziroma predloge. Vzorci so ponazorjeni s programi, ogrodja pa so programi.

3 KLASIFIKACIJA OGRODIJ

3.1 Obstoječe klasifikacije ogrodij

Čeprav obstaja veliko objektno orientiranih ogrodij, ni bilo do sedaj veliko narejenega na področju klasifikacije ogrodij. V literaturi lahko najdemo le nekaj klasifikacij, pomembnejše so opisane v nadaljevanju.

Prvo klasifikacijo ogrodij sta naredila leta 1988 Johnson in Foote [4]. Ogrodja sta klasificirala glede na način, kako jih razširjamo. Tako sta definirala ogrodja v obliki bele škatle (*whitebox frameworks*) in ogrodja v obliki črne škatle (*blackbox frameworks*). Ogrodja v obliki bele škatle uporabljajo za razširjanje dedovanje in dinamično povezovanje, ogrodja v obliki črne škatle pa za razširjanje uporabljajo definiranje vmesnikov za komponente, ki jih lahko vključujemo v ogrodje z uporabo kompozicije. Fayad, Schmidt in Johnson [3] so leta 1999 razširili klasifikacijo z ogrodji v obliki sive škatle. Ogrodja v obliki sive škatle so načrtovana tako, da se izognemo slabostim, ki jih imajo ogrodja v obliki bele oz. črne škatle. Nekateri avtorji [5] so dodali še ogrodja v obliki steklene škatle, v katere je možen vpogled, spreminjati pa jih ne moremo.

Leta 1994 so v podjetju Taligent (zdaj IBM) ogrodja glede na področje delili v tri kategorije [6]: aplikacijska, domenska in podporna ogrodja. Aplikacijska ogrodja naj bi ponujala funkcionalnost, potrebno za različne aplikacije, npr. uporabniški vmesnik, dostop do baze, ipd. Domenska ogrodja so uporabljena le v eni domeni, npr. bančništvo, zavarovalništvo. Podporna ogrodja naslavljajo domene, kot so porazdeljeno računalništvo, dostop do datotek ipd.

Fayad in Schmidt [7] sta leta 1997 ogrodja glede na področje razdelila v sistemsko infrastrukturna, povezovalna in organizacijsko aplikacijska ogrodja. Sistemsko infrastrukturna ogrodja poenostavljajo razvoj prenosljive in učinkovite sistemske infrastrukture, to so npr. komunikacijska ogrodja, ogrodja za razvoj uporabniškega vmesnika. Povezovalna ogrodja se uporabljajo za integracijo porazdeljenih aplikacij in komponent. Primeri takšnih ogrodij so CORBA, sporočilni sistemi ipd. Organizacijsko aplikacijska ogrodja naslavljajo domene, kot so telekomunikacije, proizvodnja itd.

3.2 Predlagana klasifikacija ogrodij

V nadaljevanju bo predstavljen nov predlog h klasifikaciji ogrodij, ki razširja zgoraj omenjene klasifikacije. Opredelili smo več kriterijev klasifikacije, ki so razširljivost, področje, pristop, standardiziranost, zrnatost, licence in format.

3.2.1 Vrste ogrodij glede na razširljivost

Pri predlagani klasifikaciji glede na razširljivost so združene nekatere zgoraj opisane obstoječe klasifikacije. Tako imamo glede na razširljivost *ogrodja v obliki bele, črne, sive* in *steklene škatle.*

3.2.2 Vrste ogrodij glede na področje

V predlagani klasifikaciji smo kot osnovo vzeli klasifikacijo, ki sta jo predlagala Fayad in Schmidt, nato pa smo naredili nekaj sprememb. Organizacijsko aplikacijska ogrodja delimo v dve vrsti ogrodij: domenska in organizacijska.

Domenska ogrodja vsebujejo znanja o določeni domeni in ne pokrivajo nobenega drugega vidika.

Organizacijska ogrodja so posebna vrsta ogrodij. Od ostalih ogrodij se razlikujejo predvsem po obsegu in osredinjenosti. Glede na druga ogrodja so po obsegu

večja in bolj kompleksna, in za razliko od drugih poskušajo zaobjeti več vidikov, tako infrastrukturnega kot tudi domenskega, poseben poudarek pa je tudi na arhitekturi. Organizacijska ogrodja so podrobneje opisana v četrtem poglavju.

Kot dodatno smo dodali še eno vrsto ogrodij – poslovno-sodelovalna ogrodja. Ta ogrodja omogočajo elektronsko poslovanje in integracijo. Poskušajo premagati ovire tehnologije EDI (*Electronic Data Interchange*), ki je bila zelo draga in omejena na uporabo le v velikih podjetjih. Večina poslovno-sodelovalnih ogrodij temelji na uporabi jezika XML (*eXtensible Markup Language*). Primer takšnega ogrodja je ebXML (*Electronic Business using eXtensible Markup Language*).

Glede na področje smo definirali torej pet vrst ogrodij: *sistemsko infrastrukturna, povezovalna, domenska, organizacijska* in *poslovno sodelovalna ogrodja.*

3.2.3 Vrste ogrodij glede na pristop h gradnji

Z razvojem objektne tehnologije so se začeli uporabljati novi pristopi h gradnji ogrodij. Objektno orientirana analiza in načrtovanje sta vodila k objektnim ogrodjem (npr. MVC, MFC, Swing), s pojavom komponent pa so se pojavila komponentna ogrodja (npr. DCOM, EJB).

Storitveno orientiran razvoj (*Service-oriented development*) je naslednji korak v tem razvoju [15]. Medtem ko prejšnji pristopi temeljijo na uporabi vmesnikov objektov in komponent, pa storitveno orientiran razvoj temelji na uporabi storitev. Storitev je pogodbeno definirano obnašanje, ki je lahko implementirano in ponujeno s strani katerekoli komponente za uporabo v katerikoli komponenti. Pri klasičnih projektih tipa strežnik/odjemalec so implementacije odjemalca in strežnika v veliki meri odvisne ena od druge, implementacija storitve pa naj bi bila neodvisna od implementacije odjemalca. Primera storitveno orientiranih ogrodij sta Jini in Microsoft .NET.

Pred časom je bilo kot nova paradigma v razvoju programske opreme predstavljeno aspektno orientirano programiranje (AOP) [16]. AOP definira nov programski konstrukt, imenovan aspekt (*aspect*), ki ga uporabljamo za zajetje vseh tistih delov, ki se razpredajo po celotnem sistemu, in jih želimo povezati v ločene programske entitete. Tako komponente v aplikacijah ohranijo svojo odgovornost, obnašanje, ki je bilo prej razpredeno po celotnem sistemu, pa je sedaj omejeno le na en aspekt. Aspektno orientirana ogrodja podpirajo delitev komponent in aspektov na različnih nivojih. Gre za tridimenzionalni model, ki je sestavljen iz komponent, aspektov in nivojev. Komponente ponujajo osnovno funkcionalnost, nivoji pa predstavljajo aspekte in komponente, razdeljene v bolj obvladljive podprobleme.

Glede na pristop h gradnji ločimo torej *objektna, komponentna, storitveno orientirana* in *aspektno orientirana* ogrodja.

3.2.4 Vrste ogrodij glede na standardiziranost

Glede na standardiziranost ločimo *standardizirana, nestandardizirana* in *delno standardizirana ogrodja.*

Standardizirana ogrodja so tista, ki jih je kot standard sprejelo neodvisno mednarodno priznano telo za standardizacijo. Primer takšnega ogrodja je CORBA, ki ga je sprejelo združenje OMG (*Object Management Group*). Prednost standardiziranih ogrodij je, da ponujajo najrazličnejšim podjetjem skupna izhodišča, ki so v pomoč pri razvoju informacijskih sistemov. Verjetno pa je največja pomanjkljivost standardov, da se večinoma sprejemajo zelo dolgo.

Nestandardizirana ogrodja so tista, ki so bila največkrat narejena le v okviru enega samega podjetja in se običajno najhitreje odzovejo na spremembe na tržišču. Ena njihovih največjih pomanjkljivosti je, da so večinoma nestabilna. Primer takšnih ogrodij so Microsoftovi MFC, COM in DCOM in IBM SanFrancisco.

Delno standardizirana ogrodja so tista, ki nastanejo kot produkt neke delovne skupine, v katero je vključena večina najpomembnejših proizvajalcev programske opreme. Gre za kompromis med načinom dela različnih teles za standardizacijo in načinom dela podjetij, ki sama postavljajo standarde. Primer delno standariziranih ogrodij so javanska ogrodja podjetja Sun, sprejeta v okviru procesa JCP: EJB, Swing/JFC, Collection Framework, JAF, JSP, Java Servlet in JavaServer Faces.

3.2.5 Vrste ogrodij glede na zrnatost

Glede na zrnatost ločimo *drobnozrnata* in *grobozrnata ogrodja.*

Drobnozrnata ogrodja so v splošnem manjša z manj funkcionalnosti in kot takšna so lažja za implementacijo in ponovno uporabo. Grobozrnata ogrodja so v splošnem večja, sestavljena iz več komponent in vzorcev, imajo kompleksne vmesnike in so težja za implementacijo in ponovno uporabo.

3.2.6 Vrste ogrodij glede na licenco

V zadnjem času prosta programska oprema, še posebej odprtokodna programska oprema, vse bolj pridobiva

na pomenu. Vse več podjetij prepoznava prednosti uporabe proste programske opreme.

Glede na licenco lahko ogrodja razdelimo na *komercialna* in *nekomercialna* ogrodja.

Komercialna ogrodja so tista, za katera moramo kupiti licenco. Primeri takšnih ogrodij so implementacije modela CORBA ter IBM-ov SanFrancisco.

Nekomercialna ogrodja lahko uporabljamo brezplačno. Mednje spada večina ogrodij, sprejetih v okviru procesa JCP. Nastaja pa tudi vse več ogrodij znotraj različnih projektov za odprto kodo (*open source*). Najbolj znana tovrstna ogrodja so JUnit, ogrodja Avalon, Struts in Cocoon v okviru Apachevega projekta Jakarta ter Expresso podjetja JCorporate.

3.2.7 Vrste ogrodij glede na format

Butler Group je predlagal klasifikacijo komponent glede na format [17]. Mi pa smo razširili to klasifikacijo, da je uporabna za ogrodja.

V vsaki fazi razvojnega cikla lahko obstaja ogrodje v različnih oblikah. Lahko obstaja kot *logična specifikacija, fizični načrt, izvorna koda* ali *binarna oz. izvršljiva koda*. Lahko je uporabljena oz. ponovno uporabljena v kateremkoli od teh formatov, seveda odvisno od cilja, ki ga želimo doseči.

Logična specifikacija predstavlja celovit opis tega, kar zagotavljajo storitve ogrodja skupaj s kakršnimikoli omejitvami glede na to, kako naj bi ogrodja te storitve zagotavljale. Primer takšnih specifikacij so specifikacije ebXML.

Fizični načrt vključuje popolne specifikacije razredov, vmesnikov in metod, ki definirajo ogrodje. Neko podjetje lahko naredi fizični načrt, medtem ko drugo podjetje naredi implementacijo tega načrta. Primer takšnega fizičnega načrta je specifikacija za Java Servlet, ki je definirana v procesu JCP, implementirana pa s strani fundacije Apache v produktu Tomcat.

Večina ogrodij je dostavljenih kot izvršljiva koda, kar je tudi format za večino komercialnih ogrodij.

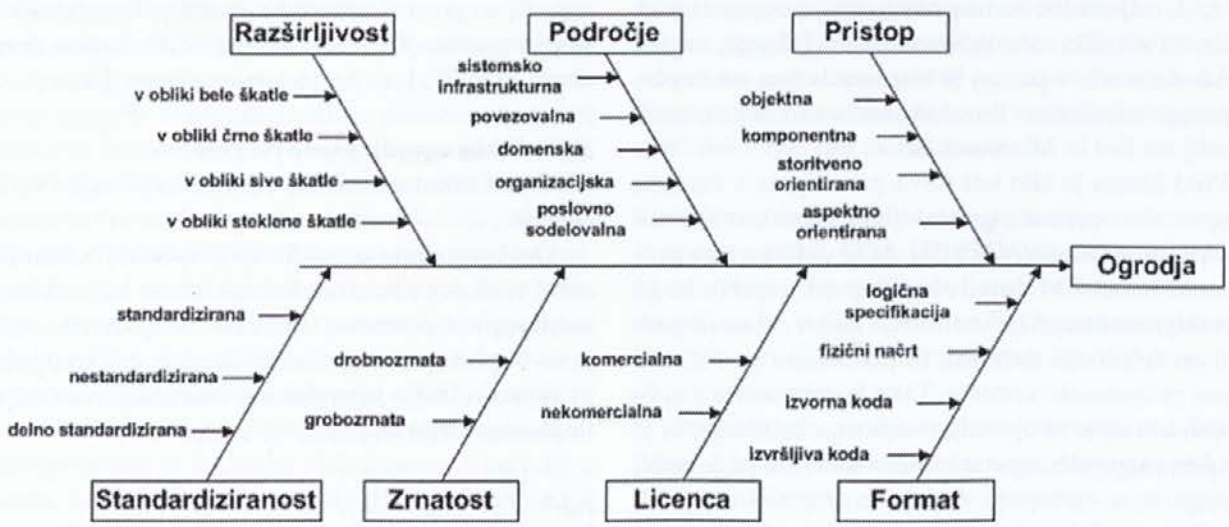
Nekatera ogrodja (npr. odprtokodna ogrodja) so dostavljena skupaj z izvorno kodo.

Slika 3 prikazuje predstavljeno klasifikacijo ogrodij. Poudariti velja, da pri klasifikaciji ne gre za medsebojno izključevanje kategorij, temveč se predvsem odražajo različne dimenzije ogrodij.

Da lahko neko ogrodje uvrstimo v več kategorij, dokazujejo organizacijska ogrodja. Organizacijska ogrodja ne zaobjemajo le enega, temveč več vidikov, zaradi česar je tudi razvoj takšnih ogrodij nekoliko drugačen od razvoja klasičnih ogrodij. Naslednje poglavje podrobneje predstavlja organizacijska ogrodja in pristop k njihovi gradnji.

4 Organizacijska ogrodja

Organizacijska ogrodja (*Enterprise Frameworks*) so posebna vrsta ogrodij. Od ostalih ogrodij se razlikujejo predvsem po obsegu in osredotočenosti. Če jih primerjamo z ostalimi ogrodji, so organizacijska ogrodja po obsegu večja in bolj kompleksna, v njih pa so lahko vsebovani najrazličnejši objekti, komponente in



Slika 3: **Predlog celovite klasifikacije ogrodij**

tudi druga ogrodja. Kar se osredotočenosti tiče, ostala ogrodja večinoma pokrivajo le en poseben vidik, bodisi domenski vidik (npr. finančne transakcije v spletni trgovini), infrastrukturni vidik (npr. porazdeljenost in trajnost objektov) ipd. Posebnost organizacijskih ogrodij je, da poskušajo zaobjeti več vidikov, tako infrastrukturnega kot tudi domenskega, poseben poudarek pa je tudi na arhitekturi [18, 19]. Veliko organizacijskih ogrodij temelji na vzorcu organizacijsko komponentno ogrodje.

4.1 Vzorec organizacijsko komponentno ogrodje

V zadnjem času sta eni od najpomembnejših programskih arhitektur J2EE in .NET. Kljub določenim razlikam lahko med njima najdemo tudi precej podobnosti. Ena od njih je ta, da obe arhitekturi temeljita na skupnem arhitekturnem vzorcu, imenovanem organizacijsko komponentno ogrodje (*Enterprise Component Framework* oz. ECF) [20].

Vzorec ECF opisuje osnovne mehanizme za uporabo komponent v porazdeljenem okolju, predvsem storitve za medprocesno komunikacijo, transakcije in trajnost.

Vzorec vsebuje sedem vlog, ki medsebojno sodelujejo: *Odjemalec*, *Tovarniški namestnik*, *Oddaljeni namestnik*, *Kontekst*, *Komponenta*, *Vsebnik* in *Trajnostna storitev* (slika 4).

Odjemalec (*Client*) je katerakoli entiteta, ki zahteva storitve od komponente (*Component*) v ogrodju. Uporabnik ne komunicira neposredno s komponento, temveč prek dveh posrednikov, tovarniškega namestnika (*Factory Proxy*) in oddaljenega namestnika (*Remote Proxy*), ki delegirata uporabnikove zahteve h komponenti. Ta koncept posrednih klicov omogoča podporo naslednjima zelo pomembnima funkcijama:

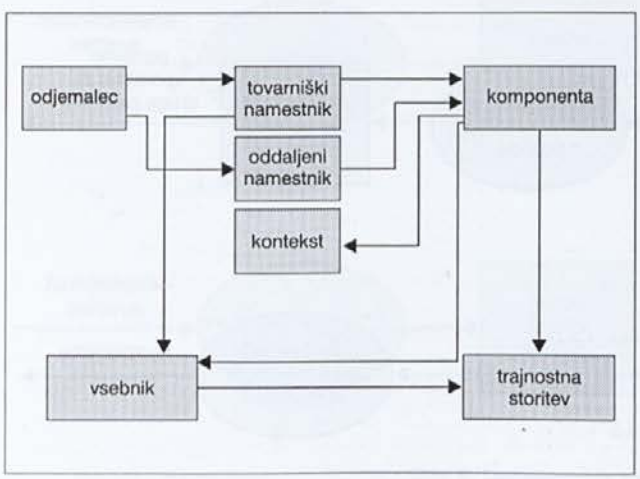
- *transparentnost lokacij*: na nižjem nivoju abstrakcije so lahko posredniki realizirani z uporabo koncepta *stub-skeleton*, ki se uporablja pri arhitekturah CORBA, Java RMI in COM+.
- *prestrezanje sporočil*: uporaba posrednikov omogoča, da okolje, v katerem se uporabljajo komponente, prestreza klice metod in zagotavlja storitve, ki temeljijo na množici atributov, ki so definirani v času izvajanja, in deklarativno opredeljujejo zahteve glede varnosti, transakcij itd.

Posrednika tovarniški namestnik in oddaljeni namestnik sta v bistvu dve izpeljanki vzorca namestnik (*Proxy*) [10] in kot taka predstavljata neke vrste podvzorec v vzorcu ECF. Tovarniški namestnik izvaja operacije za ustvarjanje objektov (npr. operaciji *create* in *find*), oddaljeni namestnik pa izvaja poslovne operacije, specifične za komponente (npr. metode *set* in *get* za lastnosti).

Komponente in oba posrednika se nahajata v vsebniku (*Container*). Vsebnik predstavlja izvajalno okolje za ogrodje, saj omogoča različne porazdeljene storitve, kot so medprocesna komunikacija, varnost, transakcije in trajnost. Vsebnik vsebuje tudi entiteto kontekst (*Context*), ki za vsako komponento vzdržuje specifične kontekstne informacije, kot so stanja transakcij, podatke o trajnosti in varnosti.

Trajnostne storitve (*Persistence Service*) za komponente (npr. shranjevanje in črpanje informacij iz podatkovne baze) se lahko koordinirajo s komponento samo, koordiniranje pa se lahko tudi delegira k vsebniku.

Vzorec ECF lahko ob J2EE in .NET uporabimo pri izgradnji drugih arhitektur. Vzorec ECF namreč vsebuje koncepte, ki so skupni mnogim poslovnim procesom.



Slika 4: **Vzorec ECF**

4.2 Pristop k razvoju ogrodij

Razvojni cikel programskih produktov je tipično sestavljen iz faz, kot so zajemanje zahtev, analiza, načrtovanje, implementacija, itd. Organizacijska ogrodja so svojstveni produkti, saj pokrivajo ostale programske produkte (komponente sistema) in tako je razvojni cikel organizacijskega ogrodja prepleten z razvojnimi cikli ostalih produktov. Najbolje bi bilo, če bi nam uspelo ločiti organizacijsko ogrodje in njegov razvojni cikel od drufih produktov in njihovih razvojnih ciklov. Gledano na široko je organizacijsko ogrodje skupek domensko specifičnih vidikov (domenska funkcionalnost) in domensko neodvisnih vidikov.

Najbolje bi bilo, če bi lahko te vidike obravnavali ločeno karseda dolgo in jih povezali šele zelo pozno. Vendar pa je ločevanje teh vidikov precej težko in prav to predstavlja enega ključnih faktorjev za učinkovito ponovno uporabo organizacijskih ogrodij. Razvoj organizacijskih ogrodij temelji na poznanih pristopih k razvoju ogrodij, ki so predstavljeni v nadaljevanju.

Za razvoj ogrodij se uporabljajo različni načini, žal pa še vedno nimamo splošno sprejetega in uveljavljenega načina razvoja ogrodij. V splošnem lahko rečemo, da se pojavljata dva pristopa [19]:

- analitični pristop (*top-down*): ogrodje razvijamo z izvajanjem procesa domenskega inženirstva, pri čemer začnemo z analizo domene, načrtovanjem, realizacijo itd.,
- sintetični pristop (*bottom-up*): začnemo z razvojem aplikacije znotraj domene ogrodja in potem vpeljujemo točke variabilnosti.

4.2.1 Ogrodja kot produkt domenskega inženirstva

Domensko inženirstvo je množica aktivnosti, katerih namen je razviti ponovno uporabne produkte iz realnega primera v funkcionalni domeni. Različne metode domenskega inženirstva imajo lahko različne rezultate ali predlagajo različne hevristike, vendar se bolj ali manj vse opirajo na te aktivnosti:

- izvajanje neke vrste analize različnosti oz. podobnosti z namenom, da identificiramo tiste vidike, ki so skupni za vse aplikacije znotraj domene in da identificiramo tiste vidike, ki ločijo neko aplikacijo od druge aplikacije,
- izpeljava vedno bolj konkretnih opisov za te vidike, pri čemer začnemo z opisi na nivoju analize, potem pa se spuščamo vse niže do opisov programske kode.

Proces domenskega inženirstva se izvaja inkrementalno, saj gre za kompleksne postopke, ki lahko trajajo precej časa. Prav tako je zaželeno, da se ogrodje oz. arhitektura domene testira dokaj zgodaj, preden gremo v širšo implementacijo komponent.

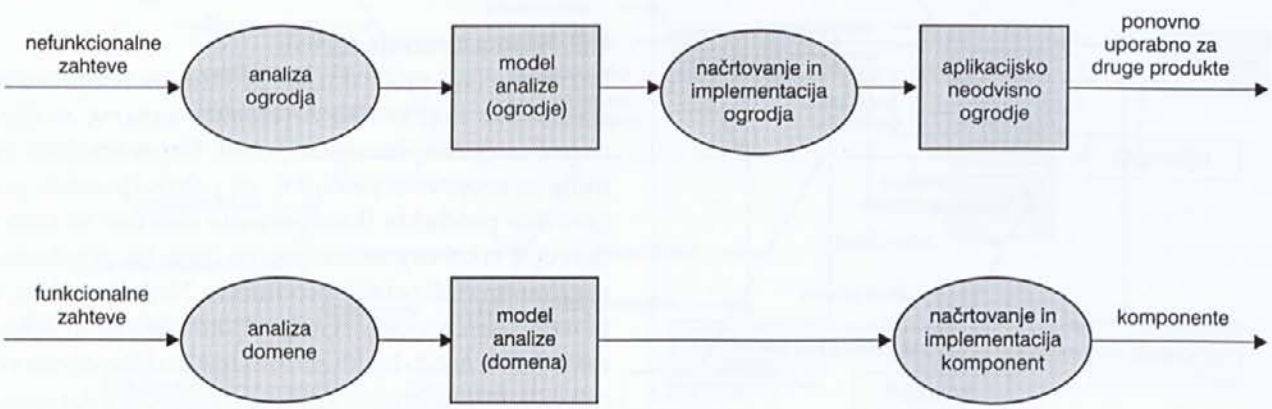
Katere vidike naj obravnavamo najprej? Splošno mnenje je, da moramo začeti s tistimi vidiki, ki imajo največji vpliv na arhitekturo. Tako naj bi se v prvem inkrementu ukvarjali s tistimi funkcijami, ki za delovanje zahtevajo večji del ali celotno infrastrukturo, kasneje bi dodajali nove funkcije, vendar le-te ne bi smele zahtevati nove infrastrukture.

Če npr. želimo razviti ogrodje za spletno bančništvo, potem bodo z vidika infrastrukture pomembni naslednji vidiki poslovnih aplikacij:

- porazdeljenost: aplikacijska logika naj bi bila lokacijsko transparentna, kar pomeni, da moramo uporabljati porazdeljeno infrastrukturo,
- varnost: komunikacija prek mreže mora biti varna, kar zahteva upravljanje z uporabniki, avtorizacijami, certifikati, enkripcijo ipd.,
- transakcijske storitve: uporabljati moramo transakcijske monitorje, ki bodo podpirali porazdeljene transakcije.

Ker so vsi ti vidiki povezani, mora, če želimo razviti ogrodje za spletno bančništvo, prvi inkrement vsebovati najmanjšo množico funkcij, ki za delovanje zahtevajo implementacijo zgoraj omenjenih vidikov.

Slika 5 prikazuje implementacijo aplikacijsko neodvisnega ogrodja, ki je lahko ponovno uporabna. Aplikacijsko neodvisno ogrodje je bilo implementirano



Slika 5: **Implementacija aplikacijsko neodvisnega ogrodja**

pred načrtovanjem in implementacijo domenskih komponent.

4.2.2 Ogrodja kot produkt razvoja aplikacij

Pri tem pristopu razvijamo ogrodja iz podmnožic aplikacij v domeni ogrodja. Tudi tukaj se ogrodje razvija inkrementalno in vzporedno z razvojem aplikacij.

Proces se prične z identifikacijo domene ogrodja in z identifikacijo zahtev za prvo aplikacijo. Implementacija aplikacije se kasneje razvije v prvi inkrement ogrodja, kjer implementiramo prve točke variabilnosti (t.i. *hotspot*). Točke variabilnosti so tisti vidiki, ki se pogosto spreminjajo od ene aplikacije do druge. Potem, ko smo končali z razvojem prve iteracije ogrodja, začnemo z razvojem naslednje druge aplikacije, čemur sledi druga iteracija razvoja ogrodja z novimi točkami variabilnost, sledi tretja aplikacija, pa tretja iteracija ogrodja itd.

V tipičnem inkrementu se konkretni razred (točka variabilnosti) iz iteracije N v iteraciji N+1 zamenja z abstraktnim razredom ali vmesnikom in konkretnimi podrazredi.

V določenih primerih lahko točke variabilnosti obravnavamo tudi z uporabo parametrov.

Slika 6 prikazuje ogrodje, ki je bilo implementirano hkrati z implementacijo domenskih komponent. V tem primeru je končno ogrodje precej v manjši meri ponovno uporabno in je specifično zgolj za to aplikacijo oz. morebiti še za kakšne aplikacije iz te domene.

