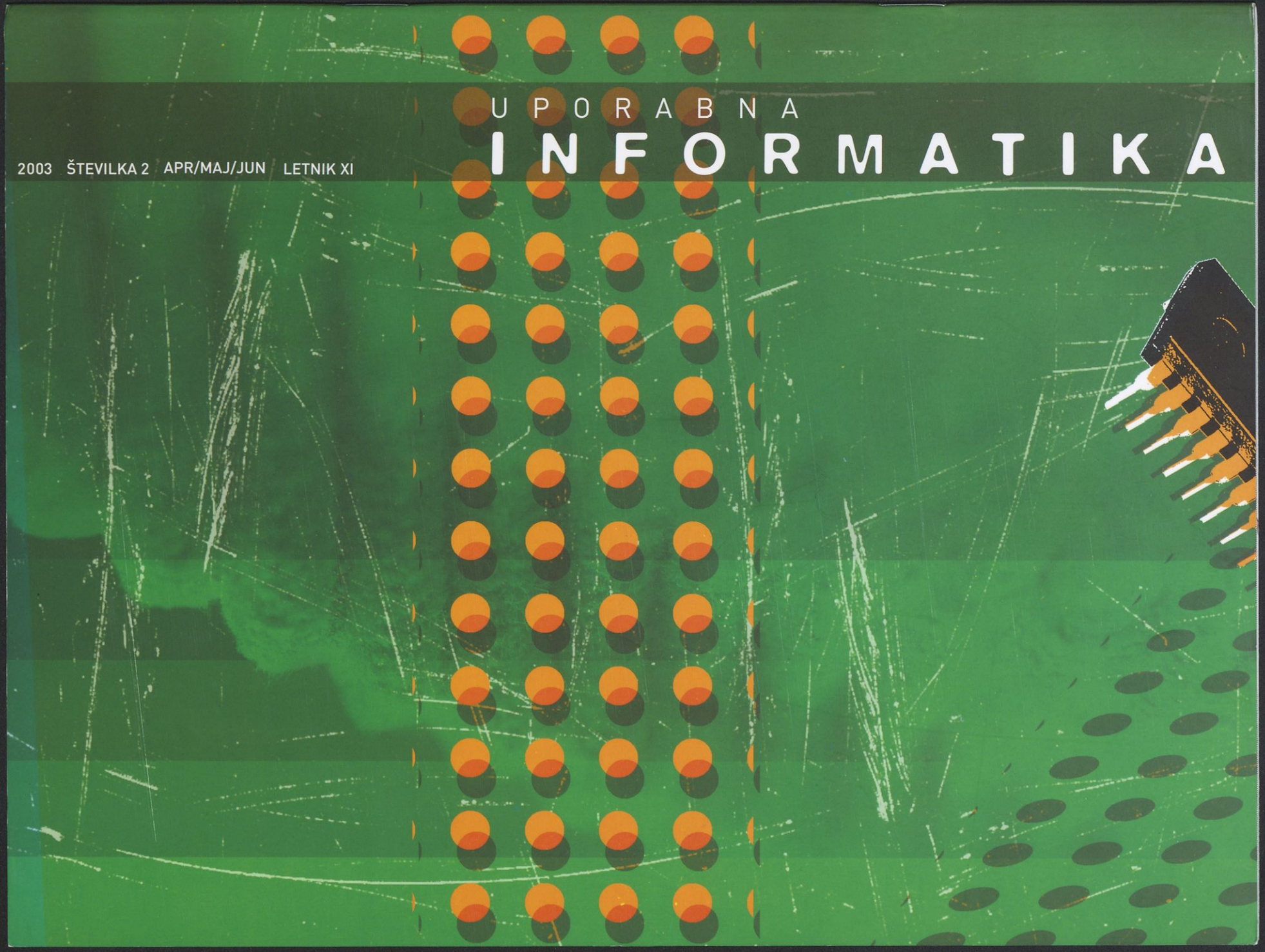
2003 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XI

UPORABNA INFORMATIKA



Testni centri ECDL

AGORA
ustvarjalne komunikacije

IZOBRAŽEVANJE
INFORMACIJSKE STORITVE

ELES - ICES

e-trade
Elit trade d.o.o. 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

LJUDSKA UNIVERZA
MURSKA SOBOTA

Micro Team

Much

spin

UPI LJUDSKA UNIVERZA ŽALEC
Šola prijaznih ljudi

VLS
computers

UPORABNA
INFORMATIKA
2003 ŠTEVILKA 2 APR/MAJ/JUN LETNIK XI ISSN 1318-1882

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 3.500 SIT. Letna naročnina za podjetja 13.800 SIT, za vsak nadaljnji izvod 8.900 SIT, za posameznike 4.600 SIT, za študente 2.000 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 mogoče 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 življenjepis avtorja (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, spoštovani bralci,

v Sloveniji se kar vrstijo konference in posvetovanja o informatiki. Organizatorji ugotavljajo, da na njih informatiki prepričujemo informatike o pomembnosti informatike in njenem vplivu na poslovno uspešnost, obenem pa tarnajo nad kronično neudeležbo menedžerjev.

Na splošno velja, da je pri nas podobno kot drugod po svetu večina (80 %) načrtovanih projektnih aktivnosti v naložbenem in vsebinskem pogledu neuspešnih. Informacijski projekti so uspešni, če ob načrtovanih vsebinskih, časovnih in stroškovnih parametrih vplivajo na poslovno uspešnost organizacije. Tega pa ni mogoče doseči zgolj z informatizacijo, temveč zlasti z razmislekom o strateških usmeritvah in premikih na področju menedžmenta in kadrov ter njihovega znanja, organiziranosti, poslovnih procesov in tudi informatike.