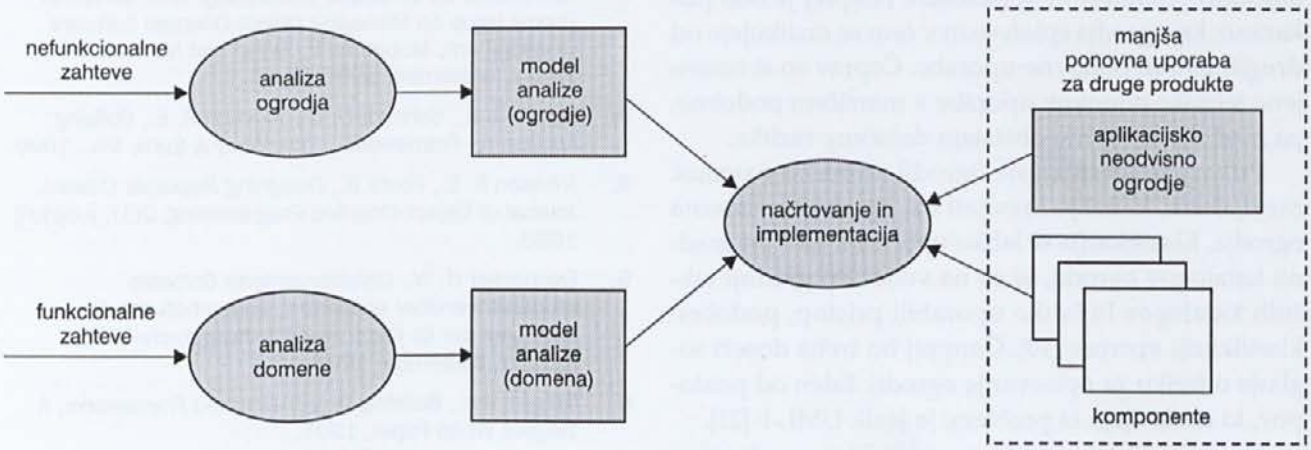
4.3 Razvoj organizacijskih ogrodij

Kot že prej omenjeno, so organizacijska ogrodja posebna vrsta ogrodij. Tako tudi zanje velja, da jih je mogoče razvijati s katerim od dveh zgoraj opisanih pristopov (analitični oz. sintetični pristop). Kateri je primernejši za razvoj organizacijskih ogrodij, če sploh kateri?

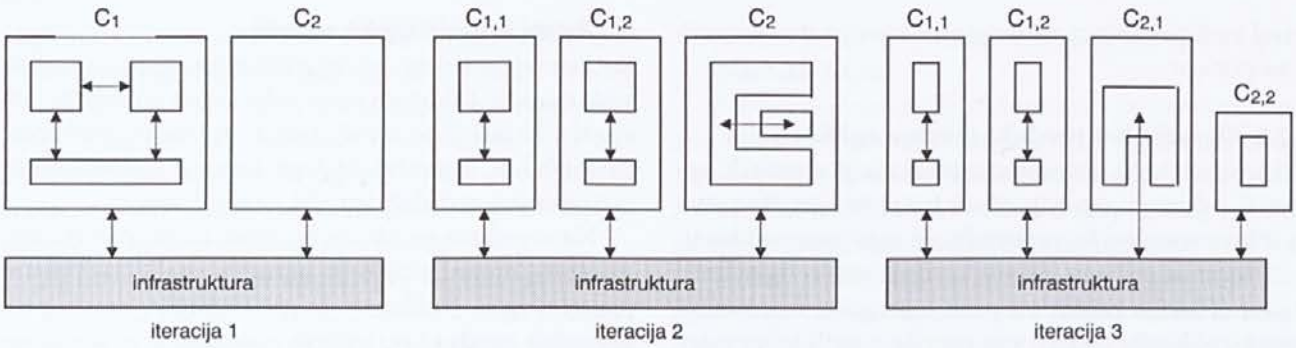
Na eni strani se zdi, da je zaradi truda, potrebnega za razvoj organizacijskega ogrodja, analitičen pristop precej tvegan. Veliko truda gre namreč v razvoj programskih produktov, katerih uporabnost in stroškovna učinkovitost ni zagotovljena. Samo dejanski projekti, ki uporabljajo ogrodje, lahko validirajo arhitekturo in komponente. Na drugi strani pa se zdi potratno priti do arhitekture skozi iteracije, saj je potrebno kar nekaj časa, preden se arhitektura izoblikuje. Sprememba arhitekture v poznih iteracijah je lahko zelo problematična, saj ima lahko to velik vpliv na že delujoče aplikacije.

Verjetno je najboljši hibridni pristop [19], ki združuje lastnosti obeh, tako analitičnega kot tudi sintetičnega pristopa. Arhitekturni vidiki ogrodja bi morali biti implementirani s centraliziranim analitičnim pristopom, pri čemer začnemo z načrtovanjem in nadaljujemo vse do implementacije. Posebno pozornost moramo posvetiti ločitvi infrastrukturnih vidikov od funkcijskih vidikov. Funkcijski vidiki arhitekture bodo razviti v iteracijskem in inkrementalnem načinu. Slika 7 prikazuje ta hibridni pristop.

Infrastruktura je razvita v prvi iteraciji, kjer se izvede omenjeni analitični pristop, začenši z zahtevami, sledi analiza, načrtovanje in dejanska implementacija



Slika 6: **Implementacija aplikacijsko specifičnega ogrodja**



Slika 7: **Življenjski cikel organizacijskih ogrodij**

infrastrukture. Kot dodatek k infrastrukturi implementiramo določene aplikacijske komponente, ki so na začetku še dokaj grobe in nedodelane, šele v naslednjih iteracijah pa se pravilno izoblikujejo, prečistijo in po potrebi tudi ločijo. Če na sliki 7 pogledamo komponento C_1, vidimo, da ima le-ta svojo interno strukturo in svoje interakcijske mehanizme. V drugi iteraciji se komponenta preoblikuje in tako podkomponenti $C_{1,1}$ in $C_{1,2}$ sedaj komunicirata prek skupne infrastrukture. Na primer, komponenta C_1 bi lahko bil obstoječi sistem, ki imajo svojo "lokalno" arhitekturo. V prvi iteraciji naredimo most (*bridge*) za uporabo tega obstoječega sistema, v naslednjih iteracijah pa preoblikujemo podkomponente obstoječega sistema, pri čemer večamo fleksibilnost infrastrukture, predvsem pa podpiramo izmenljivost komponent.

5 SKLEP

V prispevku je bila predstavljena uporaba ogrodij v objektno orientiranih aplikacijah. Najprej je bilo prikazano, kaj ogrodja sploh so in v čem se razlikujejo od drugih tehnik ponovne uporabe. Čeprav so si omenjene tehnike ponovne uporabe v marsičem podobne, pa med njimi vseeno obstajajo določene razlike.

Prikazane klasifikacije ogrodij so lahko v pomoč razvijalcem, ki želijo razvijati oz. uporabiti ustrezna ogrodja. Klasifikacija se lahko uporablja tudi pri gradnji katalogov ogrodij, ki so na voljo. Pri gradnji takšnih katalogov bi lahko uporabili pristop, podoben klasifikaciji vzorcev [10]. Čimprej bo treba doseči soglasje o jeziku za opisovanje ogrodij. Eden od pristopov, ki naslavljajo ta problem, je jezik UML-F [21].

Organizacijska ogrodja so arhitekturno orientirana ogrodja, ki poleg infrastrukture pokrivajo še del funkcionalnih zahtev aplikacij znotraj neke domene. Zaradi velikosti in kompleksnosti organizacijskih ogrodij je potrebna posebna pazljivost pri njihovem razvoju. Obstaja več pristopov k razvoju ogrodij, vendar žal še nobeden od njih ni splošno sprejet in uveljavljen.

Ogrodja so že in zagotovo bodo tudi v prihodnosti eno od najpomembnejših področij v razvoju informacijskih sistemov. Podjetja se vedno pogosteje odločajo za uporabo ogrodij. Ogrodje je zelo težko zgraditi, vendar lahko na to gledamo kot na strateško investicijo, ki se bo poplačala z uporabo ogrodja pri več aplikacijah.

6 LITERATURA

1. Morisio M., Romano D., Stamelos I., *Quality, Productivity and Learning in Framework-Based Development: an Exploratory Case Study*, IEEE Transactions on Software Engineering, vol 28, n.8, avgust 2002, pp. 340-357,

   http://morserv.polito.it/papers/tse-fwk-200_27.pdf.

2. Moser S., Nierstrasz O., *The effect of object-oriented frameworks on developer productivity*. IEEE Computer Theme Issue on Managing Object-Oriented Software Development, Mohamed E. Fayad and Marshall Cline, editors, september 1996:45-51.

3. Fayad M. E., Schmidt D. C., Johnson R. E., *Building Application Frameworks*, John Wiley & Sons, Inc., 1999.

4. Johnson R. E., Foote B., *Designing Reusable Classes*. Journal of Object-Oriented Programming, 2(1), junij/julij 1988.

5. Eisenecker U. W., *Objektorientierte Software wiederverwendbar entwerfen*, Sonderheft mit den Beiträgen der GI-Fachtagung Softwaretechnik 96, Koblenz, september 1996.

6. Taligent Inc., *Building Object-Oriented Frameworks*, A Taligent White Paper, 1994.

7. Fayad M. E., Schmidt D. C., *Object-Oriented Application Frameworks*, Communications of the ACM, Vol. 40, No. 10, oktober 1997.

8. Mattsson M., *Object-oriented Frameworks - A survey of methodological issues*, Department of Computer Science, Lund University, CODEN: LUTEDX/(TECS-3066)/ 1-130/(1996), http://www.ipd.hk-r.se/michaelm/thesis/index.html.
9. Sun, *The Java Community Process Program*, http://java.sun.com/aboutJava/communityprocess/.
10. Gamma E., Helm R., Johnson R., Vlissades J., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley, 1995.
11. Krajnc Andrej, *Komponentni razvoj informacijskih sistemov*, FERI Maribor, marec 1999.
12. Johnson R. E., *Frameworks = Components + Patterns*, Communications of the ACM Special Issue on OO Application Frameworks, ACM, Vol. 40, No. 10, oktober 1997.
13. Landin N., Niklasson A., *Development of Object-Oriented Frameworks*, CODEN:LUTEDX(TETS-5231)/1-146, Department of Communication Systems, Lund University, 1995, http://www.tts.lth.se/Personal/bjornr/Papers/OOFW.ps.
14. Schmidt D. C., *Applying Design Patterns and Frameworks to Develop Object-Oriented Communication Software*, 1997, http://www.cs.wustl.edu/čschmidt/PDF/HPL.pdf.
15. Bieber G., Carpenter J., *Introduction to Service-Oriented Programming (Rev 2.1)*, http://www.openwings.org/download/specs/ ServiceOrientedIntroduction.pdf.
16. Kiczales G., Lamping J., Mendhekar A., Maeda C., Lopes C., Loingtier J-M., Irwin J., *Aspect-Oriented Programming*. In Proceedings of ECOOP'97, Volume 1241 of LNCS. Pages 220—242. Springer Verlag, junij 1997.
17. *CBD Reference Model Part 2, What are Components? - Concepts and Classification, Version 0.95*, Butler Group, marec 1998.
18. Mili H., Fayad M., Brugali D., Hamu D. S., Dori D., *Enterprise frameworks: issues and research directions*. Software - Practice and Experience Volume 32, Number 8: 801-831 julij 2002.
19. Mili H., Mili A., *Lifecycles for Enterprise Frameworks*, Enterprise Frameworks: "Adequacies and Inadequacies", OOPSLA 2000 Workshop Submission.
20. Kobryn C., *Modeling components and frameworks with UML*, Communications of the ACM, Volume 43, Issue 10 (October 2000), 31—38.
21. Fontoura M., Pree W., Rumpe B., *The UML Profile of Framework Architectures*, Addison-Wesley, 2000.

▪

Andrej Krajnc je študent magistrskega študija na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru. Njegovo raziskovalno delo obsega objektno tehnologijo, komponentni razvoj, ponovno uporabo, ogrodja, vzorce, standarde družine XML in tehnologije medorganizacijskega povezovanja. Diplomiral je na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru.

▪

Marjan Heričko je izredni profesor na Inštitutu za informatiko na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegovo raziskovalno-razvojno delo obsega vse vidike objektne tehnologije in komponentnega razvoja, s poudarkom na metodologijah razvoja, metrikah in razvojnih okoljih. Svoje izkušnje je predstavil v številnih prispevkih na domačih in tujih konferencah ter v revijah. Je tehnični koordinator aktivnosti Centra za objektno tehnologijo ter predsednik konference OTS Objektna tehnologija v Sloveniji. Diplomiral, magistriral in doktoriral je na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru.

▪

Poslovno modeliranje v teoriji in praksi: izkušnje in napotki

Aleš Popovič, Mojca Indihar Štemberger, Jurij Jaklič, Andrej Kovačič
Inštitut za poslovno informatiko, Ekonomska fakulteta
ales.popovic@uni-lj.si, mocja.stemberger@uni-lj.si, jurij.jaklic@uni-lj.si, andrej.kovacic@uni-lj.si

Povzetek

Modeliranje poslovnih procesov je uporabno na mnogih področjih, še najbolj pa na področju prenove in informatizacije poslovanja ter pri strateškem načrtovanju informatike. Projektov prenove poslovnih procesov se danes loteva vse več organizacij, tudi slovenskih. V okviru teh projektov je treba izdelati modele poslovnih procesov, jih potem analizirati in na podlagi analize predlagati spremembe pri izvajanju procesov, pri organizaciji in predlagati njihovo informatizacijo. Za modeliranje procesov so na voljo številne tehnike in orodja, pri čemer je izbira tehnike/orodja odvisna od znanja informatika in namena modeliranja. Namen prispevka je predstaviti izkušnje avtorjev pri poslovnem modeliranju in orisati najpogostejše tehnike in nekatera orodja, ki so na voljo. V prispevku predstavljamo tudi primer poslovnega modeliranja na enem izmed slovenskih ministrstev.

Ključne besede: modeliranje poslovnih procesov, orodja za modeliranje, tehnike modeliranja, simulacije, prenova poslovnih procesov

Abstract

Business Process Modelling in Theory and Praxis: Experiences and Hints

Business process modelling is very useful in many areas, especially in business process reengineering and information systems renovation projects. Today more and more organisations have started such projects to meet their goals in a dynamic and demanding business environment. Such projects consist of developing and analyzing business process models and providing proposals in process enactment and organisational changes. Many different methods and techniques can be used for modelling business processes in order to give an understanding of possible scenarios for improvement. The main goal of the paper is to present authors' experience with business process modelling along with most frequent techniques and tools available. The example of business process modelling in one of Slovenian ministries is also presented.

Keywords: business process modelling, modelling tools, modelling techniques, simulation, business process reengineering

1 UVOD

Poslovni proces v organizaciji (podjetju) sestavljajo aktivnosti, kjer je s strukturo opisano njihovo logično zaporedje in medsebojna odvisnost in katerih namen je doseganje želenega rezultata [1]. Modeliranje poslovnih procesov omogoča enotno razumevanje in analizo poslovnih procesov, kar je osnova za temeljito razumevanje procesa. Skozi poslovne procese je možno analizirati in povezati organizacijo.

Pri organiziranju poslovanja organizacije je danes vse večji poudarek na procesih z dodano vrednostjo in manj na funkcijski hierarhiji. To je tudi eden izmed pomembnejših vzrokov za vse večjo priljubljenost modeliranja poslovnih procesov. Strokovnjaki s področja informacijskih tehnologij in poslovnega inženirstva so si enotni, da je začetek uspeha sistema odvisen od razumevanja poslovnih procesov v organizaciji [4]. Aquilar-Saven [1] ugotavlja, da predstavljajo poslovni procesi ključni dejavnik pri povezovanju organizacije. Avtorica nadalje ugotavlja, da je konceptualno modeliranje poslovnih procesov široko uporabljeno, saj omogoča razvoj programskih rešitev za podporo poslovnim procesom in analizo, prenovo oz. izboljšavo le-teh.

Čeprav je Levitt že v šestdesetih letih prvi izpostavil pomembnost obvladovanja poslovnih procesov, so le-ti postali ključnega pomena pri načrtovanju podjetja šele v zadnjem desetletju [18]. Davenport [5], Hammer [10] in Harrington [12] so le nekateri izmed avtorjev, ki so sodelovali pri uveljavljanju tega novega vidika. Vse večja priljubljenost po usmeritvah v poslovne procese [11] je botrovala hitremu porastu metodologij, modelirnih tehnik in orodij, ki jih podpirajo.

Prispevek predstavlja izkušnje avtorjev pri poslovnem modeliranju in simulaciji poslovnih procesov, ki smo si jih avtorji pridobili ob projektih prenove poslovanja in pregled nekaterih najpogostejših tehnik in orodij za modeliranje. Opozarja tudi na številne probleme, ki se pri tem pojavljajo.

2 MODELIRANJE POSLOVNIH PROCESOV

Preden se lotimo obravnave poslovnega modeliranja, skušajmo podrobneje opredeliti poslovni proces. Davenport [5] opredeljuje procese kot strukturirane, merljive sklope aktivnosti, katerih cilj je ustvariti določen proizvod ali storitev za kupca oz. trg. Hammer in Champy [11] opredeljujeta poslovni proces kot zbirko aktivnosti, ki enega ali več vhodov pretvorijo v izhod z dodano vrednostjo za kupca. Številni avtorji obravnavajo procese in poslovne procese kot sinonima. V našem prispevku povezujemo poslovne procese z organizacijo, saj določajo poti za dosego zastavljenih ciljev organizacije in so tako podmnožica množice drugih procesov. Obstajajo še številne druge opredelitve (poslovnih) procesov, katerih bistvo je skupno: (poslovne) procese sestavljajo odnosi med vhodi in izhodi, pri čemer se vhodi prek vrste aktivnosti, ki vhodom dodajo vrednost, pretvorijo v izhode.

Modeliranje poslovnih procesov je uporabno na mnogo področjih, največ pa na področju prenove in informatizacije poslovanja ter pri strateškem načrtovanju informatike. Projekti prenove poslovanja potekajo v grobem tako, da se najprej identificira poslovne procese in izdela modele obstoječega stanja (angl. »AS-IS« modele), ki se jih potem analizira in na podlagi ugotovitev analize predlaga spremembe v izvajanju poslovnih procesov (angl. »TO-BE« modele), njihovo informatizacijo ter organizacijske spremembe [23] (slika 1).

Naše izkušnje pri projektih prenove poslovanja kažejo, da je izdelava kvalitetnih modelov poslovnih procesov lahko zelo zahtevna, saj v praksi število različnih modelov na različnih nivojih hitro doseže nekaj sto. Ker so nekateri procesi zelo kompleksni in imajo veliko izvajalcev, se velikokrat zgodi, da je izdelava njihovih modelov zelo dolgotrajna in predstavlja za sam projekt in organizacijo, ki ga izvaja, tveganje. Tveganje je še večje, če se na modeliranje v organizaciji gleda, kot da je v domeni »specialistov za modeliranje«, ki naj bi edini razumeli »svoje« modele. Pravilen pristop izhaja s stališča, da je namen modelov lažje sporazumevanje med vsemi sodelujočimi pri projektu. Pri tem je izredno pomembno, da so modeli razumljivi izvajalcem procesov in da hkrati realno opisujejo zajeti proces.

Na podlagi rezultatov analize modelov poslovnih procesov se izdela predloge sprememb v izvajanju poslovnih procesov in njihove informatizacije. S pomočjo simulacij lahko enostavno ugotovimo učinke predlaganih sprememb (zmanjšanje stroškov, krajše čase izvajanja procesov ...). Ponavadi predlagane spremembe procesov vodijo tudi v organizacijske spremembe, za uvedbo le-teh pa je najpomembnejša podpora in sodelovanje vodstva.

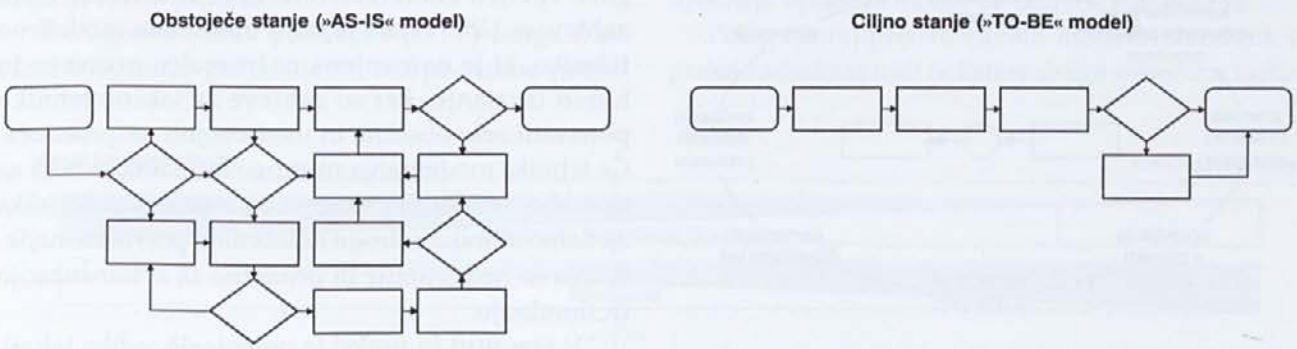
3 NAMEN POSLOVNEGA MODELIRANJA

Preden se lotimo modeliranja poslovnih procesov, je treba določiti namen modeliranja, kar vpliva na izbiro tehnike. Različne modelirne tehnike namreč ustrezajo različnim namenom.

Phalp [22] ločuje dva namena uporabe modelov poslovnih procesov: za klasični razvoj programskih rešitev in za prenovo poslovnih procesov. Slednjega v [21] podrobneje razdela na:

- pragmatični pristop za modeliranje in razumevanje procesov,
- dosledno paradigmo za analize procesov in
- predstavitev procesov.

Pri *zajemanju* (modeliranju) poslovnega procesa v okviru pragmatičnega pristopa uporabniki model največkrat le opazujejo, zato posebna interakcija z modelom ni potrebna. Pri *analiziranju* poslovnih procesov

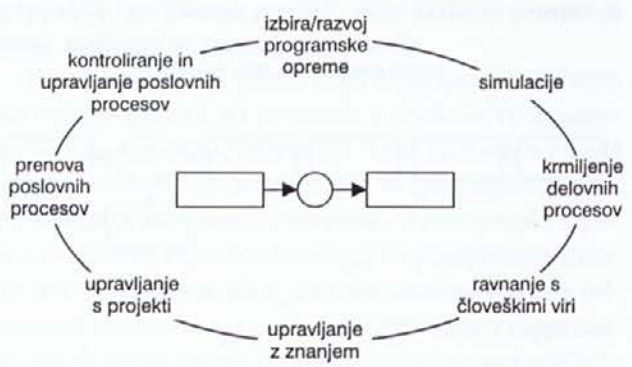


Slika 1: **Modela obstoječega in ciljnega stanja poslovnih procesov**

potrebujemo bolj dodelane mehanizme, kot so kvalitativne analize statičnih diagramov modelov. Tukaj so ustrezni tisti modeli, s katerimi lahko prikažemo tako dinamični kot funkcijski vidik procesov. V tem primeru bi lahko uporabniki želeli modele, ki jim omogočajo večjo interakcijo (npr. simulacije) pri izvajanju »kajče« (angl. what-if) analiz. Pri *predstavitvi* poslovnega procesa je najprimernejša enostavno razumljiva (največkrat diagramska) notacija.

Gigalis in Doukidis [8] namen modelov poslovnih procesov povezujeta zlasti z upravljanjem in ravnanjem s spremembami v podjetju. Spremembe se nanašajo na potrebe po učenju, analizah, spremljanju in kasnejšemu kontroliranju poslovnih procesov, za kar so potrebni opisni modeli in modeli za podporo odločanju. Najbolj znani tovrstni pristopi so: prenova poslovnih procesov (angl. Business Process Reengineering – BPR), nenehno izboljševanje procesov (angl. Continuous Process Improvement – CPI), celovito upravljanje s kakovostjo (angl. Total Quality Management – TQM) in organizacijsko preoblikovanje (angl. Organisational Transformation – OT). Podobno tudi v [28] ugotavljajo, da je glede na namen treba izdelati več različnih modelov za analizo poslovnih procesov.

Ugotavljamo, da so lahko opisi poslovnih procesov podani na različnih nivojih podrobnosti, odvisno od tega, s kakšno stopnjo abstrakcije analiziramo organizacijo. Stopnja abstrakcije pa je seveda odvisna od namena modeliranja. V [1] ugotavlja, da se modeli poslovnih procesov največkrat uporabljajo za razvoj programskih rešitev za podporo procesom ali pa za analizo samih procesov. V obeh primerih se od modela včasih zahteva *opis* procesa (prek zajema podatkov ali samo predstavitve), s katerim *spoznamo* proces.



Slika 2: **Nameni modeliranja poslovnih procesov (prirejeno po [24])**

Včasih so modeli potrebni za *odločanje o načrtu ali razvoju* procesov. Smoter je tukaj razvoj primernih poslovnih procesov, zato je namen modelov *analiza*. Pri *izvajanju* procesov so včasih potrebne odločitve, s katerimi zagotavljamo pravilno izvajanje. Iz tega izhaja potreba po modelih, s katerimi *nadziramo in kontroliramo* procese in zagotavljamo prave podatke za podporo tem odločitvam. Pri *razvoju programskih rešitev* za podporo poslovnim procesom so še posebej koristni izvedljivi modeli. Rosemann [24[podaja še druge namene modeliranja, ki jih prikazujemo na sliki 2.

Namene poslovnih modelov lahko torej združimo v štiri glavne kategorije:

- opisni modeli za spoznavanje procesov,
- opisni in analitični modeli za podporo odločanju pri razvoju in načrtovanju procesov,
- izvedbeni ali analitični modeli za podporo odločanju pri izvajanju in kontroliranju procesov in
- izvedbeno podporni modeli za razvoj programskih rešitev.

4 IZBIRA TEHNIKE IN ORODJA ZA MODELIRANJE POSLOVNIH PROCESOV IN KAKOVOST MODELOV POSLOVNIH PROCESOV

Modeliranje poslovnih procesov in njihova analiza sta ključna dejavnika za razumevanje, izboljšavo in dokumentiranje poslovnih procesov. Vse večje zanimanje za modeliranje poslovnih procesov, da bi izboljšali obstoječe stanje, je privedlo do hitre rasti števila tehnik in orodij za modeliranje [19].

Na področju modeliranja poslovnih procesov je smiselna in priporočljiva uporaba že znanih tehnik, ki so bile razvite in uveljavljene predvsem na področju modeliranja informacijskih sistemov [16]. Obstajajo primeri poskusov uvajanja univerzalne tehnike za modeliranje poslovnih procesov, vendar izkušnje in ugotovitve iz prejšnjega poglavja kažejo, da je nemogoče vpeljati enoten standard, ki bi ustrezal vsem zahtevam [25]. Težko je najti optimalno modelirno tehniko, ki je namenjena načrtovalcu procesov in hkrati izvajanju, ker so zahteve za takšno tehniko ponavadi zelo obsežne in medsebojno nasprotujoče. Če tehnika modeliranja ni strogo formalna, je lažja za uporabo, vendar ni primerna za avtomatizacijo (slika 3). Samo tehnike s strogo določenimi pravili za modeliranje so nedvoumne in primerne za avtomatizacijo in simulacijo.

V literaturi in praksi je omenjenih veliko tehnik modeliranja poslovnih procesov. V nadaljevanju

predstavljamo tehnike, ki so v domači praksi najbolj razširjene.

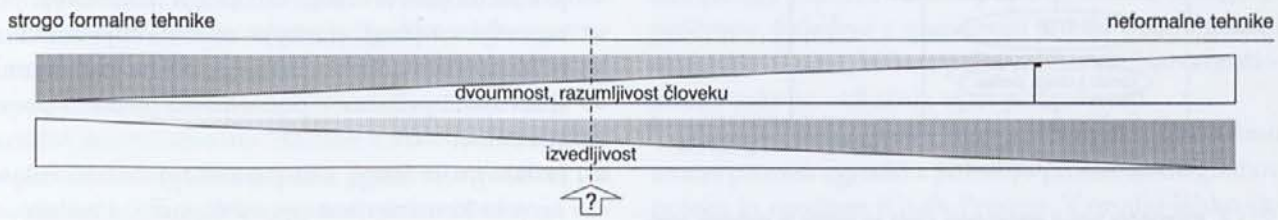
Tehnika **diagramov poteka** (angl. Flow Chart) je definirana kot formalna grafična predstavitev zaporedja programske logike, delovnega ali proizvodnega procesa, organigrama ali podobne formalizirane strukture [17]. Tehnika uporablja grafične simbole za predstavitev operacij, podatkov ali toka podatkov, s katerimi definiramo, analiziramo ali rešujemo problem. Glavna značilnost diagramov poteka je fleksibilnost. Proces lahko opišemo na več različnih načinov; odločitev je na strani analitika. Modeli, izdelani s to tehniko, so lahko razumljivi, dobri za medsebojno komunikacijo, njihova uporaba pa je enostavna. Slabost tehnike se kaže v preveliki fleksibilnosti. Meje procesa so lahko hitro nejasne, diagrami poteka so lahko hitro preveliki, tehnika ne ločuje glavnih aktivnosti in podaktivnosti. Uporaba diagramov poteka je smiselna za potrebe podrobnih modelov, manj za splošen pregled nad procesom (zaradi težav pri povezovanju organizacijskih enot – oddelkov z aktivnostmi). Sorodna tehniki diagramov poteka je tehnika **procesnih diagramov poteka**. V primerjavi z diagrami poteka nam ta tehnika na enostaven način pomaga pri ugotavljanju prehoda toka poslovnega procesa med enotami znotraj organizacije in zunaj meja organizacije, saj prikazuje odgovornost organizacijskih enot za posamezne aktivnosti.

Tehnika **EPC** (angl. Eventdriven Process Chain) je ena najbolj razširjenih tehnik na področju poslovnega modeliranja. Predstavitev poslovanja organizacije s to tehniko je dosledna: vsako aktivnost mora v modelu obvezno sprožiti (poslovni) dogodek, iz nje pa mora obvezno izhajati nov (poslovni) dogodek. Za izvajanje aktivnosti morajo biti opredeljeni izvajalci in potrebni viri (npr. podatkovni). V modelu morajo biti tudi dosledno opredeljena vsa razvejišča in združevanja tokov procesa.

Z **diagrami tokov podatkov (DTP)** (angl. Data Flow Diagrams) prikazujemo tok podatkov (informacij) v procesu. DTP prikazujejo povezave med procesi in v kakšnem odnosu so ti procesi do uporabnikov in zunanjega okolja. Z uporabo DTP lahko analitik opredeli proces na logičnem nivoju: prikazano bo, kaj se v procesu dogaja, ne pa kako. DTP se uporabljajo v komunikaciji med analitikom in uporabniki, saj so lahko razumljivi, enostavni za risanje in spreminjanje. Omogočajo razgradnjo posameznega procesa v podrobnejši podproces oz. podprocese. DTP se v funkcijskem modelu uporabljajo za prikaz pomena operacij in omejitev in funkcijsko odvisnost. Prikazujejo vstop podatkov v proces, aktivnosti, ki s temi podatki upravljajo, kje so podatki shranjeni in organizacijsko funkcijo, v katero ta aktivnost sodi. Tehnika DTP predstavlja zaradi svoje enostavne uporabe najširše uporabljeno tehniko na področju strukturne analize in informacijskega inženirstva [1].

Tehnika **Petrijevih mrež** sega v šestdeseta in sedemdeseta leta prejšnjega stoletja. Petrijeve mreže temeljijo na formalni, matematični predstavitvi z dobro definirano sintakso in semantiko. V praksi se je uporaba Petrijevih mrež pokazala kot manj primerna. Glavna razloga sta bila v pomanjkanju podatkovnih konceptov (modeli so hitro postali ogromni in so vključevali vse podrobnosti v svoji mrežni strukturi) in hierarhičnih konceptov (ni bilo možnosti gradnje večjega modela prek različnih podmodelov). Problem so rešili z razvojem barvnih Petrijevih mrež, ki vključujejo tako strukturiranje podatkov in hierarhično razgradnjo modelov. V barvnih Petrijevih mrežah sestavljajo procesni model procesi (na elementarnem nivoju so to aktivnosti), objekti in skladišča objektov, ki so med seboj povezani z usmerjenimi povezavami. Mreži procesnega modela lahko priredimo podprocese in jih razčlenimo vse do elementarnih aktivnosti. Skladiščem objektov in povezavam lahko pripišemo dodatne pogoje (lastnosti in omejitve), s katerimi podrobneje opišemo model za potrebe simuliranja.

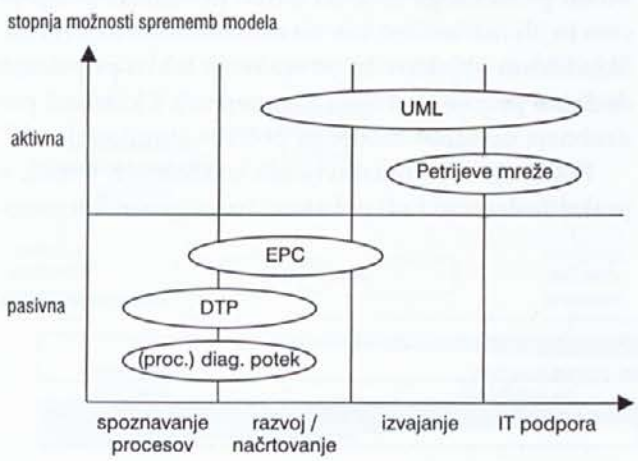
Poleg zgoraj predstavljenih modelirnih tehnik v praksi zasledimo tudi nekatere druge generične meto-



Slika 3: **Nasprotujoče si zahteve za splošno tehniko modeliranja [25]**

dologije z možnostjo modeliranja procesov. Tak primer so **simulacije**. Kelton et al. [15] opredelijo simulacije kot skupek metod za posnemanje obnašanja realnega sistema. Glede na določene značilnosti ločimo deterministične (vhodni podatki so vnaprej določeni) ali stohastične (vhodni podatki so naključni) simulacije, statične (časovna komponenta ni prisotna) ali dinamične (čas je bistvena komponenta) simulacije ter zvezne (stanje sistema se nenehno spreminja) ali diskretne (dogodki nastopijo v različnih časovnih trenutkih). Procese, ki jih obravnavamo kot sistem, lahko modeliramo s pomočjo simulacij, da bi spoznali obnašanje procesov ali pa da bi lahko ocenjevali različne scenarije spremenjenega izvajanja procesov. Same simulacije so lahko povezane z drugimi modelirnimi tehnikami (npr. Petrijevimi mrežami), tako da je tehnika modeliranja pogojena z izbiro simulacijskega orodja, a je z uporabnikovega vidika to manj pomembno, saj »komunicira« neposredno z orodjem. Nasprotno pa orodje omogoča uporabniku gradnjo modelov, ki so sestavljeni iz več modelirnih tehnik. Slabost simulacij je predvsem nezmožnost natančnega modeliranja obnašanja realnega sistema zaradi prisotnosti velikega števila spremenljivih dejavnikov. Simulacijsko modeliranje pa tudi zahteva velike investicije z vidika časa in virov.

Tukaj smo predstavili le nekatere najpomembnejše, nikakor pa ne vseh tehnik za gradnjo modelov poslovnih procesov. Za tako zgrajene modele poslovnih procesov Aquilar-Saven [1] predlaga enostavno razvrstitev glede na stopnjo možnosti za spremembe modelov.



Slika 4: **Razvrstitev tehnik modeliranja glede na namen in možnosti sprememb modela**

Avtorica ločuje *pasivne* in *aktivne* spremembe. Pri pasivnih spremembah uporabniku ni omogočena interakcija z modelom ali spremembe modela brez ponovnega modeliranja celotnega procesa. Nasprotno pa aktivni modeli uporabnikom dopuščajo spremembe ali pa so sami po sebi dinamični (npr. simulacije).

Če sedaj skušamo razvrstiti predstavljene tehnike, lahko pridemo s priredbo predloga iz [1] do matrike, ki nam posamezno tehniko opredeli glede na *namen modela* in *možnosti za spremembe modela* (slika 4).

Ob sliki 4 velja poudariti, da stopnja možnosti sprememb modela ter namen modelov nista edini, temveč le eden izmed možnih kriterijev za razvrstitev tehnik poslovnega modeliranja.

Danes je na voljo že prek petdeset orodij za modeliranje (in simuliranje) poslovnih procesov [13], kar v praksi pomeni, da je izbira primernega orodja zelo težka. Večina orodij za poslovno modeliranje predstavlja modele grafično, saj izhajajo iz dejstva, da lahko ena slika včasih pove več kot tisoč besed. Razen tega nekateri omogočajo tudi simulacije izvajanja procesov, ki so uporabne tako pri analizi obstoječega stanja kot pri izvajanju »kaj-če« (angl. what-if) analiz. V tabeli 1 predstavljamo tehnike in z njimi povezana nekatera najpogosteje uporabljena orodja za procesno modeliranje.

Z namenom zagotavljanja *kakovosti poslovnih modelov* so Becker, Rosemann in von Uthman [3] razvili priporočila k modeliranju (angl. Guidelines of Modeling - GoM). Splošna vodila so:

- *pravilnost* (angl. correctness) – sintaktična in semantična pravilnost. Model je sintaktično pravilen, če je konsistenten in popoln glede na meta model. Semantično je model pravilen, če pravilno prikazuje realni svet.
- *bistvenost* (angl. relevance) – izbira bistvenega sistema za modeliranje. Elementi modela niso bistveni, če njihova odstranitev iz modela za izvajalca ni pomembna.
- *ekonomska učinkovitost* (angl. economic efficiency) – predstavlja omejitev ostalim vodilom. Primerljiva je s kriterijem izvedljivosti (angl. feasibility).
- *razumljivost* (angl. clarity) – modeliranje nerazumljivih modelov nima smisla. Model morajo razumeti izvajalci procesa – pomembno je, da ni preveč simbolov.
- *primerljivost* (angl. comparability) – skozi celoten projekt konsistentno uporabljamo ista načela
- *sistematično načrtovanje* (angl. systematic design).

Tehnika modeliranja	Orodje
(Procesni) diagrami poteka	ABC Flow Charter 4.0, ABC Graphics Suite, ABT Project Workbench, AWD and Work.ow Analyzer, Bench, Marker Plus, BPM, Business Object Modelling Workbench, Cap Web-Flow, CLEAR, COI-Business Flow, CORE, COSA, CSEWork.ow 5.0, Docu Flow, EPM SuiteFlow Maker, Flow Path, Flowcharter, Flowmark, Form Flow, IBMBusiness Process Modeler, Ithink, Igrafx Process 2000, Process Wise, Pro Model, Process Charter, Process Maker, RKB Work Frame, SA/BPR Professional, Vectus, Visual Thought, Work Flow Analyzer
EPC	ARIS-Tools, CASE Tool, 4Keeps, BONAPART
Diagrami tokov podatkov	ARIS-Tools, CASE Tool, 4Keeps, BONAPART, GRADE, INCOME, IEW, Paradigm Plus, Popkins Systems, Architect, Softwarethrough Pictures SE , ProcessWise, With Class 98, Graphics Toll
Petrijeve mreže	INCOME, Desigh CPN, UNCOME, PACE, Process Maker and Process Weaver

Tabela 1: **Nekatera orodja za procesno modeliranje po modelirnih tehnikah (povzeto po [1] in [4])**

Tudi Beer [4] je na podlagi izkušenj s projekti na tem področju postavil priporočila, ki naj bi jih posamezno orodje za modeliranje poslovnih procesov dosegalo. S primerjavo priporočil glede orodij za modeliranje [4] s priporočilom glede tehnik modeliranja [3] pri zagotavljanju kakovosti modelov ugotavljamo, da je ključnega pomena razumljivost končnih modelov, saj je število ljudi, ki modelirajo ali modele uporabljajo, vse večje, in večina, zlasti uporabnikov, z metodologijami in orodji modeliranja procesov ni dobro seznanjena. Falkenberg [7] opredeli kriterije kakovosti modelov poslovnih procesov z izraznostjo (angl. expressiveness), arbitrarnostjo (angl. arbitrariness) ter primernostjo (angl. suitability). Barros in Hofstede [2] dopolnita kriterije iz [7] še z razumljivostjo (angl. comprehensibility), popolnostjo (angl. completeness), uspešnostjo (angl. efficiency) ter učinkovitostjo (angl. effectiveness).

5 PRIMER MODELIRANJA POSLOVNEGA PROCESA V JAVNI UPRAVI

Po naših izkušnjah je pri projektih prenove poslovnih procesov za poslovno modeliranje najprimernejša tehnika procesnih diagramov poteka, zlasti zaradi enostavnosti in razumljivosti modelov, kar bistveno olajša komunikacijo med tistimi, ki procese modelirajo, in njihovimi izvajalci. Izkušnje z uporabo različnih orodij za poslovno modeliranje in simulacije (ARIS, Income, iGrafx Process) pri naših projektih kažejo, da obsežna komunikacija z izvajalci procesov zahteva preprostost in razumljivost tehnike modeliranja. Kot je bilo ugotovljeno [6], je relevantnost modela pomembnejša od popolnosti, pri čemer so preprostejši modeli lažje razumljivi nespecialistom. Skladno z matriko razvrstitve so procesni diagrami poteka posebej uporabni pri spoznavanju obstoječih procesov in načrtovanju novih ter enostavni za spreminjanje.

Procesni diagrami poteka in orodje iGrafx Process uporabljajo podobne simbole kot tehnika diagramov poteka. Pri modeliranju s to tehniko oz. orodjem simbole medsebojno povežemo z usmerjenimi povezavami, s katerimi prikazujemo tok samega procesa. Procesne diagrame poteka sestavljajo aktivnosti, ki so razporejene v enega ali več oddelkov – tj. organizacijskih enot, zadolženih za izvajanje teh aktivnosti. Posamezna aktivnost lahko vsebuje podatke o vhodih, virih, nalogah in izhodih. Orodje iGrafx Process ponuja nazorne uporabniške vmesnike, zato lahko tudi nestrokovnjaki na področju modeliranja poslovnih procesov hitro razumejo in uporabijo to tehniko. Glavna prednost tehnike je v veliki preglednosti in enostavni razumljivosti. Izbiro tega orodja dodatno opravičujejo integrirane, zmogljive in popolne simulacijske funkcije v samem orodju in podatek, da je orodje eno izmed najbolj priljubljenih orodij za modeliranje poslovnih procesov [13].

V nadaljevanju predstavljamo primer modeliranja procesa 'napredovanje v višji naziv', ki je del upravnih postopkov na enem od slovenskih ministrstev. Upravni postopki na tem ministrstvu zajemajo več kot 30 procesov, ki so razdeljeni po različnih področjih. Zaradi pogostega izvajanja procesa 'napredovanje v višji naziv' je proces zanimiv za podrobnejše proučevanje in analize, kjer lahko pričakujemo vidne izboljšave v učinkovitosti izvajanja procesa. Namen modeliranja predstavljenega primera sta bila popis in analiza obstoječih procesov. Skladno z namenom je bilo treba zgraditi razumljiv model, ki bo upošteval načelo bistvenosti – model zato ne vključuje vseh podrobnosti.

Zaradi razumljivosti in preprostosti tehnike smo model procesa zgradili s pomočjo procesnih diagramov poteka in orodjem iGrafx Process. V orodju lahko aktivnosti podrobneje opišemo s številnimi atributi, na

primer vrsta in število virov, ki aktivnost izvajajo, trajanje aktivnosti (konstantno ali stohastično) in z različnimi vrstami stroškov. Stroške uporabe virov lahko določimo na različne načine (z rednimi urnimi postavkami, s postavkami za nadurno delo, s stroški za uporabo vira), urnike virov ter generatorje dogodkov lahko poljubno prilagajamo. Osnovne elemente modela tehnike procesnih diagramov poteka prikazujemo na sliki 5.

Orodje nam omogoča simulacije izvajanja procesov s pomočjo vizualnega sledenja, s katerim lahko ugotovimo ozka grla v procesu. Rezultati simulacije so prikazani v poročilih (stroškov, časa, virov in drugih), iz katerih dobimo podrobno analizo stroškov, časa ter virov poslovnega procesa. Ocenjujemo lahko npr. trajanje izvajanja procesa, stroške celotnega procesa ali posameznih aktivnosti, obremenjenost virov. S pomočjo funkcij za diskretne simulacije lahko tudi ocenjujemo in predvidimo učinke predlogov prenove procesov.

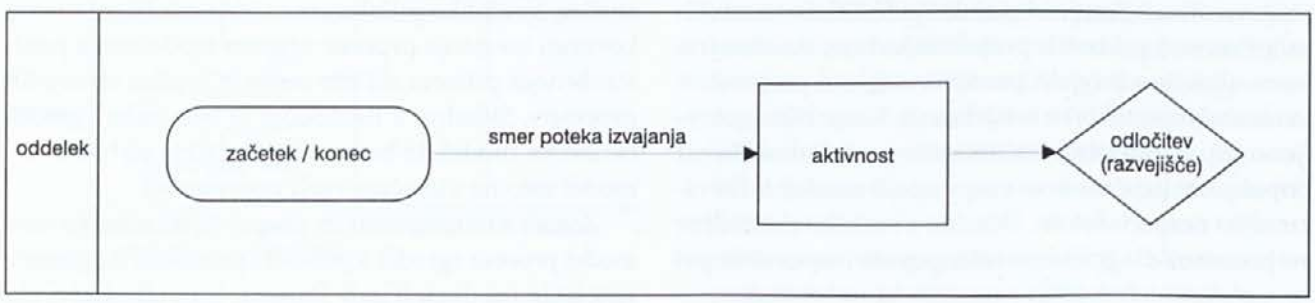
Model procesa je bil zgrajen na podlagi intervjujev z izvajalci procesa in s pomočjo razpoložljive dokumentacije. Model obstoječega stanja je bil razvit v več iteracijah, da bi izvajalci procesa lahko sproti preverjali model. V začetnih iteracijah je bil s pomočjo vodilnega kadra postavljen okvirni model procesa, ki smo ga v nadaljevanju z intervjuji operativnega kadra podrobneje razdelali. Vodilni kader je opredelil organizacijske enote, v katerih se odvija predstavljeni proces, ter temeljne aktivnosti. Z operativnimi izvajalci smo nato temeljne aktivnosti podrobneje opredelili. Pri samem poteku modeliranja smo opazili naslednje probleme, povezane z:

- *načinom razmišljanja*: izvajalci procesa so izhajali iz tega, da so ljudje osnovni elementi procesa in da je treba vsakega izmed njih vključiti v model, ne glede na to, ali pri obravnavanem procesu aktivno sodeluje ali ne;
- *neuravnoteženostjo*: pri nekaterih procesih so bile aktivnosti preveč podrobno razdelane, drugod pa preveč splošno. Vzroke vidimo na eni strani v »umetnem« ustvarjanju obsega dela, na drugi strani pa v nezainteresiranosti izvajalcev za sodelovanje v projektu oz. v nepoznavanju procesa;
- *točnostjo podatkov*: posredovani podatki s strani izvajalcev so večkrat bili podani z nerealnimi oz. nesmiselnimi vrednostmi;
- *nepoznavanjem procesa*: zaradi velike fluktuacije zaposlenih veliko izvajalcev samega procesa sploh ne pozna;
- *nedefiniranostjo*: procesi so velikokrat nedefinirani in se med seboj prepletajo, tako da je bilo težko določiti meje procesov.

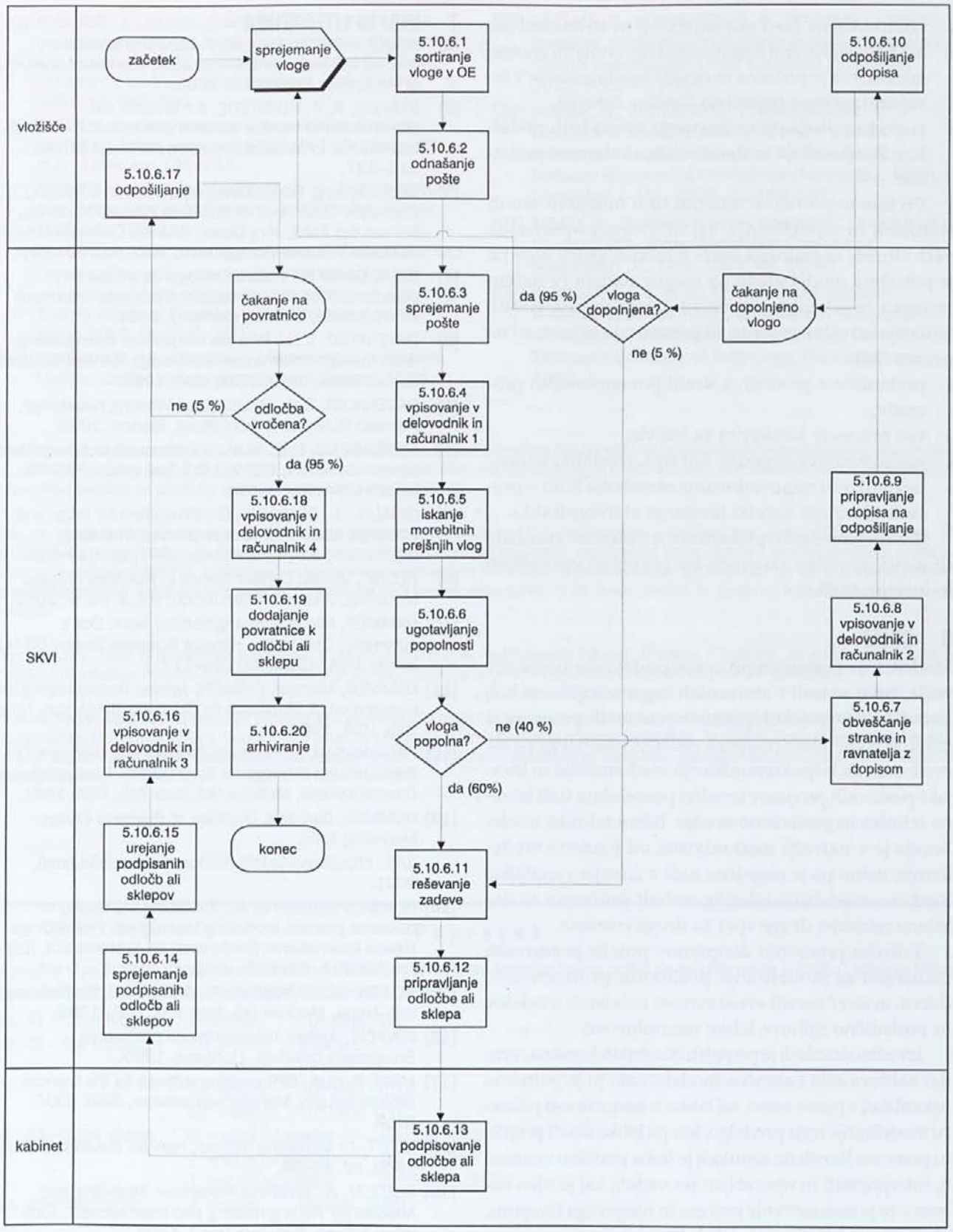
Na sliki 6 prikazujemo model procesa 'napredovanje v višji naziv'. Proces poteka v treh oddelkih: vložišču, sektorju za kadrovske in upravne zadeve ter kabinetu ministra. V procesu se obdela približno 2500 vlog letno. 60 % vlog je popolnih, stopnja pa doseže 80 % ob dopolnitvi nepopolnih vlog. Lastnik procesa je sektor za kadrovske in upravne zadeve, kjer vlogo obdelajo štirje strokovni delavci. Vloge vedno sprejemajo v vložišču. Status vloge vedno evidentirajo na dva načina: z ročnim vnosom in prek računalniškega programa. Odločbo na koncu podpiše minister.

Nekatere pomanjkljivosti v izvajanju poslovnih procesov so razvidne že iz same slike procesa, druge pa se izkažejo pri simulaciji. S pomočjo simulacije smo tako ugotovili povprečni čas izvajanja procesa - 49 dni. Efektivno traja delo manj kot 1 dan, preostali dnevi so čakanja (podpisovanje, prenos dokumentacije, čakanje na dokončanje vloge).

Kvantitativni rezultati simulacij, ne glede na njihovo podrobnost in natančnost, predstavljajo le enega izmed vidikov analize poslovnega procesa. Procesni diagrami poteka lahko pogosto pokažejo več problemov, ki smo jih sprva spregledali. V našem primeru lahko ugotovimo dvoje (zelo pogostih) problemov, ki smo jih opazili tudi pri ostalih podobnih procesih:



Slika 5: **Osnovni elementi modela tehnike procesnih diagramov poteka**



Slika 6: **Model procesa 'napredovanje v višji naziv'**

- komunikacija med ministrstvom in strankami ter med posameznimi organizacijskimi enotami znotraj ministrstva je počasna in neučinkovita, zato se v izvajanju procesa pojavljajo številna čakanja,
- pogosto prihaja do večkratnega vnosa istih podatkov, kar povečuje možnost »slabe« kakovosti podatkov.

Pri tem je potrebno omeniti tudi omejitve samih simulacij in modeliranja, saj ni mogoče »prenesti« vseh situacij iz realnega sveta v model, poleg tega pa je potrebno model glede na njegov namen (v našem primeru za izvedbo simulacij) ustrezno prilagoditi. Situacije iz realnega sveta, ki povzročajo težave, so na primer [26]:

- prekinitve v procesih s strani pomembnejših procesov,
- več procesov konkurira za isti vir,
- nepredvidljivi dogodki, kot npr. izostanki kadra,
- nezmožnost napovedovanja obnašanja ljudi v procesih (kasnejši začetki izvajanja aktivnosti itd.).

Zaradi navedenih problemov je potrebno rezultate simulacij pravilno razumeti ter previdno uporabljati in interpretirati.

6 SKLEP

Modeliranje poslovnih procesov postaja vse pomembnejše, česar se tudi v slovenskih organizacijah vse bolj zavedajo. Pri projektih prenove poslovnih procesov je izredno pomembna podpora vodstva, razen tega pa sta predvsem za lažjo komunikacijo med analitiki in izvajalci poslovnih procesov izredno pomembna tudi izbrana tehnika in posledično orodje. Izbira tehnike modeliranja je v največji meri odvisna od namena modeliranja, delno pa je pogojena tudi z znanjem analitika. Različne modelirne tehnike so bolj primerne za določene namene, druge spet za druge namene.

Tehnika procesnih diagramov poteka je po naših izkušnjah za modeliranje poslovnih procesov zelo dobra, in sicer zaradi enostavnosti izdelanih modelov in posledično njihove lahke razumljivosti.

Izvedba simulacij je po naših izkušnjah koristna, vendar zahteva zelo natančne modele, zato jo je potrebno uporabljati s pravo mero, saj lahko v nasprotnem primeru modeliranje traja predolgo, kar pa lahko škodi projektu prenove. Rezultate simulacij je treba pravilno razumeti, interpretirati in uporabljati ter vedeti, kaj je njen namen – to je razumevanje procesa in njegovega izvajanja, sama pa neposredno ne daje odgovorov.

7 VIRI IN LITERATURA

[1] AQUILAR-SAVEN, Ruth Sara: Business process modelling: Review and Framework, International Journal of Production Economics, 2003.

[2] BARROS, A. P., HOFSTEDE, A.: Towards the construction of wokfow suitable conceptual modelling techniques, Information Systems Journal 8 (4), str. 313–337.

[3] BECKER, Jorg, ROSEMANN, Michael, von UTHMAN, Christoph: Guidelines of Business Process Modeling, v Wil van der Aalst, Jörg Desel, Andreas Oberweis (Eds.): Business Process Management, 2000, str. 30– 49.

[4] BEER, Daniel B.: Process models as a base for communication and revitalization projects, Informatik im Bauwesen, Weimer, Germany, 2002.