Velik delež krivde za neuspešno načrtovanje in razvoj informatike ter uporabo informacijske tehnologije gre pripisati menedžerjem, ki navadno kot edini pravi naročniki takšnih projektov ne poznajo vplivnosti informacijske tehnologije na poslovanje. Podatki kažejo, da največkrat ne poznajo svojih informacijskih potreb ali pa se ne zavedajo možnosti in priložnosti sodobne informacijske tehnologije in rešitev. Pobudniki informacijskih projektov so tako največkrat kar informatiki sami. Rezultati so največkrat parcialni, usmerjeni zgolj na področje učinkovitosti izvajanja procesov; takšni projekti dvigujejo stroške in pogosto ne prispevajo k poslovni uspešnosti. Med slovenskimi menedžerji je relativno malo takšnih pobudnikov, ki razumejo strateški pomen informatike in bi bili pripravljeni vlagati vanjo. Po raziskavi Inštituta za poslovno informatiko pri Ekonomski fakulteti v Ljubljani se jih le okoli 14 % zaveda pomena z informatiko podprte prenove poslovanja in uporabe informatike pri poslovnem odločanju. Okoli 7 % jih ugotavlja, da potrebujejo (in želijo) dodatna znanja s tega področja. Menedžerji torej največkrat strateške odločitve, postopke izbire ter uvajanja opreme in rešitev prepuščajo informatikom. Namesto da bi izrabili priložnost za korenito prenovo poslovanja v smislu dviga uspešnosti in povezljivosti (e-poslovanje), informatizirajo obstoječe, pogosto neurejene in za informatizacijo neprimerne poslovne procese. Seveda del krivde za neuspeh lahko pripišemo tudi zavajajočim ponudnikom informacijske opreme in "lastnim" informatikom.

Poslovna uspešnost organizacije je neposredno odvisna od uveljavljanja in zagotavljanja strateške vloge informatike. Trenutna percepcija in ravnanje menedžerjev je na področju informatike pretežno stroškovno naravnano, od informatike v večini primerov niti ne pričakujejo večje učinkovitosti in preglednosti izvajanja poslovnih procesov, poslovna uspešnost je drugotnega pomena ali po njihovem mnenju nedosegljiva. Ključno izhodišče sodobnega menedžmenta za zagotavljanje poslovne uspešnosti mora izhajati iz spremenjene percepcije menedžerja o informatiki, od informacijske podpore oddelka/poslovne funkcije do strateške vplivnosti na poslovanje. Transformacija v globalno poslovanje zahteva drugačno vlogo informatike in službe za informatiko v organizaciji ter novo obliko partnerstva med menedžmentom in informatiko.

Vloga in pomembnost informatike v organizaciji sta pogojena s stopnjo odvisnosti od informacij in percepcijo menedžmenta o njeni vplivnosti na poslovanje. V slovenskih organizacijah se po našem prepričanju žal informacijska tehnologija uporablja predvsem v podporo poslovanju, močno pa je zapostavljena njena naložbena oblika oziroma strateška vloga pri poslovni uspešnosti organizacij. Koreniti premiki so na tem področju nujni, zanje potrebujemo zlasti drugačen odnos menedžmenta do naložb v informatiko ter neposredno vključevanje informatikov v strateško poslovno načrtovanje in odločanje, kar pogojuje drugačno organiziranost službe za informatiko in drugačen položaj vodje te službe v organizacijski strukturi.

V uredniškem odboru revije Uporabna informatika se bomo še posebej trudili predstavljati in obravnavati navedeno problematiko. Z veseljem bomo objavljali prispevke, namenjene osveščanju informatikov in menedžerjev o strateškem pomenu in vplivu informatike na poslovanje. Vabljeni k sodelovanju.

Andrej Kovačič,
odgovorni urednik

Vabilo avtorjem k pripravi prispevkov za tematsko številko revije Uporabna informatika

Tema: Informacijska varnost

Obseg podatkov v elektronski obliki nezadržno narašča; za okrog 250 megabytov letno na prebivalca našega planeta. Čeprav je že 95 % vseh dokumentov vsaj v eni fazi obdelanih v elektronski obliki, jih je le 3 % shranjenih v elektronskih arhivih.

Z vse večjim obsegom raste tudi potreba po zagotavljanju zasebnosti in varnosti pri elektronskem poslovanju ter zaščiti podatkov. Na eni strani lahko spremljamo razvoj standardov, na primer znanega BS 7799 za področje informacijske varnosti, ki je leta 2000 dobil tudi mednarodnega naslednika ISO/IEC 17799, na drugi pa razvoj tehnologije, ki to omogoča. Predstavniki Intela na vprašanje, zakaj potrebujemo vse zmogljivejše mikroprocesorje, odgovarjajo, da preprosto zato, ker bo v prihodnje 90 % procesorske moči namenjene varovanju in zaščiti podatkov pri različnih oblikah prenosa.

Kje smo na področju informacijske varnosti v Sloveniji? Na vprašanje bomo skušali odgovoriti v posebni številki revije Uporabna informatika. Prepričani smo, da ni dovolj, da se pomena informacijske varnosti zavedamo in pri tem računamo na to, da bodo komercialne rešitve kmalu naprodaj. Sprašujemo se, kakšne so znanstvene in razvojne rešitve na tem področju? S katerim znanjem razpolagata danes naši univerzi? Ali lahko v prihodnosti pričakujemo domače rešitve, s katerimi bomo lahko osvojili svoj delež svetovnega trga?

Vaše prispevke pričakujemo z željo, da Uporabna informatika ostane eden izmed trdnejših mostov med stroko in prakso.

Področju informacijske varnosti se v bančništvu, zavarovalništvu, zdravstvu in javni upravi že doslej nismo mogli izogniti in zagotovo razpolagamo z ustreznimi izkušnjami in znanjem. Kakšno je stanje v gospodarstvu? Kako ste k zagotavljanju informacijske varnosti pristopili v vaši organizaciji? Vabimo vas, da predstavite konkretne rešitve in koncepte, ki jih vpeljujete v vašem okolju.

Informacijska varnost predstavlja temelj za nadaljnje poslovne korake in je zagotovo aktualna tema v trenutku, ko tudi v Sloveniji vse več uporabljamo svetovni splet, mobilne komunikacije in elektronsko poslovanje.

Vaše prispevke z oznako »Tema2003« pričakujemo na e-naslov: ui@drustvo-informatika.si

do 1. septembra 2003.

Aljoša Domijan,
gostujoči urednik

Analiza varnostne ogroženosti

Matjaž Jaušovec, Boštjan Brumen, Tatjana Welzer - Družovec
Fakulteta za elektrotehniko, računalništvo in informatiko Univerze v Mariboru
Matjaz_Jausovec@hotmail.com

Izvleček

Podjetja, ki pri svojem poslovanju ne uporablja računalniškega sistema, si ne moremo več predstavljati. Med uporabo računalniškega sistema lahko pride do neželenega izpada ali nepravilnega delovanja strojne ali programske opreme, večkrat tudi zaradi nepravilne uporabe. Podjetja zaradi tega izgubljajo denar, stranke, naročila in ugled. Zato je smiselno, da izračunajo stroške neželenih dogodkov in da z določenimi tehnikami preprečijo take dogodke.

V prvem delu članka predstavljamo tehniko analize ogroženosti, s katero lahko ugotovimo in zmanjšamo ranljivost računalniškega sistema. V drugem delu članka pa predstavljamo analizo ogroženosti v enem izmed slovenskih podjetij.

Abstract

Analysis of Security Risk

It is beyond our comprehension to imagine a company operating without the use of the information technology. However, the use of computer system involves unforeseen events, such as irregular operation of the hardware and software equipment, mostly as the consequence of inappropriate handling. In such cases a company loses its money, clients, placed orders, its reputation, to mention just a few. Therefore it is highly recommended to estimate the consequences. In such a way a company adopts a technique by which the occurence of the unforeseen events is prevented.

The first part of the article describes a technique called »Analysis Of Security Risk«, which helps to establish and decrease the sensitivity factor of computer system. The second part of the article presents the actual analysis of the security risks, carried out in one of the Slovene companies.

1 Analiza ogroženosti

Načrtovanje varnosti računalniških sistemov se začne z analizo ogroženosti. To je postopek, s katerim določimo izpostavljenost računalniškega sistema in izgubo denarja, ki jo lahko povzroči neželeni dogodek. Prvi korak analize je določitev in opis vseh oblik izpostavljenosti računalniškega sistema, ki se kažejo kot izpad ali nepravilno delovanje strojne ali programske opreme, kar je večkrat tudi posledica nepravilne uporabe, nato pa za vsako od njih določitev ravnanja, ki ga je treba tudi ovrednotiti. Pri tem je vsekakor pomembno tudi dejstvo, da imajo prednost tiste rešitve, ki prinašajo največ koristi (primerjava stroškov in koristi za vse rešitve). Zadnja stopnja analize je torej analiza stroškov in koristi (angl. cost-benefit), ki nam pove, ali se srečamo z nižjimi stroški, če ukrepamo, ali pa je morda ceneje sprejeti pričakovano izgubo. [1]

Ustrezno ravnanje lahko zmanjšajo ogroženost in posredno izgubo. Smiselno je na primer shraniti pomembnejše podatke z diskov, da zaradi okvare ne ostanemo povsem brez njih [3]. Osebno shranjevanje podatkov v večjih podjetjih ni najprimernejše, treba je uvesti ukrepe, ki bi ta problem rešili na ravni podjetja. Z analizo ogroženosti na primer lahko odgovorimo na vprašanje, ali so stroški ob izgubi podatkovnih vsebin zaradi okvare diska večji od stroškov, namenjenih za ukrepe, ki bi to izpostavljenost reševali drugače.

V nadaljevanju predstavljamo razloge za analizo ogroženosti in korake te analize.

1.1 Razlogi za analizo ogroženosti

Nekateri razlogi za analizo ogroženosti so:

1. *Pozornost.* Seznanitev uporabnikov s pomembnostjo varnosti lahko poveča zanimanje in odgovornost zaposlenih.
2. *Ugotavljanje prednosti, pomanjkljivosti in ukrepanja.* Veliko podjetij se ne zaveda svojih prednosti in pomanjkljivosti. S sistematično analizo pridobimo razumljivo predstavo [5].
3. *Dvig osnove za odločanje.* Ukrepi preprečujejo ustvarjalnost uporabnikov, večkrat neupravičeno. Velikokrat pa zaradi pomanjkanja ukrepov zaznamo večjo ustvarjalnost, ki lahko poveča stopnjo tveganja. Da bi izbrali primerne ukrepe, moramo zelo dobro poznati ali oceniti stopnjo tveganja.
4. *Upravičenost izdatkov za varnost.* Nekateri varnostni ukrepi so zelo dragi in nimajo posebne prednosti. Prikazati moramo, koliko stane ukrep in za koliko se zmanjša tveganje, če ukrepamo.

1.2 Koraki opravljanja analize ogroženosti

Proces analize ogroženosti je prevzet iz upravljanja (angl. management). Vsebine, ki jih obravnavamo, se nanašajo na računalniški sistem. Temeljni koraki analize ogroženosti so [1]:

1. Ugotavljanje sredstev (angl. assets).
2. Določitev možne ranljivosti sredstev.
3. Ocena verjetnosti izrabe (angl. exploitation) sredstev. Ocenimo verjetnost nastopa situacij, v katerih je sredstvo ranljivo.
4. Izračun pričakovane letne škode.
5. Pregled primernih ukrepov za preprečevanje nastajanja škode in stroškov uporabe.
6. Načrtovanje letnega prihranka ob uveljavitvi novih ukrepov.

1.2.1 Ugotavljanje sredstev

Prvi korak analize ogroženosti je popis sredstev. Sredstva lahko razdelimo v kategorije. Prve tri so povsem računalniške, preostale tri pa niso nujno del računalniškega sistema, vendar so pogoj za primerno delovanje [2].

1. *Strojna oprema*: računalniki, monitorji, terminali, delovne postaje, tračne enote, tiskalniki, diski, kabli, povezave, komunikacijski vmesniki.
2. *Programska oprema*: izvorna koda, objektni programi, kupljeni programi, programi v hiši, aplikacijski programi, operacijski sistemi, sistemski programi (prevajalniki) in vzdrževalno-diagnostični programi.
3. *Podatki*: podatki, ki jih uporabljajo programi, podatki, shranjeni na magnetnih nosilcih, arhivirani podatki, tiskani podatki in poročila o osveževanju podatkov.
4. *Zaposleni*: potrebni za delovanje računalniškega sistema ali programov.
5. *Dokumentacija*: za programske in strojno ter administrativno proceduro.
6. *Dobava pisarniškega materiala*: papir, kartuše, nosilci za shranjevanje podatkov.

Z izdelavo takega seznama smo pravzaprav opravili inventuro informacijskih sredstev.

1.2.2 Določitev ranljivosti sredstev

Ugotavljanje sredstev je preprost postopek, ker so sredstva večinoma opredmetena. Ugotavljanje ranljivosti sredstev zahteva veliko več predvidevanj, če hočemo ugotoviti, kolikšno škodo lahko utrpijo sredstva in kateri vir jo povzroči.

Trije temeljni cilji varnega računalniškega sistema so:

- tajnost, zaupnost (angl. secrecy),
- popolnost, neokrnjenost, celovitost, povezljivost (angl. integrity),
- razpoložljivost (angl. availability).