[5] DAVENPORT, T. H.: Process Innovation: Reenginering Work through Information Technology, Harvard Business School Press, Boston, MA, USA, 1993.

[6] DAVENPORT, T. H., PRUSAK, L.: Working Knowledge, Harward Business School Press, Boston, 1998.

[7] FALKENBERG, E. D. et al.: A Framework of Information System Concepts, IFIP WG 8.1 Task group, FRISCO, Leiden University, Leiden.

[8] GIGALIS, G., DOUKIDIS, G.: Simulation for intra- and interorganizational business process modelling, Informatica, 21 (4), Ljubljana, 1997, str. 613–620.

[9] HLUPIC, Vlatka: Current Trends in Business Process Modelling, Journal of Simulation, vol. 2, no. 2, 2001.

[10] HAMMER, Michael: Reengineering work: Don't automate. Obliterate., Harvard Business Review 68 (4), Jersey, USA, 1990. str. 104–112.

[11] HAMMER, Michael, CHAMPY, James: Reengineering the Corporation: A Manifesto for Business Revolution, New York, USA, 1993.

[12] HARRINGTON, J.: Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity and Competitivness, McGraw Hill, New York, USA, 1991.

[13] HOMMES, Bart-Jan: Overview of Business Proces Modelling Tools,
[URL: http://is.twi.tudelft.nl/čhommes/scr3tool.html], 2001.

[14] HOMMES Bart-Jan et al.: Assessing the quality of business process modelling techniques, Proceedings Hawaii International Conference on Systems SCI, IEEE Los Alamitos, CA, USA, str. 5.

[15] KELTON, W., SADOWSKI, R., SADOWSKI, D.: Simulation with Arena, McGraw Hill, New York, USA, 1996.

[16] KOVAČIČ, Andrej: Informatizacija poslovanja, Ekonomska fakulteta, Ljubljana, 1999.

[17] LAKIN, R. et al.: BPR enabling software for the financial services industry, Management services, ISSN: 0307-6768.

[18] LEVITT, J.: Marketing myopia, Harvard Business Review, 1960, str. 45–56.

[19] OULD, M. A.: Business Processes: Modelling and Analysis for Re-engineering and Improvement, John Wiley & Sons, New York [etc.], 1995.

[20] PAOLUCCI, E., BONCI, F. and RUSSI, V.: Redisigning organisations through business process re-engineering and object-orientation, Proceedings of the European Conference on Information Systems, Cork, Ireland, 1997, str. 587–601.

[21] PHALP, K. T.: CAP framework for business process modelling, Information and Software Technology, 40 (13), 1998, str. 731–744.

[22] PHALP, K. et al.: Quantitive analysis of static models of processes, Journal of Systems and Software, 52 (2), 1999, str. 105–112.

[23] POPOVIČ, A., GROZNIK, A., INDIHAR ŠTEMBERGER, M.: Prenova poslovnih procesov in informatizacija poslovanja – vloga poslovnega modeliranja in simulacij, Zbornik referatov Posvetovanje informatikov v javni upravi 2003, Portorož, 2003.

[24] ROSEMANN, M.: Complexity Management in Process Models, Gabler-Verlag, Wiesbaden, Germany, 1996.

[25] ROZMAN, T., HORVAT VAJDE, R., ROZMAN, I.: Srebrni metek za modeliranje in izvajanje poslovnih procesov?, Zbornik posvetovanja Dnevi slovenske informatike 2003, Portorož, 2003.

[26] TARUMI, H., MATSUYAMA, T., KAMBAYASHI, Y.: Evolution of business processes and a process simulation tool,.*Proceedings of the Asia-Pacific Software Engineering Conference* (Takamatsu, Japan, December 7–10), 2000, str. 180–187.

[27] TUMAY, K.: Business process simulation, Proceedings of the WSC'95 – Winter Simulation Conference, Washington DC, USA, 1995, str. 55–60.

[28] WORKMAN, J. C. et al.: On the relation between business, business model, software and ACT-platform architectures, Information and Technology Division, Eindhoven University of Technology, The Netherlands, 2000.

▪

Aleš Popovič je asistent na Ekonomski fakulteti Univerze v Ljubljani. Na dodiplomskem študiju vodi vaje na poslovno-informacijski smeri. Raziskovalno se ukvarja z modeliranjem in simulacijami poslovnih procesov ter informacijsko tehnologijo v izobraževanju. Kot sodelavec Inštituta za poslovno informatiko sodeluje na projektih prenove in informatizacije poslovanja.

▪

Doc. dr. Mojca Indihar Štemberger je docentka za poslovno informatiko na Ekonomski fakulteti v Ljubljani, kjer je leta 2000 doktorirala. Na dodiplomskem in podiplomskem študiju sodeluje kot predavateljica pri več predmetih, raziskovalno pa se ukvarja s prenovo poslovnih procesov, sistemi za podporo odločanju, e-poslovanjem ter uporabo informacijske tehnologije pri izobraževanju. Sodeluje tudi pri več aplikativnih projektih s področja prenove poslovnih procesov in strateškega načrtovanja informatike, ki jih izvaja Inštitut za poslovno informatiko na Ekonomski fakulteti.

▪

Doc. dr. Jurij Jaklič je predavatelj predmetov s področja informatike na Ekonomski fakulteti Univerze v Ljubljani. Je sodelavec Inštituta za poslovno informatiko. Njegovo področje raziskovanja so zlasti podatkovne baze, e-poslovanje in sistemi za podporo odločanju.

▪

Prof. dr. Andrej Kovačič je predavatelj predmetov s področja informatike in prenove poslovanja ter predstojnik Inštituta za poslovno informatiko na Ekonomski fakulteti v Ljubljani. Vodi in izvaja projekte s področja informatizacije in prenove poslovanja. Je veščak Zveze ekonomistov Slovenije na področju upravljanja in ravnateljevanja, pooblaščeni revizor informacijskih sistemov ter svetovalec (management consultant) na mednarodnih projektih PHARE.

▪

Popravek

V prejšnji številki revije Uporabna informatika (1/2004) je pri pripravi besedila za tisk prišlo do neljubih napak v članku Ladislava Mikole *Uporaba desetiških SI predpon in predpon v informatiki*, in sicer:

str. 47 – desni stolpec,	4. vrstica:	10^6 nam. 106
str. 48 – levi stolpec,	10. vrstica:	*L* nam. L
	17. vrstica:	μF nam. mF
	19. vrstica:	10^{12} nam. 1012
	23. vrstica:	10^9 nam. 109
str. 48 – desni stolpec,	6. vrstica od spodaj:	(6) nam. [6](6)
	3. vrstica od spodaj:	2^{10} nam. 210
	1. vrstica od spodaj:	$(2^{10})^1$ nam. (210)1
str. 49 – levi stolpec,	4. vrstica:	(6) nam. [6](6)

Bralcem in avtorju se opravičujemo in zahvaljujemo za razumevanje.

Ugotavljanje vsebnosti točk nad posplošenimi mnogokotniki

Matej Gomboši
Laboratorij za geometrijsko modeliranje in algoritme multimedijev
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru, Smetanova 17, 2000 Maribor
matej.gombosi@uni-mb.si

Izvleček

V članku je opisan razširjen algoritem za določanje vsebnosti točk nad posplošenimi mnogokotniki, ki poleg daljic vsebujejo tudi krožne loke. Algoritem uporablja klasično metodo sekanja žarka. Razlika je v tem, da moramo testirati dve vrsti objektov. Nalogo opravimo z enostavnimi in učinkovitimi testi, ki nam hitro odgovorijo na vprašanje. Z uporabo ustreznih podatkovnih struktur nalogo rešimo zanesljivo in enostavno. Kljub razširitvi deluje algoritem še vedno v linearni časovni zahtevnosti.

Abstract

Containment Test for Generalized Polygons

The paper describes an extended algorithm for solving the point-in-polygon problem. The polygon in this case consists of straight edges and also of circular arcs. This represents a generalization of Reuleaux polygon. The algorithm uses the classical ray intersection method. The difference is that we have two types of geometric objects to test for intersections. Processing is done with simple and efficient tests, which quickly answer our question. Using the appropriate data structure, this task can be done safely and easily. Despite the extension of the classical ray intersection method, the algorithm still runs in linear time complexity.

1 Uvod

Vsebnostni test je eden osnovnih in pogosto uporabljanih algoritmov v računalniški geometriji in njenih aplikacijah [9]. To še posebej velja za računalniško grafiko, sisteme CAD/CAM in GIS aplikacije.

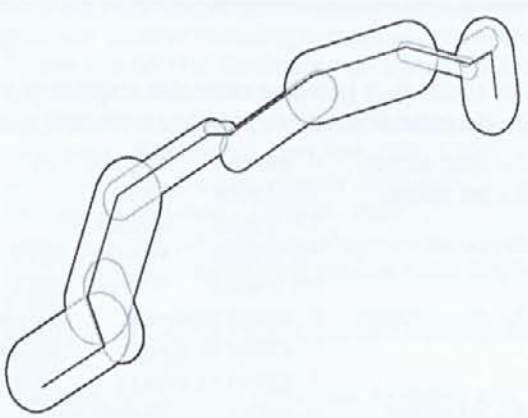
Algoritem je odvisen od tipa mnogokotnikov, s katerimi delamo:

1. mnogokotniki z ravnimi robovi,
2. mnogokotniki, ki vsebujejo tudi krožne loke.

Reševanje problemov prvega tipa je lažje in hitrejše. Večinoma se tudi srečujemo s takšnimi mnogokotniki. Posledica tega je, da so bile dosedanje raziskave usmerjene pretežno v to smer. Tako obstaja precej dobrih algoritmov, ki rešujejo ta problem: metoda sekanja žarka [4], [6], kodirani koordinatni sistem [2], metoda trikotnikov [3], Swathova metoda [10], algoritem na osnovi uniformne delitve ravnine [14].

Vsi ti algoritmi sprejmejo le mnogokotnike z ravnimi robovi. Mi pa se želimo osrediniti na posplošene mnogokotnike, ki vsebujejo tudi krožne loke. Rešitev za ta tip mnogokotnikov še ni bila podana, zato smo se odločili, da skonstruiramo svoj algoritem. Za osnovo smo vzeli metodo sekanja žarka. Dejstvo, da operiramo s krožnimi loki, nekoliko spremeni in oteži osnovni vsebnostni test. Naša želja je bila skonstruirati hiter in zanesljiv algoritem, ki bi poleg daljic učinkovito obravnaval tudi krožne loke.

Glavno motivacijo so nam predstavljali problemi iz geodezije, kjer se večkrat srečujejo s krožnimi loki. Realen primer na sliki 1 ponazarja naš problem. To je problem onesnaženja okolice cest z izpušnimi plini. Predstavljajmo si cesto, ki je ponazorjena s povezanimi daljicami. Vedeli bi radi, kako se izpušni plini širijo v



Slika 1: **Problem širjenja izpušnih plinov ponazorjen s pomočjo posplošenega mnogokotnika**

okolico ceste in kateri objekti in zgradbe so znotraj onesnaženega področja. Za predstavitev tega problema potrebujemo krožne loke, saj se plini širijo v vse smeri enako. Na sliki 1 vidimo primer ceste in širjenja plinov. Povezane daljice v sredini predstavljajo cesto. Področje širjenja izpušnih plinov je predstavljeno kot mnogokotnik s krožnimi loki. Imamo majhne mnogokotnike za posamezne odseke ceste in en skupni mnogokotnik za celo področje, ki smo ga dobili z združevanjem manjših. To nam je omogočil algoritem za tvorbo očrtij [13].

Drugo poglavje podaja nekaj osnovnih definicij, uporabljenih v tem članku. Poglavje tri predstavlja glavno idejo algoritma. Četrto poglavje obravnava robne primere. Analizo algoritma podaja peto poglavje. V šestem poglavju so podani zaključki in ugotovitve. Sedmo poglavje prikazuje relevantno literaturo.

2 Definicije

Posplošimo definicijo enostavnega mnogokotnika [7]: Imamo n točk $p_0, p_1, \ldots, p_{n-1}$ v ravnini. Pari točk p_0p_1, $p_1p_2, \ldots, p_{n-1}p_0$ so povezani z daljicami ali krožnimi loki. Določajo enostaven posplošen mnogokotnik P_g, če velja:

- sosednje daljice ali krožni loki se dotikajo v eni sami skupni točki p_i, $0 \leq i < n$,
- nesosednje daljice nimajo skupnih točk.

Daljice (označene z e_i na sliki 2) in krožni loki (označeni z a_i) s središči c_i predstavljajo robove posplošenega mnogokotnika, točke v_i pa robne točke. Zaporedje robov, ki deli ravnino v omejen in neomejen del, se imenuje zanka [7]. Vsak posplošen mnogokotnik ima natanko eno zanko. Prstan predstavlja luknjo znotraj mnogokotnika. Luknje so lahko tudi vgnezdene in tvorijo hierarhijo. Zanka je protiurno usmerjena, luknja pa sourno.

Poglejmo malo bliže strukturo krožnega loka a_i na sliki 3. Polmer je označen z r_i. Končni točki sta v_i in v_{i+1}. Ti dve točki določata tetivo. Ker je mnogokotnik orientiran, nam ta tetiva predstavlja vektor (v_iv_{i+1}) in nam pove tudi, na kateri strani tega vektorja leži mnogokotnik. V primeru slike 3 leži mnogokotnik na desni.

3 Algoritem

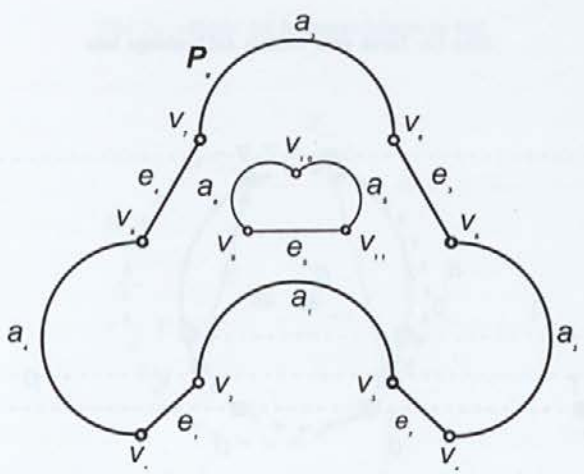
Za ugotavljanje, ali je neka točka znotraj ali zunaj mnogokotnika, se uporablja metoda sekanja žarka. Iz točke, ki jo testiram, pošljemo žarek in preverimo kolikokrat seka mnogokotnik. Za določitev vsebnosti uporabimo liho-sodo pravilo. To pravi, da če imamo liho število presečišč, je točka znotraj mnogokotnika, če jih je pa sodo število, je točka zunaj mnogokotnika. Za lažje in hitrejše računanje je naš žarek vodoraven. Tako dobimo vodoraven poltrak. Usmerjenost levo ali desno ni pomembna. Na sliki 4 vidimo primer žarka p iz točke t, ki je zunaj mnogokotnika P.

Žarek seka eno daljico in dva krožna loka. Krožni lok a_1 je presekan na dveh mestih. To nam da štiri presečišča in liho-sodo pravilo pravi, da je v tem primeru točka zunaj mnogokotnika.

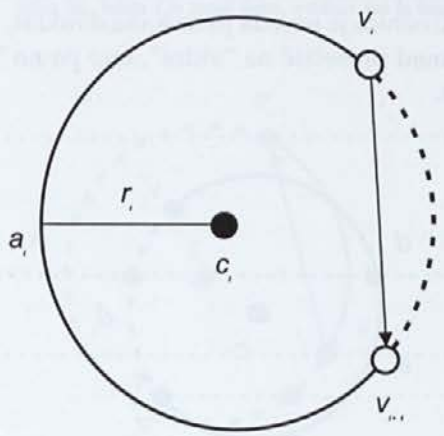
Algoritem deluje v dveh korakih:

1. iskanje presečišč med ravnimi robovi mnogokotnika in poltrakom,
2. določanje presečišč med krožnimi loki in poltrakom.

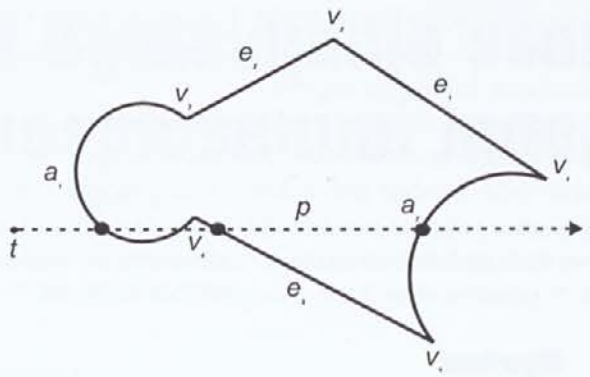
Prvi del algoritma je lažji, saj je test presečišča med vodoravnim poltrakom in daljico dobro poznan in ni



Slika 2: **Posplošen mnogokotnik P_g**



Slika 3: **Definicija krožnega loka**



Slika 4: **Metoda sekanja vodoravnega žarka iz točke *t***

zahteven. Dejanskega presečišča nam ni potrebno računati, ker nas zanima samo obstoj le-tega, ne pa njegove koordinate.

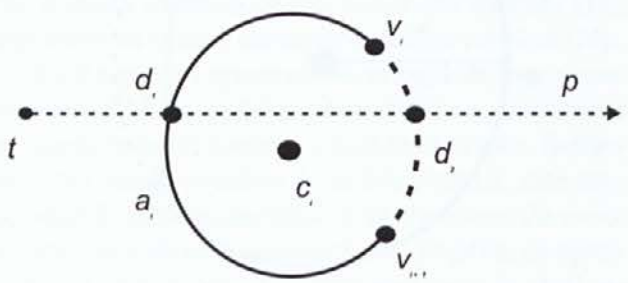
Drugi del algoritma je za nas zanimiv. Tu je bilo treba sestaviti ustrezne teste, ki bi hitro in enostavno določili obstoj presečišča med poltrakom in krožnim lokom. Slika 5 prikazuje krožni lok in poltrak, ki ga seka v točkah d_i.

Ugotoviti moramo, katera presečišča štejejo (d_1) in katera ne (d_2). Pomembno je, kako je krožni lok predstavljen. Potrebovali bomo vse podatke, omenjene v definiciji.

V splošnem imamo dve možni situaciji pri krožnih lokih:

- točka je zunaj krožnice,
- točka je znotraj krožnice.

V prvem primeru se lahko pojavita dve možnosti. Poltrak lahko seka ali pa ne seka tetive. Slika 5 kaže primer, ko poltrak seka tetivo. V tem primeru takoj brez nadaljnjih testov vemo, da imamo samo eno presečišče. Krožnica je seveda presekana dvakrat, vendar je eno izmed presečišč na "vidni", eno pa na "nevidni" strani.



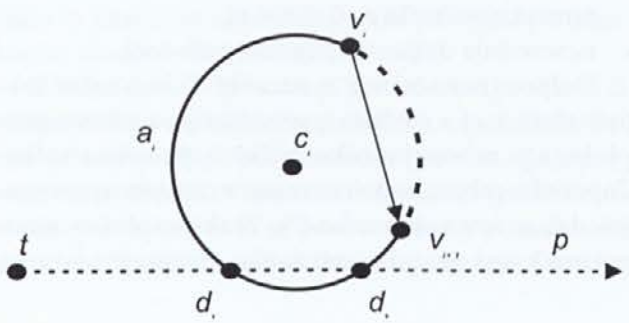
Slika 5: **Krožni lok *a*, je presekan v dveh mestih**

Tetiva je presekana, ko je žarek p znotraj vodoravnega pasu, omejenega s točkama v_i in v_{i+1} ($t.y < v_i.y$ in $t.y > v_{i+1}.y$ ali obratno) in je točka t na desni strani vektorja v_iv_{i+1} $(v_iv_{i+1} \times v_it > 0)$. Za ugotavljanje, na kateri strani vektorja leži določena točka, vedno uporabljamo vektorski produkt, ker predstavlja hitrejši test kot dejansko računanje presečišča.

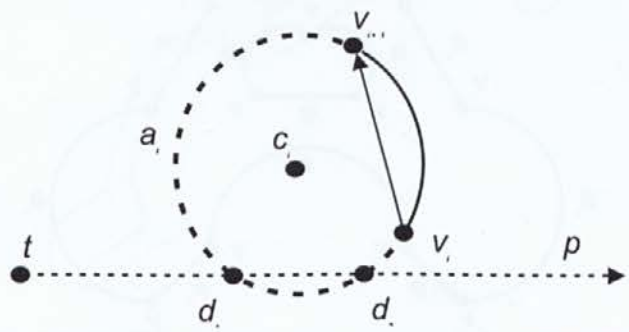
V primeru, da poltrak ne seka tetive, nam to sploh ne vpliva na rezultat, zato nam tega primera ni potrebno obravnavati. Poglejmo, zakaj.

Poltrak lahko še vedno seka preostali del krožnega loka ali pa se ga dotika. Treba bi bilo preveriti razdaljo od središča. Če je ta večja od polmera, nimamo presečišča. Če je ta manjša kot polmer, potem imamo dve presečišči. Če sta ti dve presečišči na "vidni" strani (slika 6a), obe upoštevamo, če sta na "nevidni" strani (slika 6b), pa nobenega. V obeh primerih pa to ne spremeni rezultata. Nas zanima samo sodost oz. lihost števila presečišč. Če k rezultatu prištejemo 0 ali 2, bo ostal enak. V primeru, da je razdalja enaka polmeru, imamo dotikališče, česar pa nam ni treba upoštevati, saj dotikanje ne vpliva na rezultat.

Vidimo, da smo se elegantno izognili odvečnemu testiranju in s tem pospešili algoritem.



Slika 6a: **Žarek seka »vidni« del krožnega loka**



Slika 6b: **Žarek seka »nevidni« del krožnega loka**

Podobne situacije imamo tudi v primeru, da je točka t znotraj krožnice. Točka je lahko znotraj vodoravnega pasu, ki ga določata končni točki ali izven. V primeru, da je znotraj, je število presečišč odvisno od tega, na kateri strani se nahaja "vidni" del kroga. Če je levo od tetive, nimamo presečišč (slika 7a). To pa zato, ker je potem na desni strani, torej tisti strani, v katero gre poltrak, "nevidni" del kroga. Torej se v tem primeru presečišče d_1 ne upošteva.

V nasprotnem primeru, ko je "vidni" del na desni strani, imamo seveda eno presečišče. Situacija je popolnoma enaka, le da se presečišče d_1 zdaj nahaja v "vidnem" delu kroga, ker sta "vidni" in "nevidni" del zamenjana (slika 7b).

Če pa je točka zunaj tega pasu, so ugotovitve ravno nasprotne. Če je "vidni" del na levi, imamo eno presečišče (slika 8a), v nasprotnem primeru pa ni presečišč (slika 8b).

S tem smo opisali vse možne situacije, ki lahko nastopijo pri testiranju krožnega loka. Dejanskega testiranja je zelo malo. Povzemimo zdaj vse možnosti na enem mestu (tabela 1):

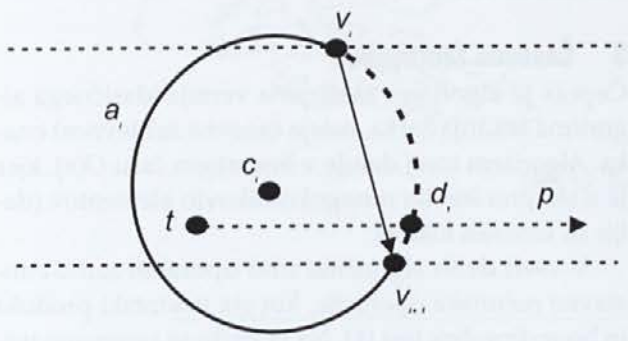
Situacija	Št. presečišč
Točka je zunaj krožnice Poltrak seka tetivo	+1
Točka je znotraj krožnice Točka je znotraj pasu tetive "Vidni" del na desni strani	+1
Točka je zunaj pasu tetive "Vidni" del na levi strani	+1

Tabela 1: **Situacije pri testiranju**

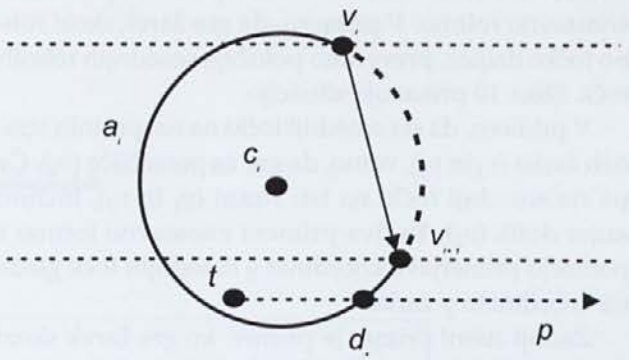
V primeru, da mnogokotnik vsebuje luknje, nam to samih pravil za testiranje ne spremeni. Vse luknje se hkrati z zanko upoštevajo pri testiranju. Tako avtomatično dobimo pravilno število presečišč.

4 Robni primeri

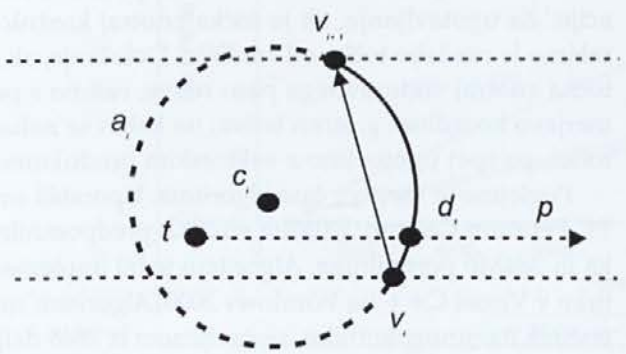
Običajen robni primer se zgodi, ko je testna točka t na mnogokotniku. To je lahko daljica ali pa krožni lok. Te situacije enostavno odkrijemo in nam ne spremenijo poteka algoritma. Če je točka na daljici, odkrijemo to s pomočjo vektorskega produkta $(v_1t_1 \times v_1v_2 = 0)$. Če je na krožnem loku, odkrijemo s pomočjo primerjave



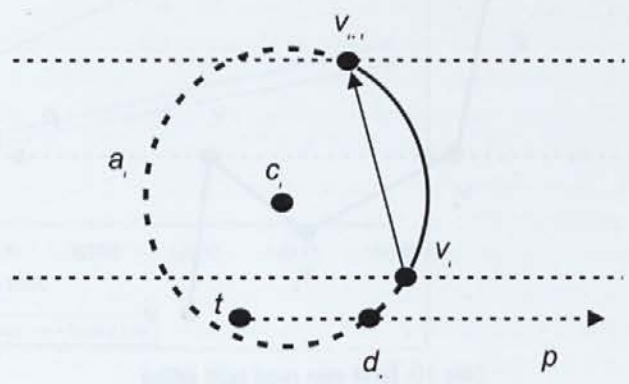
Slika 7a: **»Vidni« del krožnega loka je na levi**



Slika 8a: **Točka *t* je zunaj pasu, »vidni« del je levo**

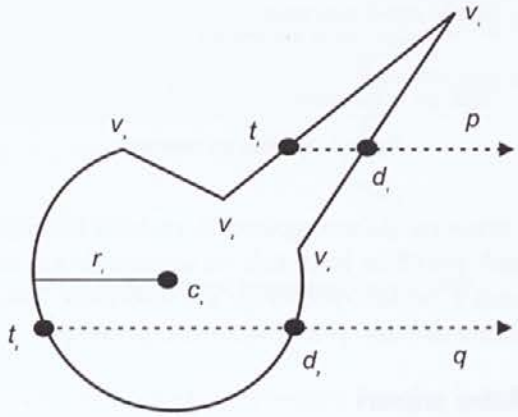


Slika 7b: **»Vidni« del krožnega loka je na desni**



Slika 8b: **Točka *t* je zunaj pasu, »vidni« del je desno**

razdalje do središča krožnega loka $d(c_1,t_2) = r_1$. Slika 9 prikazuje ta dva primera. Testna točka (t_1) je na daljici, točka (t_2) pa na krožnem loku.