Ranljivost je vsaka situacija, ki lahko povzroči izgubo katerekoli od teh treh lastnosti.

Smiselno je narediti primerjavo med sredstvom in njegovo ranljivostjo. Ta primerjava je razvidna iz naslednje tabele:

Sredstva	Tajnost/ zaupnost	Popolnost/ neokrnjenost/ celovitost/ povezljivost	Razpoložljivost
Strojna oprema			
Programska oprema			
Podatki			
Zaposleni			
Dokumentacija			
Dobava pisarniškega materiala			

Za vsako kombinacijo sredstva in njegove ranljivosti moramo ugotoviti, kaj neželenega se lahko zgodi. Pomagamo si lahko z naslednjimi vprašanji:

- Kakšne so posledice nenamernih napak?
- Kakšne posledice lahko povzročijo namerno narejene napake zaposlenih?
- Kako na sistem vplivajo zunanji dejavniki?
- Kakšne so posledice naravnih nesreč in fizičnih okvar?

1.2.3 Ocena verjetnosti izrabe sredstev

V tem koraku analize ogroženosti podajamo svoja pričakovanja o pogostosti pojava neželenega stanja. Upoštevamo naslednje:

- Verjetnost dogodkov pri splošni populaciji. Največ teh podatkov imajo zavarovalnice.
- Verjetnost dogodkov, ki se nanašajo na opazovanje posameznega sistema.
- Ocenitev števila dogodkov v določenem času.
- Ocenitev verjetnosti iz tabele (navedeni sta pogostost pojavljanja in ocena).
- Delfi raziskava. Med strokovnjaki obravnavanega področja poskušamo ugotoviti splošno mnenje.

1.2.4 Izračun pričakovane letne škode

Določiti moramo pričakovane stroške vseh neželenih dogodkov. Stroške menjave strojne opreme določimo hitro, medtem ko je težje določiti stroške menjave dela programa. Stroške drugih področij večinoma lahko ocenimo na podlagi izkušenj.

Pravna pravila določajo, katere podatke je treba zaščititi zaradi tajnosti oziroma zaupnosti poslovanja podjetja. Stroške, ki bi nastali ob nedovoljeni objavi takih podatkov, je težko določiti.

Neprijetne posledice lahko nastanejo, če računalniški sistem ali programska oprema ne deluje ali če manjka ključna oseba.

Višino stroškov lahko določimo z naslednjimi vprašanji:

- Kakšne so pravne obveznosti do popolnosti, celovitosti, povezljivosti in tajnosti oziroma zaupnosti podatkov?
- Ali bi objava podatkov povzročila škodo posamezniko m ali podjetjem? So pravne posledice predvidljive?
- Ali lahko nedovoljen dostop do podatkov povzroči izgubo poslovnih priložnosti? Ima konkurent zaradi tega prednost? Kakšna je izguba pri prodaji?
- Kakšen je psihološki učinek na delovanje računalniškega sistema? Lahko povzroči zadrego, izgubo verodostojnost in posla? Koliko strank bo oškodovanih? Kakšna vrednost je povezana s temi strankami?
- Koliko nam pomeni dostop do podatkov? Lahko obdelavo podatkov (angl. computation) preložimo za časovno predvidljivo obdobje? Jo lahko opravimo kje drugje? Koliko bi to stalo?
- Kakšno vrednost vidi konkurent v naših podatkih? Koliko sredstev je pripravljen vložiti, da pridobi te podatke?
- Kakšne težave povzroči izguba podatkov? Lahko podatke nadomestimo ali obnovimo in koliko dela moramo v to vložiti?

Tovrstne stroške je težko oceniti. Ogroženost računalniškega sistema je večinoma veliko večja, kot jo pričakuje vodstvo.

Vrednost neželenega dogodka določimo na opisani način. Pomnožimo jo s pričakovanim številom ponovitev dogodka v enem letu. Tako ovrednotimo pričakovane letne stroške (angl. annual loss expectancy).

1.2.5 Pregled primernih ukrepov za preprečevanje nastajanja škode in stroškov uporabe

Za trenutno znane in veljavne ukrepe za preprečevanje nastajanja škode poznamo pričakovane stroške. Če so stroški nesprejemljivo visoki, moramo poiskati nove ukrepe, s katerimi jih bomo znižali.

Eden izmed načinov identifikacije dodatnih ukrepov temelji na izpostavljenosti. Pred izgubo podatkov se lahko zavarujemo z varnostnimi kopijami, s podvajanjem, z nadzorom dostopa, ki preprečuje brisanje, s fizično zaščito, ki preprečuje krajo diska, ali s programi, ki preprečujejo dostop do podatkov. [3]

Drugi načini se nanašajo na pregled poznanih ukrepov za preprečevanje nastajanja škode [1]:

- vtajnopisje (angl. cryptographic controls),
- varni protokoli,
- nadzor pri razvoju programov,
- nadzor okolja pri izvajanju programov,
- uporaba varnostnih ukrepov operacijskega sistema,
- identifikacija,
- verodostojnost,
- načrtovanje in razvoj varnega operacijskega sistema,
- nadzor dostopa do baze podatkov [3],
- zanesljivost dostopa do baze podatkov,
- uporaba večplastnih varnostnih ukrepov za podatke, podatkovne baze in operacijske sisteme,
- varnost osebnih računalnikov,
- varnostni ukrepi za dostop in uporabo računalniške mreže,
- varnost fizičnih dostopov.

Poznavanje mehanizmov in primernost za uporabo sta ob določeni nevarnosti temeljna pogoja za pravilno izbiro oziroma odločanje o uporabi. S tem zmanjšamo tudi stroške odločanja in druge stroške, ki bi lahko nastali v povezavi z omenjenim področjem.

1.2.6 Načrtovanje letnega prihranka ob uveljavitvi novih mehanizmov upravljanja

Naslednji korak v procesu opravljanja analize ogroženosti je izračun, ali smo z novimi ukrepi za preprečevanje nastajanja škode in njihovimi stroški v boljšem položaju kot v sedanjem stanju. Po izračunu pridemo do enega izmed treh odgovorov:

1. Novi mehanizmi upravljanja povečajo varnost, vendar so stroški višji kot izguba neželenega dogodka.
2. Novi mehanizmi upravljanja povečajo varnost, stroški pa so manjši, kot izguba zaradi neželenega dogodka.
3. Novi mehanizmi upravljanja ne povečajo varnosti.

2 Analiza ogroženosti v enem izmed slovenskih podjetij

Podjetje, ki smo ga analizirali, je eno iz skupine sorodnih podjetij na slovenskem področju. Omenjena skupina je ustanovila hčerinsko podjetje, ki opravlja del informacijskih storitev (pretežno prek integriranega informacijskega sistema), drugi del informacijskih storitev, ki so povezana s posebnostmi poslovanja teh podjetij, pa podjetja opravljajo sama.

Analizirano podjetje ima upravo in devet poslovnih enot na področjih, ki jih podjetje pokriva s svojim poslovanjem. Analizo ogroženosti smo opravili s stališča službe za informatiko.

Ker bi iz nadaljevanja lahko razbrali, za katero podjetje gre, smo navedene številske vrednosti prilagodili. Navajanje dejanskih vrednosti bi namreč pomenilo izdajanje poslovne skrivnosti podjetja.

V nadaljevanju podrobneje opisujemo naslednje korake:

1. Ugotavljanje sredstev.
2. Določitev ogroženosti sredstev.
3. Ocena verjetnosti izrabe sredstev.
4. Izračun pričakovane letne škode.
5. Pregled primernih ukrepov za preprečevanje nastajanja škode in stroškov uporabe.
6. Načrtovanje letnega prihranka ob uveljavitvi novih ukrepov.

2.1 Ugotavljanje sredstev

Prvi korak pri izvajanju analize ogroženosti je popis sredstev.

1. Strojna oprema:

Oprema	Število
TERMINAL	14
MONITOR	414
RAČUNALNIK	448
OPTIČNI BRALNIK	22
TISKALNIK	222
ROUTER	13
PREKLOPNIK	11
MODEM	3
STREŽNIK	11

Pri računalnikih in monitorjih prihaja do večje razlike, ker so k računalnikom prišteti tudi tanki odjemalci [2].

2. Programska oprema

Programska oprema	Število
WINDOWS 95	130
WINDOWS 98	190
WINDOWS NT	17
WINDOWS 2000	25
OFFICE 97 (Word, Excel, Access, Outlook)	350
PERSONAL COMMUNICATION	350
DB/2	110
FOREST&TREES	110
VISIO TECHNICAL	35
NORTON (protivirusni program)	350
ON DEMAND	65
AUTOCAD	18
ARCVIEW	2
MS PROJECT	3
LOTUS NOTES	150
APPS – pošiljanje in spremljanje transakcij na APP (Agencija za plačilni promet)	1
Elmark – trgovanje	10

3. Podatki

Za podatke po pogodbi o opravljanju storitev skrbi drugo podjetje. Obravnavamo jih zato, ker bi odtujitev podatkov povzročila velike finančne posledice. Mislimo na vrednost informacije, ki se skriva za podatki.

4. Zaposleni

Za nemoteno delovanje računalniškega sistema skrbi služba za informatiko. V njej je zaposlenih sedem ljudi. Stopnja izobrazbe uporabnikov informacijskih storitev v podjetju (tj. zaposlenih) je taka:

Izobrazba	Število uporabnikov
POLKVALIFICIRAN	6
KVALIFICIRAN	20
SREDNJA IZOBRAZBA	162
VIŠJA IZOBRAZBA	103
VISOKA in več	72

5. Dokumentacija

Dokumentacijo strojne in programske opreme hrani služba za informatiko. Dokumentacija IIS (Integrirani IS) je v domeni drugega podjetja.

6. Dobava pisarniškega materiala

Dobavitelja pisarniškega materiala v podjetju izbirajo enkrat na leto na podlagi javnega razpisa. Predstavnik službe za informatiko sodeluje ob pripravi razpisne dokumentacije. Izkušnje kažejo, da je dobavitelj pripravljen upoštevati morebitne spremembe in želje naročnika, saj se zaveda, da se izbor oz. javni razpis ponavlja vsako leto.

2.2 Določitev ranljivosti sredstev

Glej tabelo 1.

2.3 Ocena verjetnosti izrabe sredstev

Za razumevanje postopka ocene verjetnosti izrabe sredstev je dovolj, da predstavimo tabelo strojne opreme. Druge tabele izdelujemo po podobni shemi. Vsekakor pa za nadaljevanje postopkov potrebujemo še preostale tabele, ki jih ne prikazujemo (programska oprema, podatki, zaposleni, dokumentacija, dobava pisarniškega materiala).

Pri oceni verjetnosti izrabe smo se odločili za naslednje vrednotenje: številka v razpredelnici označuje število ponovitev pojava (frekvenco) v enem letu. Glej tabelo 2.

2.4 Izračun pričakovane letne škode

Kot v prejšnji točki tudi tukaj predstavljamo tabelo strojne opreme. Tabela 3 na naslednji strani.

Podatke o vrednotenju posameznega izrednega dogodka navajamo v tabeli. Številska vrednost v tabeli pomeni vrednost v evrih, ki jo izgubljamo ob enkratnem izrednem dogodku.

Tabela 1

Sredstva	Tajnost / zaupnost	Popolnost/neokrnjenost/ celovitost/povezljivost	Razpoložljivost
Strojna oprema		Preobremenjena/uničena	Nepravilno delovanje/ukradena/okvara/nedostopna
Programska oprema	Ukradena/prekopirana/ zlorabljena	Nepravilno delovanje/ nedovoljeno nadzorovana (angl. trojan)	Izbrisana/ni na pravem mestu/pretek licenc
Podatki	Objava/ dostop nepooblaščenim	Spremenjeni zaradi napake strojne ali programske opreme in uporabnika	Izbrisani/niso na pravem mestu/uničeni
Zaposleni bolniška		Bolniška	Prenehajo z delom/so upokojeni/na dopustu/
Dokumentacija			Izgubljena/ukradena/uničena
Dobava pisarniškega materiala			Izgubimo/ukradeno/uničeno

Tabela 2

Oprema	Preobremenjena	Uničena	Nepravilno deluje	Ukradena	Nedostopna	Število
TERMINAL	3	0,5	2	0,1	5	14
MONITOR	2	0,5	5	0,1	10	414
RAČUNALNIK	100	1	120	0,1	50	448
OPTIČNI BRALNIK	30	0,2	12	0	5	22
TISKALNIK	70	1	60	0,1	120	222
ROUTER	1	0,5	5	0,1	5	13
PREKLOPNIK	10	0,5	30	0,05	20	11
MODEM	5	0,1	2	0,1	4	3
STREŽNIK	10	0,1	20	0,01	20	11

V tabeli 4 je predstavljena dejanska vrednost, ki jo izgubljamo glede na pričakovano število ponovitev izjeme. Dobimo jo tako, da pomnožimo vrednosti iz tabele, ki prikazuje oceno verjetnosti izrabe sredstva, z vrednostmi iz tabele, ki prikazuje vrednotenje posameznega izrednega dogodka.