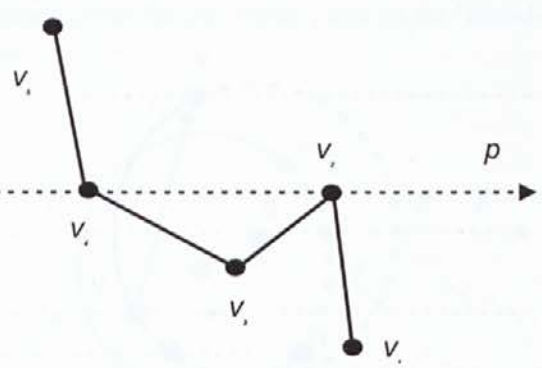
Slika 9: **Testni točki na robu mnogokotnika**

Odvisno od potrebe aplikacije se točka v takšnem primeru določi kot zunanja ali notranja.

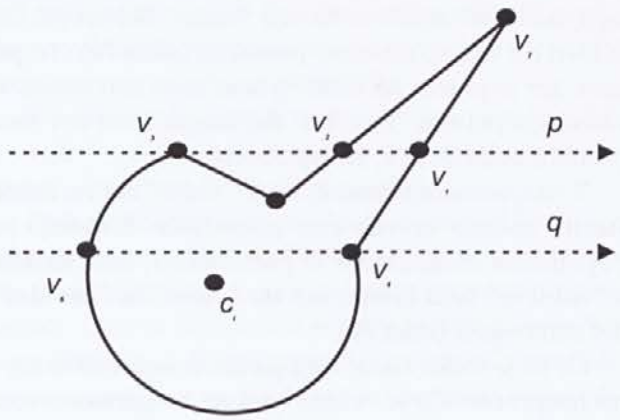
Bolj zanimivi so primeri, ko gre žarek skozi robno točko mnogokotnika. To so lahko robne točke daljic ali končne točke krožnih lokov. Te primere prav tako enostavno rešimo. V primeru, da gre žarek skozi robno točko daljice, preverimo položaja sosednjih robnih točk. Slika 10 prikazuje situacijo.

V primeru, da sta sosednji točki na nasprotnih straneh žarka (v_3 in v_5), vemo, da gre za presečišče (v_4). Če pa sta sosednji točki na isti strani (v_1 in v_3), imamo samo dotik (v_2). Ta dva primera enostavno ločimo s pomočjo primerjave koordinat y sosednjih točk glede na koordinato y žarka.

Zadnji robni primer je primer, ko gre žarek skozi končno točko krožnega loka. Slika 11 kaže dva primera. Žarek p gre skozi točko v_3, žarek q pa skozi točko v_4. V prvem primeru imamo dotik, v drugem pa presečišče. Poglejmo, zakaj.



Slika 10: **Žarek seka robni točki daljice**



Slika 11: **Žarek seka končne točke krožnega loka**

V primeru točke v_3 vidimo, da se oba sosednja dela mnogokotnika (rob v_3v_6 in krožno lok a_1) nadaljujeta na isti strani žarka. V primeru točke v_4 pa se sosednja dela (rob v_4v_1 in krožni lok a_1) nadaljujeta na različnih straneh žarka.

5 Časovna zahtevnost

Čeprav je algoritem razširjena verzija klasičnega algoritma sekanja žarka, ostaja časovna zahtevnost enaka. Algoritem torej deluje v linearnem času $O(n)$, kjer je n skupno število mnogokotnikovih elementov (daljic in krožnih lokov).

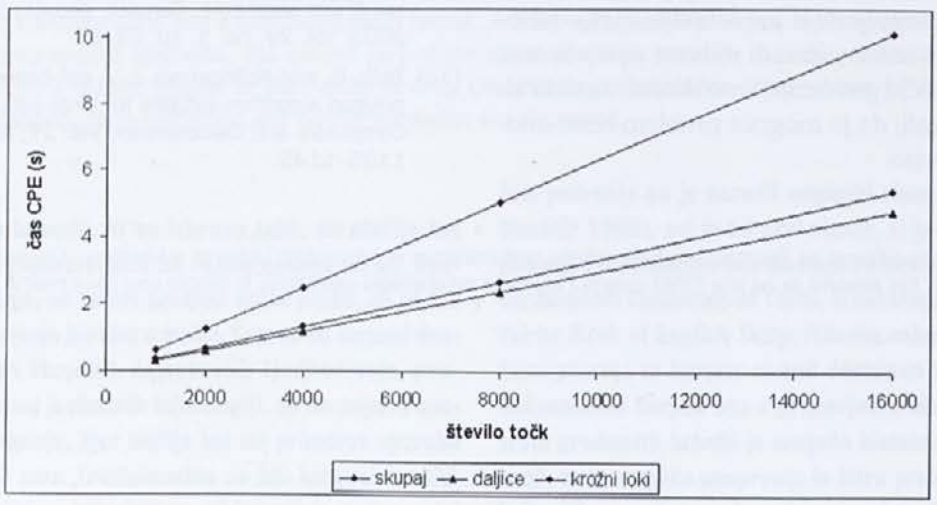
V vseh delih algoritma smo uporabili samo enostavne računske operacije, kot sta vektorski produkt in bounding-box test [1]. Na ta način se izognemo težjim računskim operacijam, saj ti dve zahtevata samo celoštevilsko množenje in seštevanje.

Pri testiranju krožnih lokov imamo podobno situacijo. Za ugotavljanje, ali je točka znotraj krožnice, rabimo le razdaljo točke od središča. Določanje, ali je točka znotraj vodoravnega pasu tetive, rešimo s primerjavo koordinat y, stran tetive, na kateri se nahaja točka, pa spet ugotovimo z vektorskim produktom.

Poglejmo še meritve časa algoritma. Uporabili smo PC Pentium Celeron 300Mhz s 128Kb predpomnilnika in 384Mb pomnilnika. Algoritem je bil implementiran v Visual C++ na Windows 2000. Algoritem smo testirali na mnogokotniku, sestavljenem iz 2848 daljic in 3334 krožnih lokov (slika 12). V tabeli 2 so podani

časi algoritma za različna števila testnih točk. Skupni čas algoritma je razdeljen tudi na čas za testiranje daljic in čas za testiranje krožnih lokov. Slika 13 kaže grafično predstavitev meritev časa.

Slika 12: **Testni mnogokotnik**



Slika 13: **Graf porabljenega časa CPE**

Št. točk	Čas daljic	Čas kr. lokov	Skupni čas
1000	0,30	0,33	0,63
2000	0,59	0,66	1,25
4000	1,18	1,32	2,50
8000	2,40	2,66	5,06
16000	4,75	5,35	10,1

Tabela 2: **Meritve časa CPE [s]**

Ugotovili smo tudi, da je razmerje med časom, porabljenim za testiranje daljic in krožnih lokov 1,1. To pomeni, da zahtevajo krožni loki okoli 10 % več procesorskega časa. Iz teh rezultatov lahko sklepamo, da so testi krožnih lokov učinkoviti, saj se skoraj enakovredno kosajo s testiranjem daljic.

6 Sklep

Predstavili smo razširjen algoritem za določanje vsebnosti točk v posplošenih mnogokotnikih, ki vsebujejo tudi krožne loke. Takšni primeri se najdejo predvsem v GIS in CAD/CAM sistemih, kjer obstajajo zelo veliki posplošeni mnogokotniki. Algoritem je sestavljen tako, da z uporabo preprostih nezahtevnih računskih operacij hitro in učinkovito reši problem.

Algoritem testira vse daljice in krožne loke na presečišče z vodoravnim žarkom. Posebej za krožne loke smo skonstruirali teste za določitev presečišča. Ti se razlikujejo od testov za daljice, vendar so še vedno dovolj hitri.

Z analizo algoritma ugotovimo, da kljub razširitvi še vedno deluje v linearni časovni zahtevnosti $O(n)$. To prikazujejo tudi meritve porabljenega časa.

Posplošeni mnogokotniki predstavljajo težjo nalogo z vsebnostnim testom. Zaradi njihove uporabnosti, predvsem v področju geodezije, smo skonstruirali ta algoritem in pokazali, da je mogoče problem rešiti enostavno in učinkovito.

7 Literatura

[1] Berg, M., M. van Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry, Algorithms and Applications*, Springer-Verlag, Berlin, 1997.

[2] Chen, M., Townsend, P., Efficient and consistent algorithms for determining the containment of points in polygons and polyhedra, *Proceedings of* Eurographics'87, 423–437, Elsevier Science, Amsterdam, 1987.

[3] Feito, F., J.C. Torres, A. Urena, Orientation, simplicity, and inclusion test for planar polygons, *Computers & Graphics*, Vol. 19, No. 4, 1995, 595–600.

[4] Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F., *Computers Graphics-Principles and Practice, 2nd ed.*, Addison-Wesley, Reading, MA, 1990.

[5] Huang, C.-W., T.-Y. Shin, On the complexity of Point-in-Polygon Algorithms, *Computers & Geosciences*, Vol. 23, No. 1, 1997, 109–118.

[6] Manber, U., *Introduction to algorithms - a creative approach*, Addison-Wesley, Reading, MA, 1990.

[7] Mortenson, M. E., *Geometric Modeling*, John Wiley & Sohns, 1985.

[8] O'Rourke, J: *Computational Geometry in C*, Cambridge University Press, 1993.

[9] Preparata, F. P. ,M. I. Shamos, *Computational Geometry An Introduction*, Springer-Verlag, 1985.

[10] Salomon, K.B., An efficient point-in-polygon algorithm, *Computers & Geosciences*, Vol. 4, No. 2, 173–175, 1978.

[11] Taylor, G., Point in polygon test. *Survey review*, Vol. 32, 1994, 479–484.

[12] Žalik, B., Gomboši, M., Podgorelec, D., A Quick Intersection Algorithm for Arbitrary Polygons, *Proceedings of the Spring Conference on Computer Graphics SCCG 1998*, Budmerice, Slovakia, April 23–25, 1998.

[13] Žalik, Borut, Zadravec, Mirko, Clapworthy, Gordon J. Construction of a non-symmetric geometric buffer from a set of line segments. *Comput. geosci.*. ŠPrint ed.Ć, 2003, vol. 29, no. 1, str. 53–63.

[14] Žalik, B. and Kolingerova, I., A cell-based point-in-polygon algorithm suitable for large sets of points, *Computers and Geosciences*, Vol. 27, No. 10, 2001, 1135–1145.

Mag. Matej Gomboši je asistent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Diplomiral je leta 1999 in magistriral leta 2002. Kot asistent se od leta 1999 ukvarja z algoritmi računalniške geometrije in njihovo uporabo v realnih problemih.

Korpus kot podpora slovarju informacijskega izrazja slovenskega jezika

Tomaž Erjavec
Odsek za tehnologije znanja, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana
tomaz.erjavec@ijs.si
Špela Vintar
Oddelek za prevajalstvo, Filozofska fakulteta, Aškerčeva 2, 1000 Ljubljana
spela.vintar@guest.arnes.si

Povzetek

Prispevek predstavi uporabo zbirke besedil (jezikovnega korpusa) pri izdelavi terminološkega slovarja. Spletni slovar informacijskega izrazja slovenskega jezika nastaja pri jezikovni sekciji Slovenskega društva informatika (SDI), društvo pa organizira tudi letne konference »Dnevi slovenske informatike« (DSI) s tiskanimi zborniki. V prispevku najprej predstavimo slovar, nato pa se osredotočimo na izgradnjo korpusa s področja informatike, ki trenutno zajema zbornik konference DSI 2003. Izdelava korpusa temelji na uporabi tehnologij XML in je sestavljena iz pretvorbe prispevkov v zborniku iz izvornega zapisa (Microsoft Word) v osnovni zapis XML, nato pa v obliko, primerno za spletno iskanje. Manjši del korpusa je dvojezični in vsebuje slovenske in angleške stavčno poravnane povzetke prispevkov. Izvorni namen izdelave korpusa DSI je slovaropisni, saj bi z njim po eni strani želeli sodelujočim olajšati izdelavo slovarja SDI, po drugi strani pa ponuditi uporabnikom dodatni vir primerov za iskani termin. V članku opišemo postopke izdelave korpusa in računalniško podprtega iskanja izrazov, pri katerem so sodelovali tudi študentje prevajalstva na Filozofski fakulteti Univerze v Ljubljani. Prispevek obravnava tudi načrte za nadaljnje delo, ki poleg razširitve korpusa predvidevajo tudi oblikoskladenjsko označevanje in lematizacijo besed v korpusu ter avtomatsko luščenje področnih terminov.

Abstract

A Corpus-driven Approach to Building the dictionary of Information Science

The paper describes the exploitation of a text corpus for the compilation of the terminological dictionary of information science, which is being created within the language section of the Slovenian Society of Information Science (SDI). Among its other activities, the Society organizes yearly meetings under the title »Days of Slovenian Information Science« (DSI) with printed proceedings. The first part of the paper presents the web dictionary and the process of building the corpus of information science, which at present contains the proceedings of the conference DSI 2003. Building the corpus included several stages, such as conversion of the original Word files into XML and transformation into a web-searcheable format. A small section of the corpus is bilingual and consists of English and Slovene sentence-aligned abstracts. The second part of the paper describes methods of corpus-based terminography, which were employed within a student project at the Faculty of Arts, Department of Translation. Finally, plans for future work, including deeper linguistic tagging, term extraction and corpus expansion are discussed.

1 UVOD

Korpus je zbirka besedil, ki so izbrana tako, da služijo kot vzorec za stanje, raznovrstnost ali razvoj nekega jezika. Uporaben je kot podlaga, na kateri gradimo opise jezika, ali pa kot sredstvo za preverjanje hipotez o jeziku. Čeprav so korpusi dandanes koristni pri številnih dejavnostih (jezikoslovje, poučevanje jezika, razvoj jezikovnih tehnologij), so se najprej uporabljali pri slovaropisju, kjer služijo kot vir primerov uporabe besed in besednih zvez. Tradicionalno so bili korpusi hranjeni na papirju (tipično v obliki listkov, od katerih je vsak navajal primere uporabe ene slovarske iztočnice), bistven premik na tem področju pa je naredil angleški slovar Cobuild (opisan v Sinclair 1998), saj je bil prvi slovar, ki je nastal izključno na podlagi računalniško hranjenega referenčnega korpusa The Birmingham Collection of Texts, iz katerega je pozneje nastala zbirka Bank of English (http://titania.cobuild.collins.co.uk/; v času pisanja ta korpus ni več dostopen za spletno iskanje. Računalniški korpus ima v primerjavi s klasičnim »papirnim« vrsto prednosti: hraniti je mogoče bistveno večjo količino besedil, ta je mogoče preprosto in hitro preiskovati po različnih kriterijih, rezultate poizvedb pa predstaviti prilagojeno specifičnemu namenu.

Od časov Cobuilda je uporaba računalniških korpusov pri izdelavi slovarjev postala že standardna praksa, ki se je z angleškega razširila tudi na jezike z manjšim številom govorcev. Za slovenski jezik je tak primer korpus FIDA (http://www.fida.net/), ki ga pri DZS, d. d., uporabljajo za izdelavo nove generacije slovarjev, na Filozofski fakulteti za jezikoslovne raziskave, na IJS in pri podjetju Amebis, d. o. o., pa za razvoj jezikovnih tehnologij. Bank of English in FIDA spadata med t. i. referenčne korpuse, katerih cilj je čim bolje vzorčiti celotno produkcijo nekega jezika. Referenčni korpusi so tipično zelo veliki (FIDA ima sto milijonov besed) in vsebujejo veliko število besedil (FIDA prek 20.000), ki so izbrana po skrbno uravnovešeni mreži kriterijev, s katero je na primer določeno razmerje med leposlovjem in strokovno literaturo, monografijami in periodiko, izvorno in prevodno literaturo itd. Poleg referenčnih korpusov pa poznamo tudi specializirane korpuse, ki so usmerjeni samo v določen segment jezika oz. njegove uporabe, npr. jezik najstnikov (COLT) ali pa jezik poizvedb po letalskih poletih. Takšen korpus, tj. specializirani korpus informatike slovenskega jezika, bo tudi predmet tega članka.

Uporabnost nekega korpusa je odvisna od njegove velikosti pa tudi urejenosti, tj. kako podrobno je dokumentiran in označen, ter standardiziranosti njegovega zapisa. Dokumentiranost omogoča vpogled v vire, ki so bili uporabljeni za izgradnjo korpusa, kakšni uredniški posegi so bili narejeni nad temi viri, oznake, ki so uporabljene v korpusu itd. Označenost korpusa, npr. oblikoskladenjska na ravni stavka, besede, z oblikoskladenjskimi oznakami itd., omogoča bogatejše izkoriščanje korpusnega materiala, saj lahko po njem iščemo po bolj abstraktnih kategorijah, npr. »najdi vse pojavitve leme "aplikacija", pred katero stoji pridevnik«. Standardiziranost zapisa pa doprinese k izmenljivosti korpusa, tako med ljudmi kot med aplikacijami, in k neodvisnosti od konkretnih računalniških platform, s tem pa tudi k večji trajnosti. Standardiziranost dandanes pomeni v prvi vrsti zapis v skladu z XML – eXtended Markup Language (W3C 2000), saj je to edini ustrezni standard za zapis digitalnih besedil, ki je tudi široko podprt v programski opremi in pridruženih standardih. V nadaljevanju se vrnemo k tem temam in opišemo naše rešitve pri izgradnji korpusa.

V prispevku predstavimo jezikovni korpus, ki služi kot podpora pri izdelavi spletnega slovarja informacijskega izrazja slovenskega jezika, nastajajočega pri jezikovni sekciji Slovenskega društva Informatika (SDI), ter uporabljene korpusno-terminološke metode za pridobivanje izrazja. V drugem poglavju na kratko predstavimo slovar, tretje poglavje opiše izdelavo našega korpusa, njegov zapis, možnosti nadaljnjega označevanja ter mrežni konkordančnik, s katerim lahko iščemo po korpusu, četrto poglavje opiše terminološko delo, ki smo ga v preskusne in izobraževalne namene izvajali s študenti prevajalstva, peto poglavje pa poda nekaj sklepov in načrte za nadaljnje delo.

2 SLOVAR SLOVENSKEGA DRUŠTVA INFORMATIKA

Slovensko društvo Informatika je v okviru svoje jezikovne sekcije leta 2001 začelo z delom na spletnem slovarju informacijskega izrazja slovenskega jezika, na kratko »Slovar informatike«. Slovar, ki se nahaja na naslovu http://www.ef.uni-lj.si/terminoloskislovar/, je namenjen vsem članom društva in široki javnosti. V slovarju se zbirajo temeljni in najsodobnejši informacijski izrazi, ki se uporabljajo v znanosti, v strokovni javnosti in med uporabniki. Pomagal naj bi pojasnjevati pomen strokovnih pojmov vsem, ki se srečujejo z informatiko, pa tudi pri ustvarjanju znanstvenih del, pri pisanju strokovnih besedil in pri komuniciranju z uporabniki.

Slovar se sproti dopolnjuje neposredno na spletu. Pri njegovem oblikovanju sodelujejo številni strokovnjaki kot uredniki področij, strokovni sodelavci, svetovalci ali kot člani sekcije. Za zdaj ima opredeljenih 16 področij informatike, npr. internet, poslovna informatika, varovanje informacijskih sistemov, naprave (strojna oprema), sociološki vidiki, odprti sistemi itd. Terminološki slovar informatike je razlagalni in informativni slovar, ki strokovno izrazje pomensko in jezikovno opisuje, vrednoti in kateremu so dodani angleški ustrezniki. Slovarski sestavek oblikujejo iztočnica (enobesedni ali večbesedni izraz), besednovrstna in stilska oznaka, ustreznik v angleškem jeziku in razlaga, ki jo lahko podkrepijo sinonimi in vsebinsko povezani pojmi, ki so obravnavani v slovarju. Za dokončno vsebino in oblikovanje slovarja so zadolženi uredniki. Ti izraze in razlage preverjajo glede na že obstoječe, objavljeno izrazje, pa tudi v skladu s pravili slovenskega jezika.

Uporabniki slovarja lahko iščejo slovenske ali angleške besede, lahko pa nove slovenske besede ali prevode tudi vpisujejo. Slovar se naslanja na angleške izraze, zato vpis slovenskega izraza brez angleškega

ni mogoči. Vsi izrazi, ki jih uporabniki ne najdejo, se beležijo in po presoji uredništva vnašajo v slovar. Slovar zajema samo informacijsko izrazje, besed splošnega pomena ne vsebuje.

3 KORPUS DNEVOV SLOVENSKE INFORMATIKE

Slovensko društvo Informatika organizira letne konference »Dnevi slovenske informatike« (DSI) s tiskanimi zborniki. Ker zborniki pokrivajo isto področje kot slovar, obenem pa so znanstveni prispevki dragocen vir svežega slovenskega izrazja, se je pojavila ideja, da se zbornike pretvori v korpus, ki bi nato lahko služil kot podpora pri izdelavi, pa tudi uporabi slovarja.

Vir za izdelavo korpusa, ki ga predstavimo v tem razdelku, so digitalni izvirniki posameznih prispevkov (torej brez predgovorov in drugega spremnega besedila v zborniku), ki so služili kot predloga tiskanemu zborniku za leto 2003. Pri izdelavi korpusa smo izhajali iz določenih standardov; tako za zapis korpusa uporabljamo XML (W3C 2000), za pretvorbe pa pridruženi standard XSLT (W3C 1999). Če bo korpus v prihodnosti prerasel svojo sedanjo namembnost in velikost, načrtujemo tudi prilagoditev korpusa priporočilom Iniciative za zapis besedil TEI – Guidelines for Text Encoding and Interchange (Sperberg-McQueen and Burnard, 2002).

3.1 Opis vira

Triindevetdeset člankov, ki so služili kot osnova korpusu, je zapisanih v formatu Microsoft Word, pri čemer je stil predpisan s strani SDI. Stil je sicer podan opisno, spletne strani z navodili za avtorje pa ponujajo tudi primer pravilno oblikovanega članka. Predloga ponuja poleg standardnih tudi svoje stile, ki definirajo nekatere strukturno pomembne dele članka, kot so npr. naslov, avtorji, njihovi naslovi, slovenski in angleški povzetek itd. Uporaba takšnih stilov zelo olajša pretvorbo v korpus, žal pa jih avtorji niso upoštevali, čeprav je v urejevalniku word mogoče doseči isto podobo besedila z različnimi prijemi.

3.2 Pretvorba v XML

Za pretvorbo oblike Microsoft Word v XML obstaja razmeroma bogata ponudba večinoma komericalnih programov. Mi smo izbrali program UpCast (http://www.infinity-loop.de/), ki v prosto dostopni »osebni licenci« ponuja polno funkcionalnost pri pretvorbi dokumentov, je pa potrebno za vsak dokument posebej sprožiti pretvorbo. To delo smo zaupali študentom prevajalstva, ki so v okviru predmeta korpusi in saj podatkov v tretjem letniku dodiplomskega študija izdelovali korpuse za terminografske namene.

```
<article>
  <para role="naslov_prispevka">JEZIKOVNI VIRI SLOVENSKEGA
    STROKOVNEGA JEZIKA</para>
  <para role="avtor">Tomaž Erjavec</para>
  <para role="avtor_naslov">Odsek za inteligente sisteme, Institut "Jožef
    Stefan", Jamova 39, 1000 Ljubljana</para>
  <para role="avtor_naslov">tomaz.erjavec@ijs.si</para>
  <para role="povzetek_naslov">Povzetek</para>
  <para role="povzetek">Prispevek predstavi področje jezikovnih tehnologij,
    metod, ki olajšajo uporabo jezika v ...</para>
  <para role="abstract_title">Abstract</para>
  <para role="abstract">
    <phrase>LANGUAGE RESOURCES FOR SLOVENE TECHNICAL
      LANGUAGE</phrase>
  </para>
  <para role="abstract">The paper discusses the field of Language
    Technologies, i.e. methods that ...</para>
  <section>
    <title>
      <phrase role="upcast-HEADINGNUMBER">1.</phrase>
      UVOD
    </title>
    <para role="Normal">
      Prispevek predstavi po
      <phrase>dročje jezikovnih tehnologij: metod, ki ...
```

Slika 1: **Primer pretvorbe iz Worda v XML z orodjem upCast**

```
<s><w>Večina</w> <w>sorodnih</w> <w>člankov</w><c>,</c> <w>ki</w>
<w>smo</w> <w>jih</w> <w>zasledili</w><c>,</c> <w>obravnava</w> <w>le</w>
<w>algoritme</w> <w type="abbr">oz.</w> <w>postopke</w> <w>za</w>
<w>razvrščanje</w> <w>besedil</w> <w>kot</w> <w>v</w> <w>članku</w>
<c type="open">[</c><w type="dig">15</w><c type="close">]</c> <w>ali</w>
<c type="open">[</c><w type="dig">14</w><c type="close">]</c><c>.</c></s>
```

Slika 2: **Primer stavka iz korpusa DSI**

UpCast ponuja izhod v lastnem tipu dokumentov XML, eksperimentalno pa tudi v zapisu DocBook (http://www.docbook.org/), ki se sicer uporablja predvsem za zapis računalniške dokumentacije, je pa dovolj pregleden pa tudi dobro dokumentiran; zapis začetka enega od prispevkov v tem izhodnem formatu ilustriramo v sliki 1.

Kot vidimo, ima format precejšnje število koristnih podatkov, čeprav ni brez pomankljivosti, tako je npr. v naslovu nepojasnjeno ena od črk mala, v besedilu pa se brez prave logike pojavi element <phrase>. Kakorkoli že, s to pretvorbo preidemo v standardiziran in poenoten format (XML DocBook), ki je z uporabo primernega stila – vsaj teoretično – še vedno prikazljiv enako kot original in torej ni izgubil informacije.

3.3 Jezikovno označevanje

Po pretvorbi v enoten zapis TEI lahko zbirko besedil že poimenujemo korpus, saj je uniformno in standardizirano zapisan. Seveda pa se s tem prava jezikovna analiza besedila šele začne. Kaj točno hočemo v korpusu označiti, je v veliki meri odvisno od namembnosti. Osnovna koraka, ki sta vedno koristna, sta označitev besed in stavkov v besedilu, t. i. tokenizacija in segmentacija. Čeprav že ta stopnja označevanja skriva določene pasti (pika npr. ne označuje vedno konca stavka), pa v splošnem ni preveč zahtevna – to je tudi stopnja, do katere smo trenutno označili korpus DSI, primer stavka iz korpusa pa podamo v sliki 2.

Naslednja faza, ki je pogosto koristna, je t. i. oblikoskladenjsko označevanje (Van Halteren, 1999): tu vsaki besedi v korpusu pripišemo njene oblikoskladenjske oznake, npr. »samostalnik moškega spola v rodilniku ednine«, dostikrat pa tudi leme oz. gesla, npr. za besedo »berači« lemo »beračiti«. Za takšno označevanje je potrebno najprej imeti slovar ali pa program, ki za besedne oblike določi vse možne oblikoskladenjske oznake in po možnosti pripadajoče leme. Neka besedna oblika ima v slovarju ponavadi več možnih interpretacij, tako je npr. »berači« lahko glagol v velelniku ali povedniku ali samostalnik v imenovalniku ali orodniku množine. V konkretnem besedilu pa bo besedna oblika imela seveda samo eno ustrezno oznako. Naloga programov za oblikoskladenjsko označevanje je izmed možnih oblikoskladenjskih oznak neke besede določiti glede na sobesedilo, njeno pravo oznako.

Izdelanih je bilo že veliko označevalnikov, ki se lahko naučijo zakonitosti nekega jezika iz ročno označenih korpusov. Ena bolj odmevnih metod z uporabo t. i. skritih markovskih verig določi najbolj verjetno zaporedje oblikoskladenjskih oznak besed v nekem stavku glede na njihov lokalni kontekst. Za angleški jezik dosežejo takšni označevalniki ob uporabi zadosti velike učne množice približno 96-odstotno natančnost. Za slovanske jezike, ki imajo precej bogatejše oblikoslovje in s tem večje število možnih oznak, je ta natančnost manjša, predvsem pa odvisna od velikosti učnega korpusa. Pri lastnih poskusih (Džeroski et al., 2000) smo dosegli natančnost reda 92 %. Kot primer rezultata tokenizacije, segmentacije in oblikoskladenjskega označevanja podamo v sliki 3 stavek iz korpusa MULTEXT-East (Erjavec, 2004).

3.4 Stavčna poravnava

Kot je v navadi za večino strokovnih publikacij, morajo tudi prispevki srečanja DSI vsebovati povzetek v

```
<s id="Osl.1.2.2.1">
<w lemma="biti" ana="Vcps-sma">Bil</w>
<w lemma="biti" ana="Vcip3s--n">je</w>
<w lemma="jasen" ana="Afpmsnn">jasen</w>
<c>,</c>
<w lemma="mrzel" ana="Afpmsnn">mrzel</w>
<w lemma="aprilski" ana="Aopmsn">aprilski</w>
<w lemma="dan" ana="Ncmsn">dan</w>
<w lemma="in" ana="Ccs">in</w>
<w lemma="ura" ana="Ncfpn">ure</w>
<w lemma="biti" ana="Vcip3p--n">so</w>
<w lemma="biti" ana="Vmps-pfa">bile</w>
<w lemma="trinajst" ana="Mcnpnl">trinajst</w>
<c>.</c>
</s>
```

Slika 3: **Stavek iz korpusa MULTEXT-East**

slovenskem in angleškem jeziku. Iz teh povzetkov je torej mogoče oblikovati vzporedni korpus, ki je za terminografske namene tudi najbolj uporaben tip korpusa. Ker je bil v izvirnih dokumentih za povzetka uporabljen poseben slog, smo povzetke iz besedil izluščili avtomatsko s pomočjo ustreznih oznak XML.

Stavčna poravnava je postopek, pri katerem se vsaki stavčni enoti izvirnika priredi ustrezna enota v prevodu. Postopek je delno avtomatiziran in ga zna opraviti tako rekoč vsak prevajalski program, vendar je rezultate samodejne poravnave navadno treba ročno pregledati in popraviti. Poravnava je tako predstavljala eno od študentskih opravil, zanjo pa smo uporabili prevajalski program DejaVu proizvajalca Atril (http://www.atril.com).

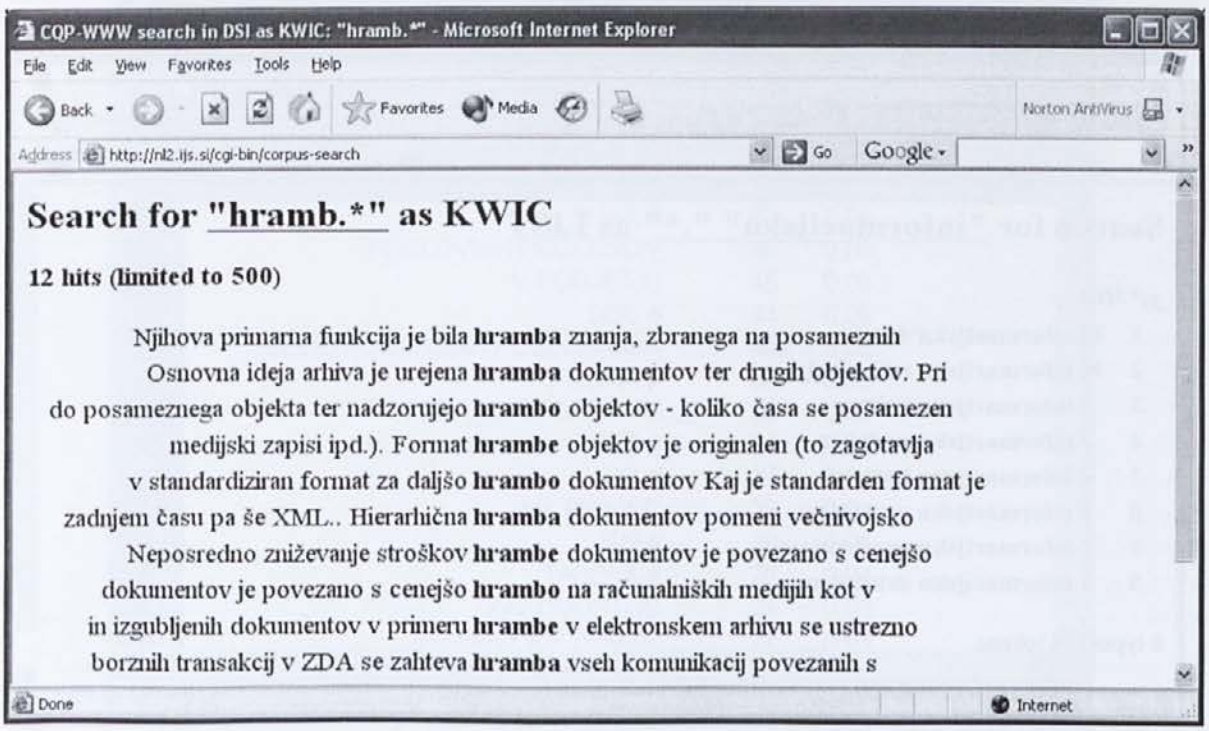
Postopek poravnave ustvari vzporedno besedilo, ki je na voljo bodisi v obliki dvostolpčne tabele v programu DejaVu, lahko pa ga izvozimo v MS Excel ali v besedilno datoteko, kjer sta izvirni in prevodni segment med seboj ločena s posebnim znakom, na primer s tabulatorjem.

3.5 Konkordance

Ko je korpus narejen, ga je seveda potrebno dati na razpolago. V našem primeru ciljno skupino uporabnikov, vsaj v prvi fazi, sestavljajo avtorji oz. uredniki slovarja, ki bi jim korpus pomagal pri preverjanju hipotez o slovenskih terminih. Za takšno delo se uporabljajo t. i konkordančniki, programi, ki prikažejo neko besedo ali besedno zvezo v vseh pojavitvah v korpusu skupaj s sobesedilom. Konkordančnik je temeljno orodje sodobnih slovaropiscev, saj ilustrira uporabo (in s tem posredno tudi pomene) iskanih besed ali besednih zvez. Poizvedovalni jeziki konkordančnikov so lahko precej bogati in obsegajo regularne izraze nad nizi, poizvedovanje glede na oznake ter logične operatorje.

V Sloveniji obstaja že večje število mrežnih konkordančnikov, na primer za referenčna korpusa FIDA (http://www.fida.net/) in Nova beseda (http://bos.zrc-sazu.si/) in za slovensko-angleški Evrokorpus (http://www.gov.si/evrokorpus/). Slika 4 prikaže izpis na poizvedbo v konkordančniku IJS; to orodje je dostopno na http://nl2.ijs.si/, ki ponuja večje število korpusov, sedaj tudi korpus DSI.

Konkordančnik IJS je v uporabi več kot štiri leta, uporabljajo pa ga predvsem prevajalci in študentje prevajanja. Kot hrbtenico uporablja IMS Corpus Workbench (CWB, http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/), program za linux, ki je sposoben po kompleksnih kriterijih hitro iskati po



Slika 4: **Izpis enojezične konkordance**

velikih korpusih. CWB je nato prek skripta CGI postavljen na mrežo s HTTP strežnikom Apache. Poizvedovanje po izbranem korpusu poteka kar v iskalnem jeziku, ki ga ponuja CWB, ali pa v poenostavljenem načinu (npr. »info*«), ki se nato avtomatsko prevede v bolj kompleksno sintakso CWB (»"info.*"«). Izbrati je možno več načinov izpisa: poleg besede v kontekstu še vzporedni prikaz, primeren za dvojezične korpuse, in izpis golega seznama zadetkov, koristen npr. za iskanje sopojavnic – primer je podan v sliki 5.

Da mrežnemu konkordančniku dodamo nov korpus, kot smo to storili z DSI, je potrebno le-tega pretvoriti v vhodni format (kar je iz XML-ja z XSLT enostavno), ga tam indeksirati in dodati novo izbiro v katalog, skripto CGI ter krovno stran iskalnika.

Trenutno se korpus DSI prek mrežnega konkordančnika uporablja za iskanje novih terminov in preverjanja njihove uporabe s strani registriranih avtorjev slovarja. V bodoče nameravamo tudi dodati povezavo na neposredno iskanje terminov kot možnost za uporabnike slovarja.

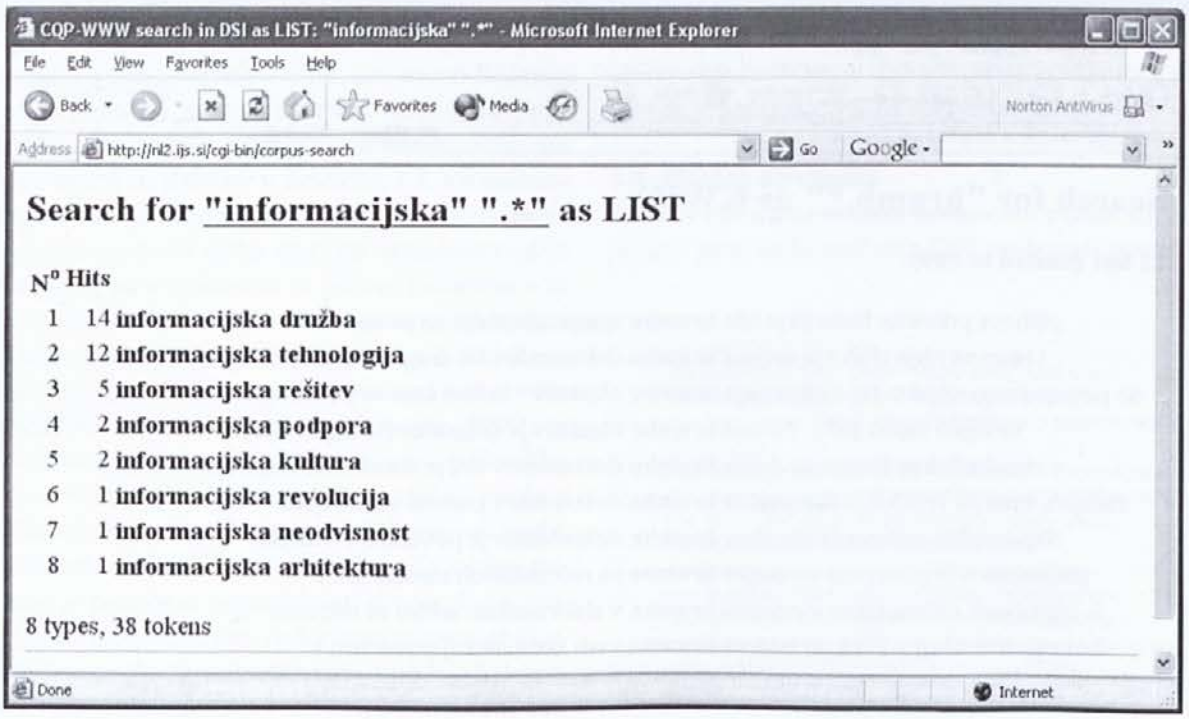
4 METODE KORPUSNO PODPRTE TERMINOGRAFIJE

V okviru študija prevajalstva že tretje leto poteka seminar korpusi in baze podatkov, pri katerem se študenti seznanijo z izdelavo korpusa za terminološke namene ter izdelujejo terminološke baze za različna področja. Jeseni 2003 je bila vzpostavljena naveza sodelovanja z društvom SDI, ki je pripeljala tudi do zamisli o sodelovanju študentov pri gradnji korpusa DSI in iSlovarja. Pred pričetkom dela je urednica slovarja Katarina Puc študentom predstavila značilnosti projekta, nato pa se je desetčlanska skupina študentov posvetila področju informatike.

Naloga je zajemala pretvorbo Wordovih datotek v XML z že opisanim orodjem UpCast, izdelavo dvojezičnega korpusa DSI s poravnavo, izbor gesel na podlagi obdelave s programom Wordsmith, slovarsko obdelavo gesel in nazadnje vnos obdelanih gesel v terminološki program TRADOS MultiTerm (http://www.trados.com). Ker smo prva dva koraka podrobno opisali že v prejšnjih razdelkih, se tu osredotočamo na postopke obdelave korpusa za terminološke namene.

4.1 Orodje Wordsmith

Čeprav je bil v času študentskega projekta že na voljo tudi mrežni konkordančnik, smo tu namesto njega uporabljali druga orodja, ki prvič omogočajo delo tudi brez internetne povezave, kar je za nekatere študente še vedno pomemben dejavnik, drugič pa poleg iskanja po



Slika 5: **Izpis v obliki seznama**

besedilih omogočajo tudi druge jezikovnotehnološke obdelave.

Pri ugotavljanju, katere besede ali besedne zveze so terminološko relevantne, si lahko pomagamo z orodjem Wordsmith (http://www.lexically.net), ki poleg brskanja po besedilni zbirki in izpisa konkordanc nudi še številne druge funkcije, na primer izdelavo besednih seznamov po pogostosti ali abecedi, in sicer posameznih besed ali večbesednih skupkov. Pri izdelavi seznamov je mogoče samodejno izločiti t. i. prazne besede, kot so vezniki, pomožni glagoli in podobno, s komponento Keywords pa lahko primerjamo dobljeni besedni seznam z referenčnim korpusom in ugotovimo, katere besede so s svojo relativno pogostostjo tipične za obravnavano stroko. Kot vsak boljši konkordančnik zna tudi Wordsmith izračunati kolokacije, pa tudi grafično prikazati distribucijo posameznega izraza v korpusu.

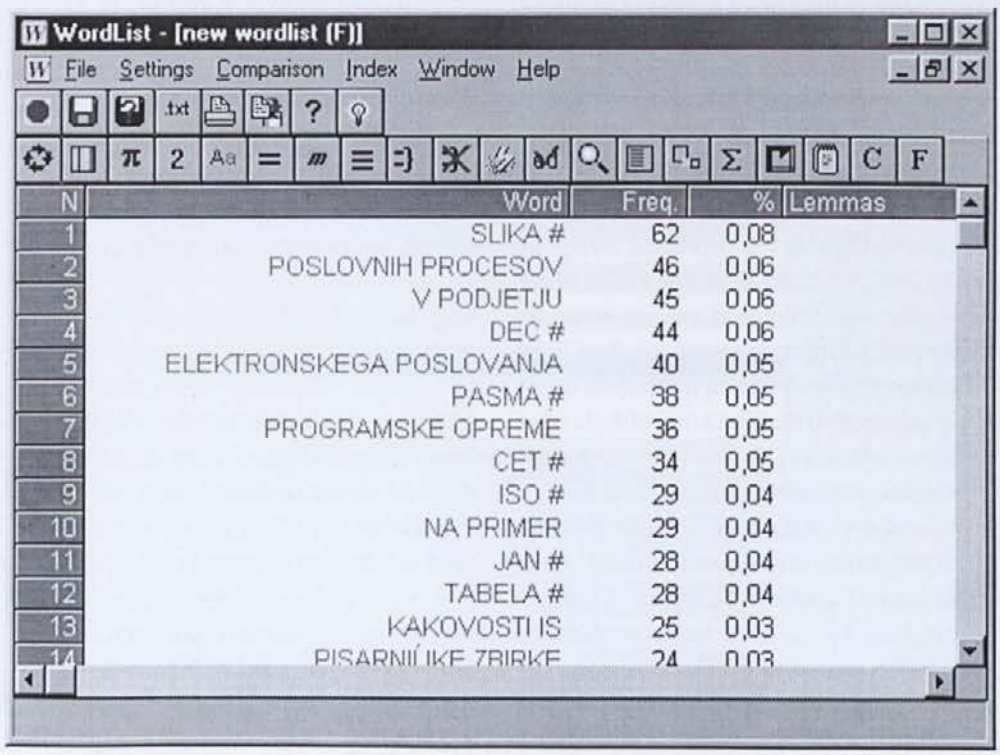
Pri zbiranju gesel si torej lahko precej pomagamo z različnimi besednimi seznami eno- in večbesednih enot ter ključnih besed. V našem primeru sta bila poglavitna kriterija za izbiro gesel pogostost in pa preverba, da geslo še ni vnešeno v spletni iSlovar. Slika 6 kaže izsek seznama pogostih dvobesednih enot v programu Wordsmith.

Program je sposoben obdelovati tudi besedila v zapisu XML ali SGML. Morda je še največja pomanjkljivost programa, da ne zna zadovoljivo obdelovati vzporednih korpusov; prikaz dvojezičnih konkordanc namreč ni mogoč. Kadar imamo opravka z dvema jezikoma, nam lahko Wordsmith pomaga le pri statistični primerjavi leksikalne gostote, povprečne dolžine odstavkov, stavkov in besed v posameznem jeziku.

Za iskanje po vzporednih korpusih so sicer tudi na voljo različna orodja, vendar je bil v našem primeru vzporedni korpus le pomožni vir izrazja, obenem pa je bilo zaradi njegove majhnosti možno po njem iskati tudi na »enojezični način«, se pravi s prikazom iskanega niza in prevedenega segmenta v isti vrstici. Za enostavno dvojezično iskanje je primeren tudi prej omenjeni program DejaVu.

4.2 Termini, razlage in kolokacije

Čeprav je informatično izrazje večinoma angleškega izvora in je privzeta smer iSlovarja angleško-slovenska, smo v našem primeru zaradi sestave korpusa izhajali iz slovenskih iztočnic, ki smo jim v drugi fazi iskali ustreznice. Ne glede na dobro računalniško podporo je izbor gesel še vedno najbolj težavna in potencialno sporna naloga v slovaropisju. Kot večina področij je



N	Word	Freq.	%	Lemmas
1	SLIKA #	62	0,08	
2	POSLOVNIH PROCESOV	46	0,06	
3	V PODJETJU	45	0,06	
4	DEC #	44	0,06	
5	ELEKTRONSKEGA POSLOVANJA	40	0,05	
6	PASMA #	38	0,05	
7	PROGRAMSKE OPREME	36	0,05	
8	CET #	34	0,05	
9	ISO #	29	0,04	
10	NA PRIMER	29	0,04	
11	JAN #	28	0,04	
12	TABELA #	28	0,04	
13	KAKOVOSTI IS	25	0,03	
14	PISARNIŠKE ZBIRKE	24	0,03	

Slika 6: **Orodje Wordsmith**

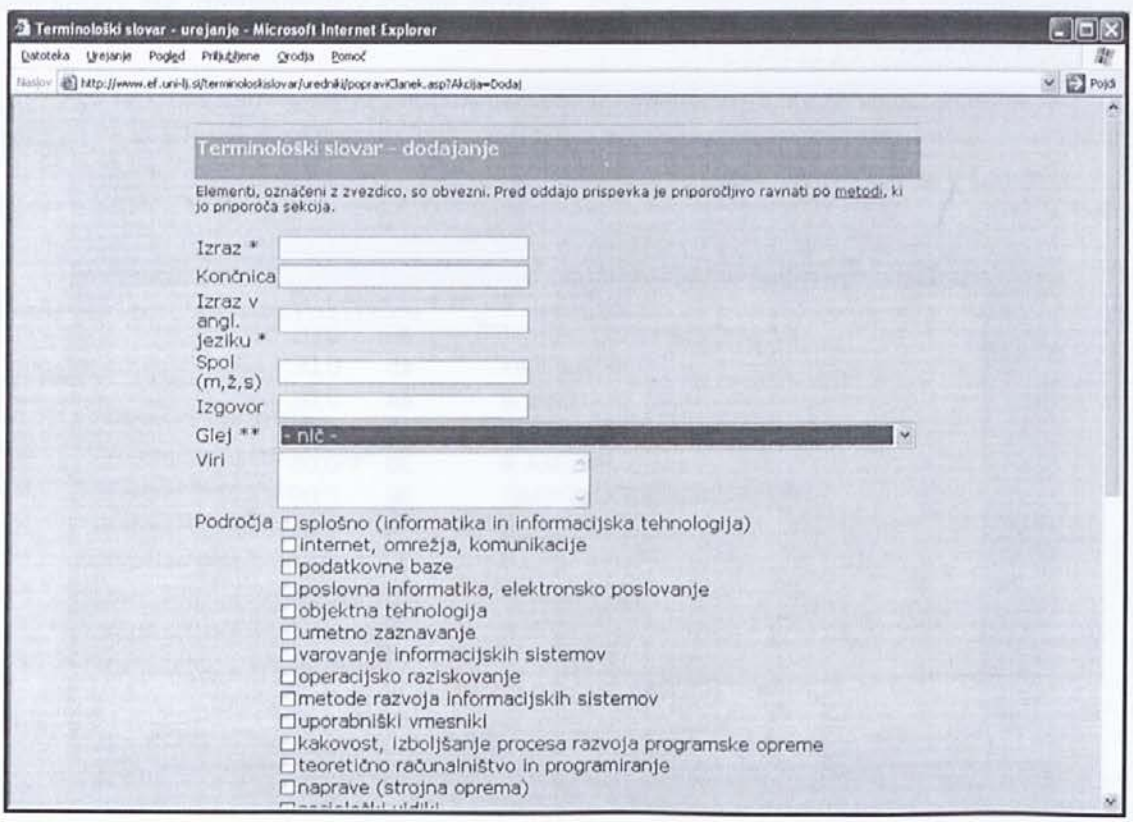
namreč tudi informatika interdisciplinarna, tako da v njej srečujemo gostujoče termine s številnih področij. Na posvetu DSI 2003 je bilo precej prispevkov namenjenih e-poslovanju, zato so bili izrazi s področja (e-)ekonomije zelo pogosti, na primer *poslovni proces, poslovni sistem, poslovna aplikacija* itd.

Študenti so imeli precej težav pri razlikovanju med terminološkimi kolokacijami, ki so se pojavljale pri vrhu Wordsmithovih besednih seznamov, in pravimi termini. Tako se na primer pojavi izraz *language technology application*, ki je verjetno kompozitivna kolokacija, kjer se pomen sestavi iz *language technology* in *application*. Čeprav smo v začetku izhajali iz načela, da bomo nediskriminatorno med gesla uvrščali tudi glagolsko izrazje in druge nesamostalniške zveze, se kmalu pokaže, da se nam glagoli kljub pogostosti in specifičnemu pomenu mnogokrat ne zdijo primerni za uvrstitev med iztočnice. Vsaj tisti, ki najbolj odstopajo od svojega splošnojezikovnega pomena, na primer *shraniti, brskati*, bi si zagotovo zaslužili terminološko obdelavo.

Med izrazi, ki so jih študenti izbrali za uvrstitev v bazo, so se znašli tudi precej splošni izrazi, ob katerih se postavlja vprašanje meje med splošnih in strokovnim besediščem, na primer *declaration – deklaracija, communication channel – komunikacijski kanal* itd. Prvi je denimo razložen kot *del uvoda kakega dokumenta*, kar je za pomensko umestitev v svet informatike zagotovo premalo, vendar bi z bolj računalniško razlago izraz lahko utemeljeno uvrstili med iztočnice in mu dodali bolj specifične podpomenke, na primer *deklaracija spremenljivk*.

Ko je bil izbor gesel končan, se je začela obdelava, se pravi opremljanje iztočnic s čim bogatejšimi slovarskimi podatki. Tu smo se navezali na obstoječo strukturo iSlovarja, ki pod posamezno iztočnico predvidi podatkovna polja izraz, končnica, izraz v angleškem jeziku, spol, izgovor, glej, viri, področje, razlaga, sinonimi in viri. Masko za spletno vnašanje, ki je dostopna le urednikom slovarja, prikazuje slika 7.

Zgornja shema ima sicer na voljo precej polj, vendar nekaterih podatkovnih kategorij ne ločuje dovolj jasno. Tako niti iz sheme niti iz pojasnila metode na spletni strani ni razvidna razlika med istoimenskima poljema Viri in razlika med polji Glej in Sinonimi. Na spletni strani je sicer podana opomba, da se pod Glej



Slika 7: **Maska za vnašanje gesel v iSlovar**

vnaša sinonime, ki imajo zaradi pomembnosti samostojen vnos. Dobra stran polja Glej je tudi, da se pojavi spustni meni, ki navaja vse že vnešene izraze. Pojavlja pa se vprašanje, kam – če sploh – je možno vnašati sorodne izraze, ki nikakor niso sinonimni, dajejo pa vseeno pomembne podatke o izrazu in sorodnih pojmih (npr. *information security – varnost podatkov; information system security – varnost informacijskega sistema).*

Najpomembnejši del obdelave gesla je iskanje razlage, ki naj bi jedrnato in hkrati dovolj natančno opredeljevala pomen iztočnice. iSlovar je namenjen predvsem slovenskim uporabnikom, zato so tudi razlage izključno slovenske. Študenti so si pri iskanju angleških razlag pomagali z različnimi viri, med drugim s funkcijo *define* v iskalniku Google in obstoječimi spletnimi slovarji informatike na internetu, najdene razlage pa so prevedli v slovenščino. Izjemoma je bilo ustrezno slovensko razlago možno najti na slovenskem spletu.

Čeprav je metoda prevajanja angleških razlag vse prej kot idealna, je vseeno vsaj osnova za oblikovanje končne različice razlage, saj smo se vseskozi zavedali, da bodo zbrana gesla vsekakor morali pregledati še strokovnjaki. Tako so razlage večinoma precej splošne in nenatančne, saj so študentje izbirali takšne, ki so jih sami razumeli. Nekaj primerov navajamo spodaj:

- *dekripcija*: vrnitev podatkov v izvirno obliko
- *digitalno potrdilo*: elektronski dokument, ki potrjuje verodostojnost osebe pri poslovanju oziroma drugih elektronskih transakcijah, na katerem je ime uporabnika, veljavnost, digitalni podpis pooblaščene osebe, ki je dokument izdala itd.
- *aplikacija jezikovnih tehnologij*: računalniški model za prepoznavanje in razumevanje naravnega človeškega govora

Skupno so študenti izbrali in obdelali okrog dvesto novih izrazov iz korpusa, poleg tega pa so z razlagami oskrbeli še približno sto že vnesenih izrazov. Ker v času pisanja še nismo imeli na razpolago povratne informacije urednikov, tega izdelka še ne moremo kakovostno ovrednotiti. Zavedati pa se moramo, da bi bilo s korpusno metodo v enakem času mogoče pridobiti tudi precej večje število izrazov, vendar brez slovarske obdelave.

S stališča prevajalcev kot rednih uporabnikov večjezičnih terminoloških del je pri večini obstoječih slovarjev, in iSlovar tu ni izjema, zanemarjena frazeološka plat strokovnega jezika. Delno se ta problem sicer rešuje s povezavo med slovarjem in korpusom, podobno zapolnitev te vrzeli sta udejanila projekta Evrokorpus in Evroterm za področje prava in evropske zakonodaje (Željko 2002). Pa vendar nam korpusne metode pri izdelavi slovarjev omogočajo lep vpogled v kolokacije in frazeologijo ob iztočnicah, česar pa se trenutno ne da umestiti v strukturo iSlovarja.