V tabeli 5 smo zbrali rezultate prikazanih in neprikazanih tabel. Izračun kaže, da pri posamezni vrsti sredstva izgubljamo naslednje zneske:

Iz tabele je razvidno, da je znesek za strojno opremo zelo visok. Ugotovimo, da k temu precej prispeva nepravilno delovanje računalnikov in preklopnikov.

Tabela 5

Strojna oprema	**10.900.519 EUR**
Programska oprema	8.627.000 EUR
Podatki	6.760.000 EUR
Zaposleni	483.000 EUR
Dokumentacija	128.000 EUR
Dobavljeno	3.000 EUR
Skupaj	26.901.519 EUR

Tabela 3

Oprema	**Preobremenjena**	**Uničena**	**Nepravilno deluje**	**Ukradena**	**Nedostopna**
TERMINAL	2.000 EUR	120.000 EUR	5.000 EUR	120 EUR	5.000 EUR
MONITOR	1.000 EUR	70.000 EUR	5.000 EUR	70 EUR	1.000 EUR
RAČUNALNIK	5.000 EUR	210.000 EUR	10.000 EUR	300.000 EUR	10.000 EUR
OPTIČNI BRALNIK	1.000 EUR	60.000 EUR	4.000 EUR	65.000 EUR	2.000 EUR
TISKALNIK	5.000 EUR	70.000 EUR	6.000 EUR	80.000 EUR	3.000 EUR
ROUTER	50.000 EUR	1.500.000 EUR	70.000 EUR	1.700.000 EUR	70.000 EUR
PREKLOPNIK	30.000 EUR	1.000.000 EUR	40.000 EUR	1.050.000 EUR	40.000 EUR
MODEM	20.000 EUR	50.000 EUR	20.000 EUR	70.000 EUR	20.000 EUR
STREŽNIK	40.000 EUR	700.000 EUR	40.000 EUR	1.500.000 EUR	50.000 EUR

Tabela 4

Oprema	**Preobremenjena**	**Uničena**	**Nepravilno deluje**	**Ukradena**	**Nedostopna**
TERMINAL	6.000 EUR	60.000 EUR	10.000 EUR	12 EUR	25.000 EUR
MONITOR	2.000 EUR	35.000 EUR	25.000 EUR	7 EUR	10.000 EUR
RAČUNALNIK	500.000 EUR	210.000 EUR	1.200.000 EUR	30.000 EUR	500.000 EUR
OPTIČNI BRALNIK	30.000 EUR	12.000 EUR	48.000 EUR	0 EUR	10.000 EUR
TISKALNIK	350.000 EUR	70.000 EUR	360.000 EUR	8.000 EUR	360.000 EUR
ROUTER	50.000 EUR	750.000 EUR	350.000 EUR	170.000 EUR	350.000 EUR
PREKLOPNIK	300.000 EUR	500.000 EUR	1.200.000 EUR	52.500 EUR	800.000 EUR
MODEM	100.000 EUR	5.000 EUR	40.000 EUR	7.000 EUR	80.000 EUR
STREŽNIK	400.000 EUR	70.000 EUR	800.000 EUR	15.000 EUR	1.000.000 EUR
Vsota	1.738.000 EUR	1.712.000 EUR	4.033.000 EUR	282.519 EUR	3.135.000 EUR 10.900.519 EUR

2.5 Pregled primernih ukrepov za preprečevanje nastajanja škode in stroškov uporabe

Na dveh stroškovno največjih težavah prikazujemo možnost vpeljave nove kontrole.

1. Nepravilno delovanje računalnikov = 1.200.000 evrov.

Ugotovimo, da je vpisana sorazmerno visoka številka ponavljanja problema 120. Ocenjena vrednost ob nastopu pojava je 10.000 evrov. Zakaj nastajajo težave? Ugotovili smo, da se večkrat pojavljajo problemi v računalnikih, čeprav so ti menjali lokacijo in uporabnika. Poskusi odprave »čudnega delovanja« na servisu niso obrodili sadov.

Predlog: Dogovor z dobaviteljem, da zamenja opremo. Tako bi se v prihodnje zmanjšalo število ponavljanj za polovico.

2. Nepravilno delovanje preklopnikov = 1.200.000 evrov.

Težava se pojavi 30-krat na leto, kar stane pa 40.000 evrov. Do problema prihaja zaradi nejasne odpovedi strojne opreme na prehodu iz mreže token-ring v mrežo ethernet in med povezavo med preklopniki.

Prvi del problema je nastajal zaradi globalnosti sistema, ko strojna oprema ni bila združljiva z drugo strojno opremo na povsem drugem mestu. Problem je mogoče rešiti s podrobnim pregledom nastavitev naprav.

Za drugi del problema je kriva različna verzija biosa enakih naprav, kupljenih v obdobju šestih mesecev. Problem odpravimo s posodobitvijo biosa.

Znesek, ki ga ob nastopu problema izgubljamo, je ocenjen na 40.000 evrov. Ker je posredno prizadetih veliko računalnikov z velikim številom procesov, je znesek temu primerno visok. Ob odpravi napak pričakujemo znižanje števila pojavljanj na 3. Znesek ob enkratnem pojavu neželenega dogodka se ne zmanjša.

2.6 Načrtovanje letnega prihranka ob uveljavitvi novih mehanizmov upravljanja

Izračunajmo zmanjšanje ovrednotenega tveganja ob upoštevanju teh dveh rešitev.

Ovrednotenje rešitve prvega problema: glede na to, da nas dogovor z dobaviteljem (morda zapisan v razpisnih pogojih) ne bo stal nič, je ovrednoteno tveganje 60 ponavljanj x 10.000 evrov = 600.000 evrov. Od prvotnega zneska 1.200.000 evrov odštejemo novi znesek 600.000 evrov in dobimo razliko 600.000 evrov, ki pomeni prihranek.

Rešitev drugega primera nam ovrednoteno pomeni še več. Rešitev nas stane 3 x 40.000 evrov = 120.000 evrov. Brez rešitve 1.200.000 evrov – rešitev 120.000 evrov = prihranek 1.080.000 evrov.