4.3 Nadaljnje jezikovne obdelave

Za izgradnjo terminološkega slovarja bi bilo seveda koristno, če bi lahko termine identificirali kar avtomatsko (Vintar, 2002). Metode samodejnega luščenja izrazov se v svetu razvijajo že nekaj časa, sprva predvsem za namene iskanja podatkov in samodejne klasifikacije dokumentov, danes pa se tovrstna orodja vgrajujejo tudi v prevajalske sisteme, kot je Tradosov. Identifikacija terminoloških kandidatov lahko temelji na statističnih metodah, ki kot kriterij upoštevajo relativno pogostost izraza in njegovo distribucijo po besedilu ali korpusu, jezikovno odvisne metode pa uporabljajo jezikoslovno analizirana besedila in izraze prepoznavajo na podlagi določenih oblikoskladenjskih vzorcev. Če so korpusi dovolj veliki, lahko najdenim izrazom statistično poiščemo tudi prevod, in tako nastaja dvojezični slovar terminoloških kandidatov brez človekove pomoči.

Seveda so zaenkrat, vsaj za slovenščino, tako pridobljeni seznami uporabni le kot osnova za nadaljnje slovarske obdelave, vendar je za številna področja, ki jim »ročno« slovaropisje ne uspe slediti, to vseeno boljše kot nič.

5 SKLEP

V članku smo predstavili izdelavo korpusa DSI ter njegovo uporabo pri dopolnjevanju spletnega slovarja SDI. Čeprav je predstavljeni postopek ilustriran na primeru, pa bi takšna uporaba jezikovnih tehnologij tudi za druga terminološka področja lahko bistveno olajšala in pohitrila slovaropisje v Sloveniji, da bi laže sledilo dinamiki strokovnega, pa tudi splošnega jezika.

Na tem mestu je vseeno potrebno opozorilo, ki zadeva celotno korpusno jezikoslovje: na korpusih temelječe analize in viri samo povzemajo jezik, ki se nahaja v korpusu. Metoda je torej deskriptivna in ne preskriptivna oz. z drugimi besedami, če so v korpusu uporabljani zastareli in neustrezni termini, bodo takšni tudi v konkordancah oz. v avtomatsko generiranem terminološkem slovarju. Za vire, ki naj bi imeli normativno funkcijo, je zato naknadna redakcija nujna.

Prihodnje delo je bilo, kar se tiče bolj bogatega označevanja, že nakazano v predhodnih poglavjih. Želeli pa bi si seveda korpus tudi razširiti. V kratkem bomo vanj dodali zbornik DSI za leto 2004, dolgoročnejši načrti pa predvidevajo zajem revije Uporabna informatika in tudi virov, ki ne izhajajo iz SDI – tu imamo v mislih predvsem vladne publikacije s področja informatike.

Izdelava korpusov in drugih jezikovnih virov je predraga, da bi bilo smiselno že v prvi fazi prepustiti njihov nastanek ekonomskim faktorjem, še posebej za jezike s tako majhnim številom govorcev, kot jih ima slovenski jezik. Z vladnim financiranjem in sodelovanjem akademskih institucij, društev, lahko pa tudi komercialnih partnerjev, kot so založbe, je nujno najprej omogočiti izdelavo predkompetitivnih virov, saj šele ti lahko dajo eno od prepotrebnih osnov za nadalnji razvoj raziskovanja in uporabe slovenskega jezika. Ti viri bi morali biti čim širše dostopni, kar lahko dosežemo po eni strani z uporabo mednarodnih standardov pri njihovem zapisu in označevanju, po drugi strani pa s čim bolj liberalnimi pogoji nadaljnjega razširjanja in uporabe. Korpus DSI je prosto dostopen za iskanje, za prepis pa ga nameravamo narediti dostopnega za raziskovalne in pedagoške namene.

Zahvala

Pri študentskem projektu so sodelovali Božo Borčnik, Katja Veber, Patricija Mencingar, Irena Perne, Jasna Čretnik, Teja Mlakar, Simona Vučak, Karmen Žerdin, Anja Čibej, Jana Štupnikar, Nina Mali, Barbara Damiš.

6 LITERATURA

[1] DŽEROSKI, Sašo, ERJAVEC, Tomaž, ZAVREL, Jakub: Morphosyntactic Tagging of Slovene: Evaluating PoS Taggers and Tagsets. V: *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC'00*, Athens, str. 1099–1104, 2000. http://nl.ijs.si/et/Bib/LREC00/lrec-tag-www/ SINCLAIR, John. *Looking Up: An Account of the COBUILD Project in Lexical Computing.* Collins, Glasgow. 1987.

[2] ERJAVEC, Tomaž. MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. V *Fourth International Conference on Language Resources and Evaluation, LREC'04*. Paris: ELRA. 2004. http://nl.ijs.si/ME/

[3] MANNING, Chirstoper, SCHÜTZE, Heinrich: *Foundations of Statistical Natural Language Processing.* The MIT Press. Cambridge MA. 1999.

[4] SPERBERG-MCQUEEN, C.M.; BURNARD, Lou (ur.). *Guidelines for Electronic Text Encoding and Interchange, the XML Version*. TEI Consortium, 2002. http://www.tei-c.org/

[5] VAN HALTEREN, Hans (ur.) *Syntactic Wordclass Tagging*. Kluwer, 1999.

[6] VINTAR, Špela: Avtomatsko luščenje izrazja iz slovensko-angleških vzporednih besedil. V: *Zbornik 3. konference o jezikovnih tehnologijah*, Ljubljana, str. 78–85, 2002. http://nl.ijs.si/isjt02/zbornik/sdjt02-14vintar.pdf

[7] W3C. *Extensible Markup Language (XML) 1.0 (Second Edition)*. (2000). http://www.w3.org/XML/

[8] W3C. *XSL Transformations (XSLT) Version 1.0 (1999)* http://www.w3.org/TR/xslt

[9] ŽELJKO, Miran: Pripomočki na spletu za prevajalce zakonodaje EU. V: *Zbornik 3. konference o jezikovnih tehnologijah*, Ljubljana, str. 33–39, 2002. http://nl.ijs.si/isjt02/zbornik/sdjt02-05zeljko.pdf

Dr. Tomaž Erjavec je znanstveni sodelavec na Odseku za tehnologije znanja na Institutu Jožef Stefan. Njegov raziskovalni interes je računalnišo jezikoslovje, tj. jezikovne tehnologije, korpusno jezikoslovje in strojno prevajanje, predvsem v povezavi s slovenskim jezikom. Diplomiral je na Fakulteti za elektrotehniko in računalništvo Univerze v Ljubljani (1984), magistiral pa na Fakulteti za računalništvo in informatiko (1990) in na Centre for Cognitive Science Univerze v Edinburghu (1992), doktoriral je na Fakulteti za računalništvo in informatiko Univerze v Ljubljani (1997). Je avtor več kot 50 znanstvenih člankov, član uredniških odborov mednarodnih revij CHum in IJCL, predsednik slovenskega društva za jezikovne tehnologije, bil pa je tudi član sveta Text Encoding Initiative Consortium ter European Chapter of the Association of Computational Linguistics.

Špela Vintar je na Filozofski fakulteti diplomirala iz angleščine in nemščine, zatem pa v okviru podiplomskega študija preživela nekaj večmesečnih obdobij na raziskovalnih projektih v tujini. Leta 2003 je doktorirala s področja samodejnega luščenja terminologije iz slovenskih in angleških besedil. Od leta 1998 je zaposlena na Oddelku za prevajalstvo Filozofske fakultete. Od tedaj se ves čas intenzivno ukvarja s korpusi, v zadnjem času pa tudi s poučevanjem korpusnih metod v prevajalstvu in slovaropisju.

Iskanje zakonov oblikovanja programske opreme z metodami znanosti o kompleksnosti

Peter Kokol[1,2], Milan Zorman[1,2], Mitja Lenič[1], Alojz Tapajner[1]

[1] Laboratorij za načrtovanje sistemov, FERI, Univerza v Mariboru, Smetanova 17, Maribor

[2] Center za interdisciplinarne in multidisciplinarne raziskave in študije Univerze v Mariboru, Krekova 2, Maribor

Povzetek

Z dramatičnim povečanjem kompleksnosti in velikosti programske opreme se veča tudi verjetnost pojava napak in kriznih situacij pri njeni uporabi. Žal konvencionalne metode za merjenje in kontrolo kvalitete niso dovolj uspešne, zato je potrebno uporabiti nekonvencionalne pristope, kot so teorija kompleksnosti in adaptivnih sistemov, fizikalno podprte metrike, evolucijske ekonomsko/tržne teorije ter inteligentno analizo podatkov, da bi dosegli temeljno razumevanje in večjo kakovost programske opreme in samega procesa njenega oblikovanja. V članku bomo predstavili prve rezultate evropskega projekta SQUFOL, v katerem smo za merjenje kvalitete programske opreme uporabili in nadgradili tehnike, ki so se uveljavile v zgoraj navedenih področjih.

Abstract

SOFTWARE QUALITY FOUNDED ON DESIGN LAWS AND SCIENCE OF COMPLEXITY

As the size and complexity of software systems is growing dramatically the possibility of crises from failure increases. Conventional methods for measuring and controlling quality are not successful enough, therefore it is our objective to use the science of complexity, the theory of chaos, fractals, traditionally 'physically based' methodologies, theory of complex adaptive systems, biologically and evolutionary inspired economic/market theories, intelligent data analysis and data mining to gain a fundamental understanding of the software process and the software products and to improve them. The aim of the recent paper is to present the first results of the European project SQUFOL.

1 Uvod

Razvoj in oblikovanje programske opreme je kompleksen proces [1, 4, 9, 14]. Še nedavno tega smo nanju gledali kot na umetnost, pa tudi danes še nista dosegla vseh zahtev popolnoma inženirske discipline. V zadnjih desetletjih, še posebej v zadnjih nekaj letih sta velikost in kompleksnost programske opreme dramatično narastli. Sistemi z več sto milijoni vrstic programskega koda niso več nobena posebnost, poleg tega zahteve po kompleksnih programskih sistemih naraščajo hitreje kot naše zmožnosti, da takšne sisteme načrtujemo, gradimo in vzdržujemo. S porastom naših zahtev in s porastom naše odvisnosti od računalnikov naraščajo možnosti, da bo zaradi izpada računalniškega sistema prišlo do različnih kriznih situacij, ki lahko obsegajo manjše neprijetnosti, finančne nepravilnosti in izgube, v skrajnih primerih pa celo izgube človeških življenj. Torej je jasno, da zanesljivost programske opreme ne zadeva več samo načrtovalcev programske opreme in računalniških strokovnjakov, ampak tudi širšo družbo kot celoto. Zato je konsistentno in ekonomično doseganje najvišjega nivoja kvalitete programske opreme ključnega pomena, vendar zato potrebujemo zakone razvoja programske opreme, ki jih žal še nismo odkrili. Popolnoma jasno je, da lahko zakone oblikovanja odkrijemo samo z uporabo empiričnega znanstveno/sistemskega pristopa, za kar pa potrebujemo uspešne metrike programske opreme. Žal konvencionalne metode za merjenje programske opreme niso dovolj uspešne, zato je potrebno uporabiti nekonvencionalne pristope.

1.1 Oblikovanje programske opreme je kompleksen adaptiven sistem

Tako kot kompleksni sistemi v ekonomiji in menedžmentu ima tudi razvoj programske opreme (razvoj tu zajema celoten življenjski cikel, od idej in opisov potreb do uporabe in vzdrževanja) podobne sistemske lastnosti [3, 11]: je kompleksen, dinamičen, nelinearen in adaptiven. Posledično je cilj evropskega projekta SQUFOL [5] (Software Quality Founded on Design Laws) odkriti temeljne zakone programske opreme,

njene evolucije, kvalitete in razvojnega procesa z uporabo pristopa, ki združuje teorijo kompleksnosti, razne sistemske teorije, teorijo kaosa ter fraktalov, tradicionalne fizikalno podprte metodologije, teorijo kompleksnih adaptivnih sistemov, evolucijske tržnoekonomske teorije, inteligentno analizo podatkov, podatkovno rudarjenje ter druge metode, ki še niso priznane s strani raziskovalcev oblikovanja programske opreme. Poznavanje temeljnih zakonov bo omogočalo povečanje znanja o kvaliteti programske opreme in njenem razvojnem procesu, oblikovanje enostavnega in razumljivega procesa oblikovanja in kot posledico razvoj visoko kvalitetne programske opreme, ki ne bo povzročala prej omenjenih težav. V članku bomo predstavili enega izmed prvih rezultatov projekta SQUFOL – zakonitost, da se oblikovanje programske opreme obnaša podobno kot dinamični kaotični sistemi, generirani s t. i. logistično enačbo.

2 Težave pri oblikovanju programske opreme

Porast zahtev po kvalitetni programski opremi in nezmožnost izpolnjevanja le-te sta pripeljala do dobro znane 'krize' programske opreme. Njeni vzroki in posledice so predmet mnogih razprav v literaturi [10], naše mnenje pa je, da je glavni vzrok te krize nezmožnost poiskati in razumeti temeljne zakone programske opreme, njihovo uporabo, kvaliteto in njihov proces razvoja. Menimo, da je osnova za reševanje krize odkrivanje temeljev in zakonov oblikovanja, ki v katerikoli znanstveni disciplini temelji na zmožnosti merjenja. Žal na področju razvoja programske opreme ni dovolj uspešnih programskih metrik, za kar so krivi naslednji vzroki [7]:

- Metrike so odvisne od jezika in oblike: za različne programske jezike (izvorna koda), objektne kode itd., je potrebno uporabiti različne metrike. Nadalje je programska oprema lahko predstavljena v različnih oblikah, kot so sistemski koncepti, zahteve, specifikacije, razvojna dokumentacija, uporabniški vmesniki, zbirke pomoči itd. Še dodatno so lahko vse te predstavitve predstavljene v različnih oblikah, kot so tekst v naravnem jeziku, grafična predstavitev, simbolni zapis, tekst, zapisan s formalnimi jeziki ipd. Za vsako od teh različnih predstavitev moramo uporabiti svojo metriko, kar pomeni, da nismo zmožni primerjati enakih izdelkov v različnih oblikah ali predstavitvah ter tako ne moremo slediti spremembam kompleksnosti skozi razvojne faze.
- Izhod tradicionalne metrike je številka, ki nima nikakršnega 'fizikalnega' pomena in enote (ne vemo, kaj merimo), niti nima znanih kritičnih vrednosti, ki bi povedale, kdaj je izmerjena vrednost majhna ali velika.
- Relacije metrike – programska oprema (za izdelke in procese) so redko znane, zato so tudi takšne metrike slaba podlaga za postavitev temeljnih zakonitosti. To je posledica predvsem dveh dejstev:
  - Metrika ni bila izpeljana na podlagi teorije, ampak na podlagi pragmatičnih kratkoročnih ciljev.
  - Implementacija učinkovitih metrik je pogosto izvedena brez popolnega razumevanja metrike same ali je le-ta osnovana na minimalni množici »poceni« postopkov za zbiranje podatkov.

3 Projekt SQUFOL

Da bi se izognili gornjim problemom, smo v projektu SQUFOL uporabili in nadgradili tehnike, ki so se uveljavile v okviru vse bolj priljubljene znanosti o kompleksnih sistemih:

1. Na program gledamo kot na zbirko simbolov [6, 12, 13] in tako nanj apliciramo tehnike, ki ta program pretvorijo v 'signale'. Na teh signalih uporabimo tehnike analize, ki izhajajo s področja fizike. Primeri teh analiz so izračun informacijske vsebnosti, daljnosežnih korelacij, entropije ipd.
2. Faze razvoja programske opreme lahko primerjamo s fazami iterativnih map [2].
3. Na razvoj programske opreme lahko gledamo kot na trženje idej. Vsaka ideja ima svoje gene/meme in tekmuje za vire. Najboljše ideje preživijo, se med seboj križajo in mutirajo. Zato lahko procese analiziramo z uporabo evolucijsko-bioloških ekonomsko/tržnih teorij.
4. S celovitim pristopom in sekvenčnimi študijami lahko izmerimo vsa dejstva o programski opremi in njenem razvoju.
5. Z uporabo inteligentne analize podatkov, podatkovnega rudarjenja in ekstrakcije znanja lahko odkrijemo relacije, skrito znanje in zakone oblikovanja programske opreme.

V nadaljevanju članka bomo predstavili prvi dve ideji.

4 Inverzne bifurkacije, logistična funkcija in oblikovanje programske opreme

Programsko opremo običajno oblikujemo z iterativnim procesom [2] – računalniški program se razvija

skozi različne verzije (slika 1). Na začetku procesa imamo veliko različnih idej, kako pravilno ali nepravilno predstaviti rešitev. Kreativnost je na vrhuncu, entropija je velika, vsebnost informacije je nizka. V naslednjih korakih se razvijejo pravilne in optimalne rešitve. V zadnji fazi se po navadi ukvarjamo z eno samo idejo (verjetno zelo kompleksno). Entropija in kreativnost sta nizki (imamo eno rešitev), medtem ko se informacijska vsebnost poveča.

Za modeliranje gornjega procesa lahko uporabimo zelo popularen in zanimiv koncept iz teorije kompleksnih adaptivnih sistemov – to je logistično funkcijo, ki ponazarja dinamiko biološke populacije. Zanjo predpostavljamo, da generira podobno obnašanje, kot je obnašanje procesa oblikovanja programske opreme, vendar v obratnem vrstnem redu. Logistična funkcija je definirana kot

$$X_{n+1} = \pi X_n (1 - X_n)$$

kjer X v originalu predstavlja normirano število osebkov populacije, v primeru oblikovanja programske opreme pa informacijsko vsebnost ideje.

Dinamika logistične funkcije vsebuje tri faze: kaos, bifurkacijo in normalno fazo. Bifurkacija predstavlja podvojitev števila atraktorjev, v našem inverznem procesu pa združevanje idej. Če naredimo preslikavo med logistično funkcijo in procesom razvoja programske opreme, lahko faze logistične funkcije preprosto identificiramo s kontrolnim parametrom p in tako ugotovimo, v kateri fazi oblikovanja smo. Seveda za to potrebujemo metriko informacijske vsebnosti idej (v našem primeru je ideja predstavljena kot računalniški program, psevdokod, algoritem ipd.) in jo opisujemo v naslednjih razdelkih.

5 Fraktalna programska metrika α

Fraktalna metrika α [6] temelji na izračunu daljnosežnih korelacij. Metriko smo razširili iz originalne Schenklove metode za pisana besedila na metodo znakov. Podobno kot na pisano besedilo lahko gledamo na računalniški program ali algoritem kot na niz znakov: črk, števk, ločil. Z uporabo kodne tabele pretvorimo vsak znak v pripadajoče binarno zaporedje; program tako preslikamo v model dvodimenzionalnega Brownovega gibanja, ki je osnova za izračun regresijske funkcije $F(l)$:

$$F^2(l) \equiv \overline{[\Delta y(l, l_0)]^2} - \left[\overline{\Delta y(l, l_0)}\right]^2$$

Prvi del zgornje enačbe predstavlja povprečje kvadratov razlik med dvema točkama v modelu Brownovega gibanja in drugi del kvadrat povprečij razlik, kjer razliko izračunamo s spodnjo enačbo:

$$\Delta y(l, l_0) \equiv y(l_0 + l) - y(l_0)$$

Regresijska funkcija $F(l)$ loči med dvema tipoma obnašanja:

- če je podano zaporedje znakov nekorelirano ali so prisotne lokalne korelacije, ki segajo do nekega karakterističnega ranga (npr. Markovske verige ali simbolično zaporedje, tovorjeno iz regularnih gramatik), potem je

$$F(l) \approx l^{0.5}$$

- če ne obstaja karakteristična dolžina in so korelacije "neskončne", potem je skalirna lastnost $F(l)$ opisana s potenčnim zakonom

$$F(l) \approx l^{\alpha} \text{ in } \alpha \neq 0.5$$

Koeficient α izračunamo po metodi najmanjših kvadratov kot linearno predstavitev točk na dvojni logaritemski skali $F(l)$, l in predstavlja kompleksnost računalniškega programa.

Metrika a meri vsebnost informacije programa. Večje kot je odstopanje α od 0.5, večja je informacijska vsebnost, manjša entropija in manjša kreativnost. Kot posledica vrednosti α blizu 0.5 pomenijo faze blizu kaotičnega obnašanja. To domnevo smo podkrepili s primerjavo realnih in naključnih programov z enostavnimi statističnimi metodami. Naleteli smo na razlike v vrednostih α med realnimi in naključnimi programi. Hipotezo smo preverili na podlagi dejstva, da so naključni programi (vsaj v našem poskusu) sintaktično pravilni, vendar brez pomena, kar pomeni, da imajo le sintaktično strukturo, ne pa tudi semantične; njihova struktura se torej bistveno razlikuje od strukture običajnih programov.

Da bi še bolj izpostavili lastnost metrike α, smo uvedli normalizirano obliko α:

$$\alpha_{nor} = 2|\alpha - 0.5|$$

V tej obliki ležijo vrednosti α_{nor} med 0 in 1, kjer 0 predstavlja maksimalno entropijo. Za izračun faze procesa razvoja programske opreme moramo izračunati vrednost π po naslednji enačbi:

$$\pi = \frac{X_{n+1}}{X_n(1-X_n)}$$

Bifurkacije v logistični funkciji predstavljajo število idej, medtem ko α_{nor} predstavlja inverzno relacijo števila idej (0 – veliko idej, 1 – zelo malo idej). Za izračun π potrebujemo tako inverzno relacijo normaliziranih idej

$$I = 1 - \alpha_{nor}$$

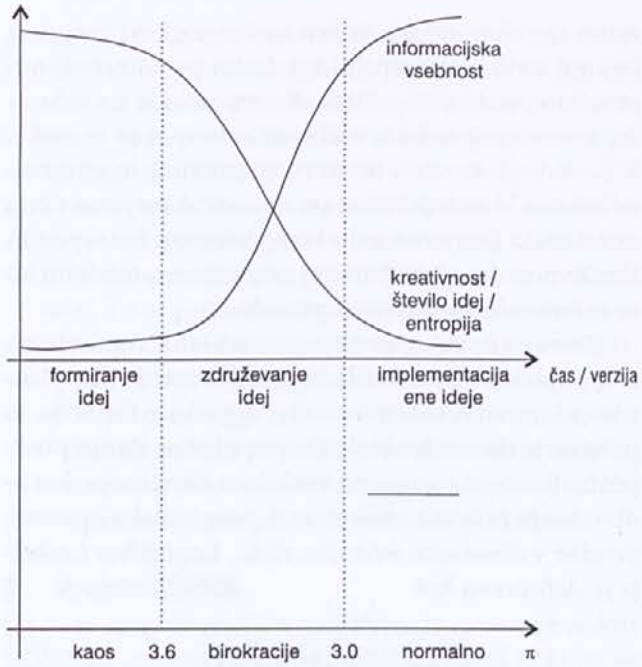
Glede na trend α vrednosti ločimo med sedmimi različnimi fazami, prikazanimi v tabeli 1.

5 Primer

Predstavljeno teorijo smo preizkusili na realnem projektu razvoja programske opreme. Prvi pogoj za izbiro projekta je bilo čim večje število verzij projekta skozi čim daljše obdobje. Pri iskanju primernega projekta smo se omejili na projekte z odprto kodo, saj za izvajanje opisane analize potrebujemo dostop do vseh verzij izvorne kode. V ta namen smo izbrali projekt Pure-FTPd 0 [15], ki se še aktivno razvija in je po navedbah avtorjev v stabilni/produkcijski fazi. Projekt je sestavljen iz več modulov. Projekt se ukvarja z razvojem storitve za prenos datotek prek FTP protokola (file transfer protocol).

V članku podrobneje prikazujemo analizo dveh modulov, ki sta se tekom oblikovanja najbolj spreminjala in obenem predstavljata jedro projekta. Prvi modul je razpoznavalnik (45 verzij), ki je vmesnik med samo storitvijo in uporabnikom/aplikacijo, ki to storitev uporablja. Drugi modul je storitev sama (FTPd) (310 verzij).

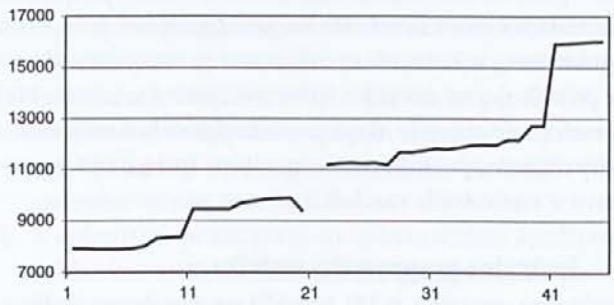
Tipična "napaka" konvencionalnih metrik kompleksnosti je tudi, da po navadi z večjo velikostjo izvornih datotek izmerijo tudi večjo kompleksnost/oceno, četudi se kompleksnost programa ni povečala. Če pogledamo sliki 2 in 3 vidimo, da α metrika nima te



Slika 1: **Razvoj programske opreme in logistična funkcija**

napake, saj se kompleksnost spreminja neodvisno od velikosti.

Da bi se izognili neprimernosti α metrike za majhne module (saj je α metrika načrtovana za izračun daljnosežnih korelacij), smo uvedli modificirano



Slika 2: **Rast velikosti razpoznavalnika skozi verzije**

α trend	Faza logistične funkcije/ Fazni trend	Faza razvojnega procesa/ Fazni trend	Trenutna faza
Stalen	Normalna	Ena ideja	Normalno/Ena ideja
Padajoč	Proti kaosu	Proti formiranju idej	π manjše kot 3 normalno/ Ena ideja
Rastoč	Proti normalnemu	Proti eni ideji	π večji od 3.6 kaos/ formiranje idej
Oscilira	Bifurkacija ali kaos	Združevanje idej ali kaos	π večji od 3.6 kaos/ formiranje idej

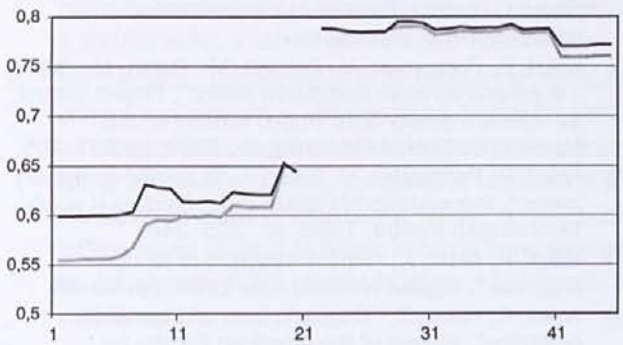
Tabela 1: **Faze in stanje programske opreme**

metriko α^*, ki po zgledu drugih matematičnih orodij za izračun korelacij upošteva, da je signal periodičen. Tako smo v izračunu α metrike spremenili:

$$\Delta y(l, l_0) \equiv y((l_0 + l) \bmod s) - y(l_0)$$

kjer je s dolžina Brownovega gibanja.

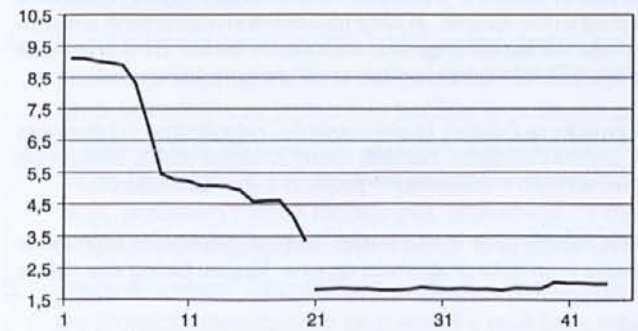
Na sliki 3 sta prikazani metriki α s sivo in α^* s črno barvo. Večje odstopanje metrik, kot je bilo pričakovano, opazimo samo v začetni fazi, tako da smo za vse nadaljnje preizkuse uporabili kar originalno α metriko.