Ugotovimo, da smo z rešitvama dveh težav prihranili 1.680.000 evrov.

Mogoča izguba se je s 26.901.519 evrov zmanjšala na 25.221.519 evrov.

3 Ugotovitve

Ugotavljamo, da je tehnika uporabna in da odgovarja na vprašanje, kje je računalniški sistem glede varnosti najbolj ranljiv.

Med opravljanjem analize ogroženosti smo si zastavili naslednja vprašanja. Analiza pogosto sloni na ocenah. Je rezultat, ki je zmnožek ocen, pravilen? Ugotovili smo, da imamo raje rezultate, ki temeljijo na teoretskih predpostavkah, kot pa da jih sploh ni. Pozitivna lastnost tehnike je, da razbija problem na posamezne dele, ki jih lahko rešujemo posamično in neodvisno od drugih.

Brez zadržka bi tehniko priporočili vsem, ki želijo ugotoviti in izboljšati varnost računalniškega sistema; ocenjujemo jo kot zelo uporabno predvsem v okolju z velikimi tveganji.

Literatura

[1] Pfleeger, C. P.: (1996): Security in computing, Prentice-Hall, Inc., Upper Saddle River, New Jersey.

[2] EMRIS – Enotna metodologija razvoja informacijskih sistemov, Vlada Republike Slovenije, Center za informatiko, Ljubljana.

[3] Ozsu, M. T., Valduriez, P.: Priciples of Distributed Database Systems, Prentice-Hall PTR, Upper Saddle River, New Jersey.

[4] Subrahmanian, V. S.: Principles of Multimedia Database Systems, Morgan Kaufmann Publishers, Inc, San Francisco, California.

[5] Adriaans, P., Zantinge, D.: Data Mining, Addison Wesley Longman Limited, Harlow, England.

Matjaž Jaušovec je diplomiral leta 1996 na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Je študent magistrskega študija na omenjeni fakulteti. Pri raziskovalnem delu se ukvarja predvsem z vrednotenjem podatkov.

Boštjan Brumen je doktorski študent na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru, kjer je zaposlen kot asistent za področje informatika. Raziskovalno se ukvarja s podatkovnimi bazami in podatkovnim rudarjenjem.

Tatjana Welzer - Družovec je izredna profesorica na Fakulteti za elektrotehniko, računalništvo in informatiko v Mariboru, kjer predava na dodiplomski in podiplomski stopnji in vodi laboratorij za podatkovne tehnologije. Raziskovalno se ukvarja predvsem s podatkovnimi bazami, kakovostjo podatkov in podatkovnim modeliranjem.

Agilne metodologije razvoja informacijskih sistemov

Marko Bajec, Marjan Krisper
Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Tržaška 25, 1000 Ljubljana
marko.bajec@fri.uni-lj.si, marjan.krisper@fri.uni-lj.si

Povzetek

Pojem agilnost in trendi, ki jih s tem povezujemo, so se na področju metodologij razvoja informacijskih sistemov pojavili kot rezultat kritičnega pogleda na celovite, večinoma težko obvladljive metodologije, ki proces razvoja predpišejo do vsake najmanjše podrobnosti. Kljub kakovosti, ki jo takšne metodologije navadno dosegajo, je njihova uporaba v praksi omejena, saj jim z željo po čimprejšnjih rezultatih težko sledimo. Novi trendi priporočajo zasnovo in uporabo prilagodljivih metodologij, ki so »ravno dovolj« predpisljive ter hitro in enostavno prilagodljive. V prispevku opredelimo pojem agilnost in opišemo z njo povezane trende. V nadaljevanju podamo naše izkušnje pri uporabi novih trendov za zasnovo sodobnih metodologij razvoja informacijskih sistemov.

Abstract

Agile Information System Development Methodologies

The term "agile" and trends that are related to it have emerged in the IS community as a result to shortcomings of the complex methodologies which prescribe each development step in its most detail. In spite of reputation such methodologies gain in theory their practical use is only limited. Today's markets are extremely dynamic and call for rapid results which make the use of heavy methodologies difficult. New trends recommend the use of agile methodologies that lead into so called adaptable software development. In the paper we describe the concept of an agile methodology and discuss our experiences in developing a contemporary methodology based on agile-oriented values, principles and practices.

1 UVOD

Prve elaborirane metodologije razvoja informacijskih sistemov so nastale že v zgodnjih sedemdesetih letih, ko so na področju razvoja programske opreme kraljevali Codd, DeMarco, Yourdon, Cox, Booch, Orr, Parnas in drugi. S tem, ko so uspeli prepričati strokovno javnost, da je potrebno tudi za področje razvoja programske opreme urediti postopke, predpisati metode, tehnike in orodja, so ovrgli smiselnost uporabe ad hoc pristopov in postavili temelje za obdobje birokracije, kot ga danes nekateri imenujejo (7). Kasneje so bile namreč razvite in dokumentirane številne metodologije, bodisi na podlagi izkušenj ali teoretičnih izhodišč. Z željo po standardizaciji postopkov in izdelkov so nastali priročniki, ki so tudi na več tisoč straneh do potankosti predpisovali vsak najmanjši korak razvoja, za vsako aktivnost so bili predpisani številnimi dokumenti in formularji, za njihovo izvedbo pa so bila določena orodja in navodila za uporabo.

Breme, ki ga nosijo udeleženci, če morajo pri projektu slediti smernicam kompleksnih metodologij, je veliko, saj morajo poleg svojega primarnega dela skrbeti tudi za številne izdelke in naloge, ki se vsaj na videz zdijo sekundarnega pomena. Do pred kratkim je veljalo, da je takšen pristop pač potreben, če želimo razvijati sisteme, ki bodo učinkoviti in jih bo moč obvladovati tudi čez več let. Če ne bi bili izpostavljeni spremembam ter konkurenci pri razvoju programske opreme, bi morda tudi danes trdili, da je to edina možnost. Prav zaradi dinamike in konkurence, ki zahtevata hitre rezultate, se danes uveljavlja drugačen pogled na obravnavano problematiko – metodološke smernice se vračajo k ad hoc pristopom.