Slika 3: **α in α^* metriki za razpoznavalnik**

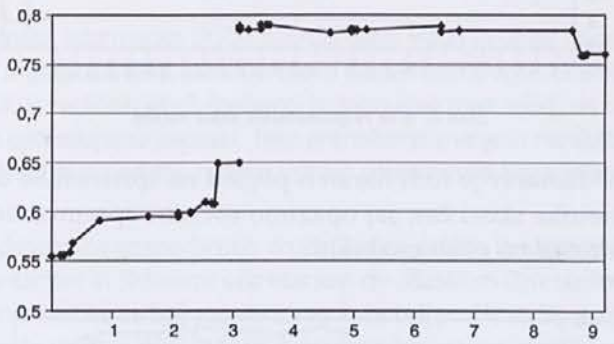
Na sliki 3 vidimo, da je med verzijo 21 (1.18) in 22 (1.19) prišlo do kvalitativnega preskoka, ki ni bil posledica bistvene spremembe velikosti datoteke – v konkretnem primeru je bil prepisan pregledovalnik, kar je doprineslo k večji informacijski vsebnosti programa/ideje.

Na sliki 4 lahko sledimo spremembi kontrolnega parametra π. Na začetku opazimo kaotičnost in počasno formiranje ideje ($\pi > 3.6$), kako razviti razpoznavalnik. Bistven preskok se ponovno zgodi med verzijama 21 in 22, kjer se očitno razjasni ideja rešitve.



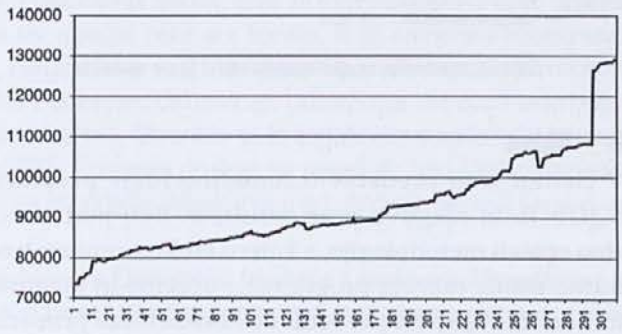
Slika 4: **π za razpoznavalnik skozi verzije**

Prikaz spreminjanja α metrike glede na verzije je lahko zavajajoč. Če pogledamo s časovno komponento (slika 5), nam osciliranje α prikaže v popolnoma drugačni luči. Opazimo, da se ideje in spremembe dogajajo časovno zgoščeno, nato pa ena rešitev začasno prevlada.

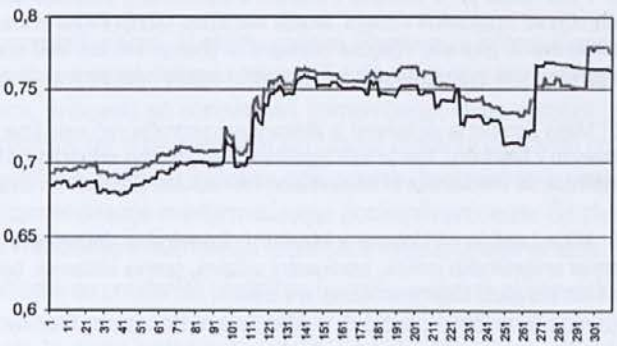


Slika 5: **α in α^* metriki za razpoznavalnik skozi čas v tednih**

Na slikah 6, 7 in 8 je prikazana analiza za modul FTPD. Opazimo lahko, da je ideja rešitve že od začetka relativno dobro znana in se skozi verzije samo kristalizira ($\pi < 3$).



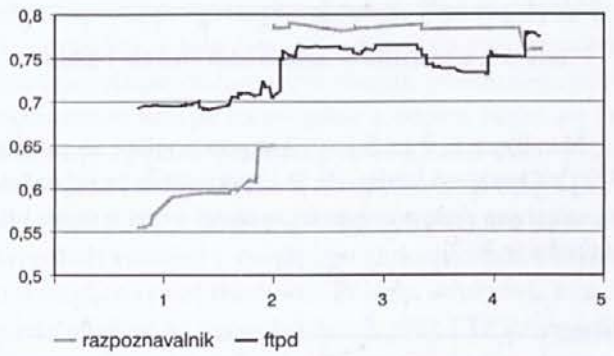
Slika 6: **Rast velikosti ftpd skozi verzije**



Slika 7: **α in α^* metriki za ftpd skozi verzije**

Slika 8: **π za razpoznavalnik skozi verzije**

Zanimiv je tudi hkraten pogled na spremembe α metrike skozi čas, saj opazimo sočasne spremembe metrike na obeh modulih.



Slika 9: **α metrika za oba modula skozi čas v mesecih**

6 Sklep

V članku smo predstavili temeljne ideje projekta SQUFOL in njegove prve rezultate. Bolj podrobno smo opisali metodologijo, s katero lahko ocenimo trenutno stanje razvoja programske opreme in njegove trende. Teorijo smo preverili na konkretnem primeru in pokazali njeno empirično veljavnost.

7 Literatura

[1] Bar Yam, Y., "Dynamics of Complex Systems", Addison Wesley, 1997.

[2] Cardoso, A. I., Crespo, G., "Is the software process a cahotic one?", Working paper of Mathematical Science Center, Madeira University, 1999.

[3] Gell-Mann, M., "What is complexity", Complexity 1(1), 1995, pp. 16–19.

[4] Kitchenham, B., "The certainty of uncertainty", Proceedings of FESMA (Eds: Combes H. et al), Technologish Institut, 1998, pp. 17–25.

[5] Kokol P. et al. Software quality founded on design laws (SQUFOL): final report [about EU project] 5th Framework Programme, reporting period under review 1/10/2002–30/9/2003, (5th European framework project). Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko.

[6] Kokol, P., Podgorelec, V., Zorman, M., Pighin, M., "Alpha – a generic software complexity metric", Project control for software quality (Eds: Rob J. Kusters et al.), Maastricht : Shaker Publishing BV, 1999, pp 397–405.

[7] Kokol, P., Podgorelec, V., Brest, J., "A wishful complexity metric", Proceedings of FESMA (Eds: Combes H et al), Technologish Institut, 1998, pp. 235–246.

[8] Kokol, P., Brest, J., "Fractal structure of random programs", Sigplan Notices, June 1998, pp. 33–38.

[9] Kokol, P., Kokol, T., "Linguistic laws and computer programs", Journal of the American Society for Information Science, 47(10), 1996, pp. 781–785.

[10] Kokol, P., Stiglic, B. and Zumer, V.: Metaparadigm: a soft and situation oriented MIS design approach. International Journal of Bio-Medical Computing 39, 1995, pp. 243–256.

[11] Morowitz, H., "The Emergence of Complexity", Complexity 1(1), 1995, pp. 4.

[12] Podgorelec V., Kokol, P.: Uporaba fraktalne metrike kompleksnosti pri avtomatskem programiranju, Zbornik devete Elektrotehniške in računalniške konference ERK, 2000, str. 133–136.

[13] Schenkel, A., Zhang, J., Zhang, Y.: Long range correlations in human writings, Fractals 1(1), 1993, pp. 47–55.

[14] Tucker, A.B. (ed.), "The Computer Science and Engineering Handbook", CRC Press, 1997.

[15] http://sourceforge.net/projects/pureftpd/

Dr. Peter Kokol je na Univerzi v Mariboru diplomiral s področja elektrotehnike in doktoriral s področja računalništva. Njegova raziskovalna področja so inteligentni sistemi, teorija sistemov, teorija kaosa in kvaliteta programske opreme. Je vodja nacionalnih in mednarodnih projektov iz imenovanih področij. Njegova bibliografija obsega več kot 500 enot, od tega več kot 60 originalnih znanstvenih člankov. Bil je predsednik organizacijskih in programskih odborov več svetovnih konferenc. Je predsednik IEEE tehničnega komiteja za računsko medicino.

Dr. Milan Zorman je diplomiral in doktoriral s področja računalništva in informatike na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru, kjer je tudi zaposlen. Raziskovalno deluje na področjih umetne inteligence, hibridnih metod strojnega učenja, inteligentnih sistemov ter medicinske in zdravstvene informatike. Sodeluje pri izvajanju več domačih in mednarodnih projektov z omenjenih področij.

Dr. Mitja Lenič je na Univerzi v Mariboru diplomiral in doktoriral s področja računalništva in informatike. Njegova raziskovalna področja so teorija programskih jezikov, inteligentni sistemi, teorija sistemov, teorija kaosa in kvaliteta programske opreme. Njegova bibliografija obsega več kot sto enot, objavljenih doma in v tujini.

Alojz Tapajner je diplomiral s področja računalništva in informatike na Univerzi v Mariboru. Njegova raziskovalna področja so inteligentni sistemi in posebno njihova uporaba na finančnih področjih. Sodeluje pri raziskovalnih projektih Laboratorija za načrtovanje sistemov.

Udeleženci posvetovanja Dnevi slovenske informatike 2004, ki je bilo 14. do 16. aprila 2004 v Portorožu, smo z namenom, da bi ocenili vlogo, pomen in cilje posvetovanja ter priporočili usmeritve za prihodnja posvetovanja, sprejeli naslednjo

DEKLARACIJO 11. POSVETOVANJA DNEVI SLOVENSKE INFORMATIKE 2004

1. Organizatorji posvetovanja ugotavljamo, da so Dnevi slovenske informatike 2004 potrdili svojo vlogo in ugled najpomembnejšega neodvisnega srečanja slovenskih informatikov iz gospodarstva, javne uprave in akademskih krogov. Ocenjujemo, da smo dosegli namen posvetovanja, da se informatiki iz različnih okolij srečamo in spoznamo med seboj, da izmenjamo mnenja, izkušnje, informacije in spoznanja ter jih posredujemo javnosti. Tako prenašamo znanje in rezultate raziskav informatikov iz akademskega okolja v prakso, informatiki iz podjetij in javne uprave pa seznanijo svoje kolege s problematiko, s katero se srečujejo v praksi.
2. Na predvečer posvetovanja je bil organiziran sestanek predstavnikov gospodarskih družb, civilne družbe in slovenskih univerz z ministrom za informacijsko družbo dr. Pavlom Gantarjem in državnim sekretarjem dr. Józsefom Györkösem, na katerem je bil dosežen načelni dogovor, kako v prihodnje informatiko še bolj popularizirati in še bolj profilirati Slovenijo kot družbo znanja. V okviru sprejema za udeležence je Slovensko društvo INFORMATIKA predstavilo poskusni snopič terminološkega slovarja informatike *Islovar* in novo podobo dokumentov uporabniškega računalniškega spričevala *ECDL*. Islovar razvija društvo že četrto leto in je javno dostopen na naslovu http://www.ef.uni-lj.si/terminoloskislovar. Nova podoba dokumentov ECDL obsega indekse in spričevala, ki so od aprila 2004 usklajeni z navodili ustanove ECDL in enaki v vsej Evropi.
3. Na otvoritvi posvetovanja je udeležence v imenu organizatorjev pozdravil predsednik Slovenskega društva INFORMATIKA Niko Schlamberger. Posvetovanje namenja trajen poudarek sodelovanju vseh subjektov na področju informatike, tako države, univerz, gospodarskih družb in nevladnih organizacij, v okviru tega pa znanstvenim in strokovnim vidikom informatike ter odličnosti in kulturnemu pomenu tega dogodka. Predstavnik Ministrstva za informacijsko družbo državni sekretar dr. József Györkös je poudaril pomen informatike ter nakazal nekatere korake, ki jih namerava ministrstvo narediti v prihodnje. Častni govornik prof. dr. Ivan Rozman, rektor Univerze v Mariboru, ki prihaja iz vrst informatikov, je poudaril različne vidike informatike v izobraževanju skladno z bolonjsko deklaracijo. Udeležence sta pozdravila predstavnika pokroviteljev, Microsofta in Oracla, Jaka Stele in Vasja Herbst. Otvoritev se je nadaljevala s podelitvijo priznanj društva ter Združenja za informatiko in telekomunikacije pri GZS. Priznanja društva so prejeli dr. Ivan Meško, podjetje Infos in dr. Mojca Indihar Štemberger. Priznanje GZS-ZIT, ki ga je podelila sekretarka mag. Andreja Ivartnik Kanduč, je prejel Franci Mugerle.
3. Sledilo je vabljeno predavanje prof. dr. Andreja Kovačiča z ljubljanske Ekonomske fakultete z naslovom *Menedžment in informatika – kako odpraviti prepad*. Vsebina je bila problematika semantičnega prepada med obema sferama; prikazana je bila tudi metodološka rešitev, ki sloni na obravnavi poslovnih pravil pri prenovi in informatizaciji poslovanja. Vabljena predavateljica Alexa Bona, GartnerGroup, je v svojem predavanju obravnavala aktualno problematiko celovitih poslovnih rešitev (ERP) ter nakazala nekaj mogočih pristopov, s katerimi lahko ob sklepanju pogodb z dobavitelji le-teh veliko prihranimo.
4. Popoldne se je posvetovanje nadaljevalo z dvema vzporednima sekcijama. V okviru sekcije *Prenova in informatizacija poslovanja* so prispevki obravnavali učinkovito izvajanje poslovnih procesov. Vloga informacijske tehnologije je v optimalni izrabi tehnologije za podporo in izboljšanje poslovnih procesov. Prispevki so obravnavali pomembnost vloge vodstva pri uvajanju in uporabi sistemov za upravljanje dokumentov in procesov ter o problematiki simulacije izvajanja poslovnih procesov. Sekcija se je nadaljevala s prispevki o konkretnih projektih na področju logistike, transporta, proizvodnje in poslovanja, predvsem z vidika modeliranja, analiziranja, prenove, standardizacije in informatizacije poslovnih procesov. Še zlasti veliko zanimanja je bilo za prispevek, ki je obravnaval informatizacijo slovenskega transportnologističnega grozda.
5. Prispevki v sekciji *Metodologije, pristopi, informacijska tehnologija* so poudarjali praktično uporabo v realnih okoljih oziroma projektih; metodologijo so postavili v vsakdanjo rabo in nakazali prednosti in slabosti. V četrtek je večina prispevkov obravnavala varnostne rešitve. Na delavnici *Upravljanje storitev IT za uspešno izvajanje poslovnih procesov* so se

udeleženci teoretično in praktično seznanili s tem, kako upravljati storitve IT, kot so e-pošta, dostop do interneta in delo s finančnimi in poslovnimi aplikacijami za uspešno izvajanje poslovnih procesov. Dan se je zaključil z *okroglo mizo o partnerstvu menedžmenta in informatike*, na kateri je razprava tekla predvsem o poslovni vplivnosti informatike, vlaganjih vanjo in njenem vrednotenju, o vlogi vodje informatike, ki je velikokrat premajhna (samo 15 % informatikov je članov najožjega vodstva slovenskih podjetij) ter o trendih in usmeritvah na področju informatizacije poslovanja. Ugotovili smo, da so za uspešno informatizacijo bistvenega pomena urejeni poslovni procesi. Da bo mogoče ustvariti partnerstvo med menedžmentom in informatiko, je nujno, da dobijo menedžerji več znanja s področja informatike in informatiki več poslovnih znanj.

6. Mednarodne vidike posvetovanja sta nadaljevali vabljeni predavanji v sekcija *Informacijska podpora odločanju*. Prof. dr. Heinrich C. Mayr z Univerze v Celovcu je orisal in analiziral dosedanji razvoj tehnologij in pristopov na področju menedžerskih informacijskih sistemov, pri čemer je opozoril na razloge, zakaj ne izpolnjujejo pričakovanj po zagotavljanju vseh potrebnih informacij menedžerjem. Dr. Nataša Milič - Frayling, Microsoft Research, Cambridge, je izpostavila praktične težave pri iskanju informacij v besedilih, kjer pogosto ni mogoče vnaprej opredeliti zahtev. Predstavila je tudi prototip orodja, s katerim bo »rudarjenje« po besedilih postalo uporabno v poslovnih okoljih. Sekcija se je nadaljevala s prispevki, v katerih so avtorji opozorili na aktualne vidike razvoja sistemov za podporo odločanju s poudarkom na podatkovnih skladiščih in podatkovnem »rudarjenju« (Data Mining). V prvem delu so bile predstavljene posebnosti in nevarnosti pri vodenju projektov podatkovnega skladiščenja ter nakazani ukrepi za zmanjševanje tveganj. Posebej je bila analizirana problematika vrednotenja uspešnosti projektov podatkovnega skladiščenja.
7. V sekciji *Strateški vidiki informatizacije* so bile predstavljene raziskave s področja vpliva informatike na menedžment slovenskih podjetij in vrednotenja projektov e-poslovanja, strategija sektorja IT ter pričakovanj slovenskega trga IT ob vstopu v EU. Mnenja smo, da bi bilo treba tudi na področju informatike postaviti standarde in merila, kdo lahko izvaja projekte in kdo je informatik. Udeleženci menimo, da je nujno treba vzpostaviti sistem strokovnih izpitov kot v drugih strokah. Sekcija se je povezala z *okroglo mizo Strategije sektorja IKT*, ki je bila namenjena kritični presoji ideje o strateški pomembnosti sektorja IKT za razvoj slovenske družbe in gospodarstva. Tako analize kot okrogla miza so potrdile, da ima sektor IKT potencial kot sektor z visoko dodano vrednostjo, z rastjo prihodkov in številom zaposlenih in ki s pozitivnim horizontalnim vplivom doprinaša k razvoju slovenskega gospodarstva. Osnovni cilj strategije je razvoj takega sektorja IKT, ki nas bo pripeljal v informacijsko družbo, pripomogel k razvoju tehnologije, gospodarstva in industrije znanja, ki bo uspešno, dobičkonosno in izvozno usmerjeno.
8. V sekciji *Informacijske rešitve* je bila obravnavana odprta koda in udeleženci z zadovoljstvom ugotavljamo, da so se v Sloveniji začeli premiki k večji uporabi odprtokodnih rešitev in da je zaznati nove poslovne modele, ki temeljijo na odprti kodi in zaračunavanju storitev. Slednje je še na začetku in večjih sprememb med proizvajalci IT v kratkem ni pričakovati, trend pa ocenjujemo kot pozitiven. Predstavljenih je bilo tudi več prispevkov, ki so obravnavali izkušnje s področja razvoja in uvedbe informacijskih rešitev.
9. Med različnimi temami, ki jih je obravnavala sekcija *Izobraževanje za informacijsko družbo*, je bilo predstavljenih nekaj prispevkov, ki so obravnavali problematiko izobraževanja na različnih nivojih. Posebej zanimivo je bilo vabljeno predavanje avtorjev iz Avstrije o varnostnih vidikih izobraževanja zapornikov z namenom zmanjševanja socialne izključenosti in o možnostih, ki jih pri tem ponuja uporaba informacijske tehnologije.
10. Sekcija *Uporaba operacijskih raziskav* se je začela z vabljenim predavanjem prof. ddr. Viljema Rupnika. V okviru predavanja se je držal rdeče niti posvetovanja in tako predstavil informatiko kot spremljevalko menedžmenta, njunega medsebojnega vpliva in problema, kako uokviriti eno v drugo in ju vsebinsko koordinirati. Vabljenemu predavanju je sledilo nekaj referatov, ki izhajajo iz realnih problemov, za katere so oblikovali model in metode reševanja.
11. Delavnico *Od jezika XML in spletnih storitev do orkestracije poslovnih procesov* sta izvedla sodelavca mariborske Fakultete za elektrotehniko, računalništvo in informatiko. Na praktičnih primerih so bili uporabljeni in predstavljeni: osnovni koncepti jezika XML, pomen besednjakov in načini njihovega oblikovanja (DTD in XML sheme), pomen ločitve podatkov od prikaza ter pomen tehnologije XSL, podane in prikazane prednosti pridobitve uporabe tehnologij XML na izmenjavi podatkov, povezovanju informacijskih rešitev ter medorganizacijskem poslovanju in povezovanju, spletne storitve in način obdelave XML dokumentov in programskih jezikov.
12. V sekciji *Informacijska družba* so referenti obravnavali različne vidike socialnega učinkovanja IT. Dejstvo je, da EU ne le spodbuja, ampak tudi sistematično meri učinke informacijske družbe. Za ta namen so bili razviti različni indikatorji ter

matrika ISBM, ki prikazuje pripravljenost, razširjenost, intenzivnost in rezultate pri uveljavljanju sestavin informacijske družbe. Predstavljen je bil program ECDL, ki odločilno prispeva k digitalni pismenosti. Sekcija se je posvetila tudi problemom žensk pri uveljavljanju v informatiki. S podobnega vidika so bile obravnavane težave, ki jih imajo levičarji pri uporabi neprilagojene strojne opreme. Odmevne so bile predstavitve rešitev aktivnih omrežij, mobilnih terminalov in programske podpore likovnemu ustvarjanju. Posebej zanimiva je bila predstavitev dejavnosti mariborske Kible.

13. Sekcija *Uresničevanje e-uprave* je obsegala predstavitev novih storitev e-uprave, ki jih predvideva akcijski načrt vlade. Podani so bili problemi in koristi e-uprave ter nekateri kritični pogledi na e-poslovanje; niso dovolj le spletne strani, pač pa je treba urediti ter ustrezno informatizirati notranje poslovanje uprave. To je mogoče doseči z reorganizacijo in prenovo poslovanja, ki je tako kot v poslovnem svetu tudi v upravi nujna. Predstavljen je bil predlog enotne arhitekture e-uprave, ki bo ključna in zahtevna naloga njenega prihodnjega razvoja. Predlog vsebuje metamodel, ki razen tehnološkega pokriva tudi zakonodajni, dokumentni in postopkovni vidik. Opazen je bil prispevek o možnostih za uporabo informatike na področju pravosodja, predvsem na sodiščih za spremljanje sodnih postopkov.
14. Izhodišče za *okroglo mizo e-uprava* je bilo, da moramo biti za informacijsko družbo in članstvo v EU pripravljeni, del tega pa se uresničuje tudi prek e-uprave. Panelisti so ugotavljali, da je e-delo izhodišče za prenovo postopkov in da je potrebna prenova poslovanja. Dotaknili so se pomanjkljivosti načina merjenja razvitosti prek modela dvajsetih storitev, med katerimi manjka e-demokracija in informatiziranost vlade, ki sta v Sloveniji dobro razviti. Poudarili so problem vpliva politike na intenzivnejši razvoj e-uprave ter podali predloge za izboljšanje storitev. Kritično je bila ocenjena izvedba e-dohodnine ter njenih tehnoloških in vsebinskih pomanjkljivosti, ki so verjeten vzrok za skromno uporabo.
15. V sekciji *Internet, spletne rešitve* se je zvrstilo enajst predavanj v dveh sklopih. Pokrivala so različne teme s področja interneta in z njim povezanih tehnologij, kot so spletne storitve in njihovo povezovanje, informacijski agenti in konvergenca tehnologij. Še posebej je opazen premik k predavanjem o uporabi interneta s temami, kot so matrike, preizkušanje, optimizacija in zaščita, kot tudi vpliv na odnose z javnostmi in igralništvo.
16. V *Študentskem forumu* so predstavili svoje prispevke dodiplomski študenti informatike. Najboljši prispevek je bil nagrajen; nagrado je prejel Dejan Lavbič, študent Fakultete za računalništvo in informatiko Univerze v Ljubljani za prispevek *Povezava rezultatov iskanja spletnega inteligentnega agenta s podatki, pomembnimi za poslovne odločitve.*
17. Na posvetovanju so bila prisotna vsa pomembnejša slovenska podjetja in ustanove s področja informatike, predstavniki mnogih so bili aktivni v programu, drugi pa so se posvetovanja udeležili kot poslušalci. Veliko podjetij se je odločilo tudi finančno podpreti posvetovanje, za kar se vsem najlepše zahvaljujemo, saj brez njihove pomoči srečanja ne bi bilo mogoče izpeljati. Udeleženci ocenjujemo, da so take vsebinske usmeritve priporočljive tudi za prihodnja posvetovanja. Okrepi naj se sodelovanje med državo, gospodarskimi družbami, univerzo in nevladnimi organizacijami. Mednarodne stike in odnose na področju informatike je treba okrepiti še zlasti v luči članstva v Evropski uniji. Organizatorji naj uveljavljene prednosti posvetovanja kot neodvisnega strokovnega, znanstvenega in kulturnega dogodka razvijajo tudi v prihodnje.

ICS-2004 - 18th International Conference on Supercomputing	26. jun.–1. jul. 2004	St. Malo, Francija	ICS	http://ics04.irisa.fr
16th Euromicro Conference on Real-time Systems (ECRTS 04)	30. jun.–2. jul. 2004	Catania, Italija	Euromicro Technical Committee on Real-time Systems	http://www.diit.unict.it/ecrts2004
CAITA 2004	8.–11. jul. 2004	West Lafayette, Indiana, ZDA		http://www.internetconferences.net
8th World Multi-Conference on SYSTEMICS, CYBERNETICS And INFORMATICS - SCI 2004	18.–21. jul. 2004	Orlando, Florida, ZDA	SCI	http://www.iiis.org/sci2004/
IADIS International Conference e-society 2004	16.–19. jul. 2004	Avila, Španija		http://www.iadis.org/es2004
10th International Conference on Information Systems Analysis and Synthesis - ISAS 2004	21.–25. jul. 2004	Orlando, Florida, ZDA		http://www.infocybernetics.org/citsa2004/WebSite/Default.asp
CITSA 2004 - International Conference on Cybernetics and Information Technologies, Systems and Applications	21.–25. jul. 2004	Orlando, Florida, ZDA		http://www.infocybernetics.org/citsa2004/WebSite/Default.asp
2nd International Conference on Politics and Information Systems: Technologies and Applications - PISTA'04	21.–25. jul. 2004	Orlando, Florida, ZDA	PISTA	http://www.confinf.org/Pista04
IFIP World Computer Congress - WCC 2004	22.–27. avg. 2004	Toulouse, Francija	IFIP	http:/www.wcc2004.org dervillers@wcc2004.org
30th Euromicro Conference	31. avg.–3. sep. 2004	Rennes, Francija	University of Linz	http://www.euromicro.org
8th International Workshop on Software and Compilers for Embedded Systems – SCOPES 2004	2.–3. sep. 2004	Amsterdam, Nizozemska	EDAA	http://www.scopes2004.org
2nd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis	8.–10. sep. 2004	Stockholm, Švedska	IFIP	http://www.ida.liu.se/codes
CASES 2004 - International Conference on Compilers, Architecture and Synthesis for Embedded Systems	23.–25. sep. 2004	Washington D.C., ZDA		http://www.casesconference.org
IPSI - Internet, Processing, Systems and Interdisciplinaries	25. sep. 2004	Stockholm, Švedska	IPSI	http://www.internetconferences.net
Vzgoja in izobraževanje v informacijski družbi	15. okt. 2004	Ljubljana, Slovenija	FOV, IJS	http://lopes1.fov.uni-mb.si/is
e-Challenges e-2004 Conference and Exhibition	27.–29. okt. 2004	Dunaj, Avstrija	CEPIS	www.iimg.comwww.eChallenges.org
IPSI - Internet, Processing, Systems and Interdisciplinaries	31. okt. 2004	Milano, Italija	IPSI	http://www.internetconferences.net
WCCE 2005 - World Conference on Computers in Education	4.–7. jul. 2005	Južna Afrika	IFIP	http://www.wcce2005.org.za

PopoIn
E-Business Suite

Vse aplikacije zasnovane enotno.
Vse informacije na enem mestu.

ORACLE®

www.oracle.si

ISSN 1318-1882

9 771318 188001