V prispevku opisujemo temeljna izhodišča nove šole mišljenja, ki jo danes radi povezujemo s pojmom *agilnost*. V slovenskem jeziku temu pravimo gibčnost ali prilagodljivost. Najprej se posvetimo metodologijam na splošno, s čimer želimo poudariti, da je metodologija več kot le skupek metod in tehnik. Sledi poglavje, kjer najprej opišemo ozadje nastanka trenda agilnih metodologij ter temeljna načela in priporočila, ki v zvezi s tem izhajajo iz literature ali temeljijo na neposredni izmenjavi mnenj največjih glasnikov novih pristopov. Naš pogled in razumevanje agilnih metodologij podamo v tretjem poglavju, kjer predstavimo

tudi izkušnje, ki smo jih pridobili z zasnovo in uvajanjem agilne metodologije na enem izmed slovenskih računalniških podjetij.

2 KAJ JE METODOLOGIJA IN ZAKAJ JO POTREBUJEMO?

2.1 Metodologija ima pomembno sociološko komponento

Ko govorimo o metodologijah, pogosto ne vemo dobro, kaj sploh je metodologija. Ni res, da številna podjetja pri svojem delu ne uporabljajo nobene metodologije, saj je ta prisotna pri vsakem organiziranem delu. Metodologija zajema vse, kar redno počnemo, da bi dosegli želen rezultat, torej izdelek ali storitev, ki je cilj našega dela. V primeru razvoja programske opreme to ne pomeni zgolj postopkov, ki so neposredno povezani z razvojem (npr. analiza, načrtovanje ipd.), temveč zajema tudi podporne postopke, načine komunikacije med sodelujočimi, pravila odločanja itn. Metodologijo lahko opredelimo tudi kot množico dogovorov (konvencij), s katerimi se projektna skupina/organizacija strinja (1).

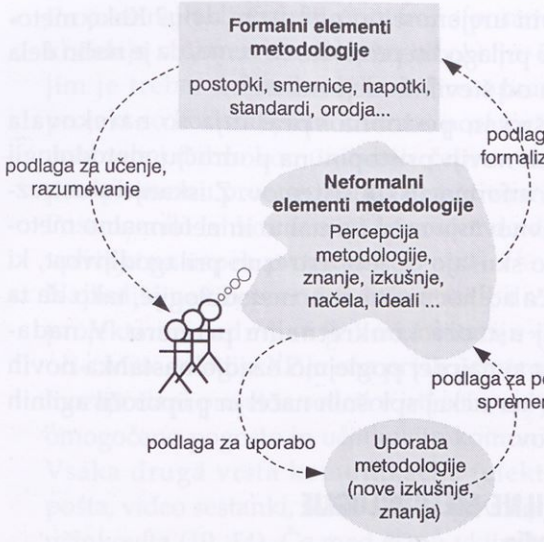
Metodologija je prežeta s filozofijo, načeli, idejami in pogledi organizacije in njenih članov, kar še posebej poudarja njeno sociološko komponento. Metodologija namreč ne nastane neodvisno od ljudi, katerim je namenjena. Čeprav obstajajo številne formalne metodologije, ki temeljijo na preizkušenih postopkih, ki so se v praksi izkazali kot dobri, si jih organizacije, ko jih privzamejo, vedno prilagodijo, tako da ustrezajo njihovemu načinu dela ter percepciji domene, za katero so vzpostavljene. Z uporabo metodologije se uporabniki učijo in pridobivajo nove izkušnje, s čimer se posledično bogati tudi metodologija sama.

2.2 Formalni in neformalni elementi metodologije

Metodologija zajema poleg formalno opredeljenih elementov, ki so navadno zapisani v obliki postopkov, pravil, napotkov, smernic, standardov itn. v navadnih ali elektronskih priročnikih, tudi številne nedokumentirane elemente. Posebno mesto med njimi nosi znanje, ki ga člani organizacije uporabljajo pri svojem delu. To je za organizacijo ključnega pomena, saj predstavlja tiste vrednote, ki so organizaciji lastne in so njena konkurenčno vrednost. Posledično velja to tudi za metodologijo. Temeljni elementi metodologije se skrivajo ravno v znanju in izkušnjah posameznikov.

V znanosti se z upravljanjem z znanjem ukvarja posebna veja (angl. *Knowledge Management*). Njen namen je iskanje tehnik in metod za zajem, formalizacijo in prenos znanja. Kot nas ugotovitve iz omenjenega področja učijo, se znanje deli na *eksplicitno* in *skrito* znanje (12). Eksplicitno znanje (angl. *Explicit Knowledge*) je moč izraziti v formalni obliki, skrito znanje (angl. *Tacit Knowledge*) pa je navadno subjektivne narave in prepleteno z izkušnjami, ideali, čustvi, intuicijo ipd., zaradi česar ga je težko izraziti. Vendar pa se lahko skrito znanje sčasoma rutinizira in tako postane izrazljivo. Omenjeni pojav ima tudi na področju metodologij svoje mesto. Dokumentirana metodologija je sprva podlaga, iz katere se uporabniki učijo in razvijejo svojo percepcijo o načinu dela, komunikaciji, odločanju ipd. Z uporabo pridobivajo nove izkušnje in bogatijo svoje znanje, kar vpliva tudi na metodologijo. Ko postane znanje, ki ga uporabljajo pri delu, dovolj rutinsko, pridobi obliko podatkov in tedaj lahko postane del formalne metodologije (9). Z uporabo se metodologija bogati s skritim znanjem posameznikov. Iz njega se lahko sčasoma izkristalizirajo formalne oblike (metode, postopki, smernice, napotki) ter postanejo del formalne metodologije. Pri tem je treba upoštevati, da je skrito znanje, ki je podlaga za nastanek eksplicitnega znanja, vedno bogatejše od stvaritve same (11, 13).

Slika 1 prikazuje cikel, ki poteka od uporabe formalne metodologije kot podlage za učenje ter



Slika 1: **Od neformalne do formalne metodologije**

razumevanje metodologije do formalizacije elementov neformalne metodologije, ki se izkažejo dovolj rutinski za standardizacijo.

Številna računalniška podjetja še vedno razvijajo programsko opremo na podlagi neformalno opredeljenih metodologij, ki niso dokumentirane. Čeprav so postopki znani in ustaljeni, jih podjetja redko eksplicitno zapišejo, še bolj poredko pa svoje metodologije osvežujejo v skladu s pridobljenimi izkušnjami in znanjem.

Meje med formalnim in neformalnim delom metodologije ni preprosto določiti. Če je formalni del metodologije obsežen, ga težko vzdržujemo, kar ima za posledico zastaranost metodologije in neustreznost dejanskemu stanju. Metodologija je namreč dinamična in se stalno spreminja. Obsežna neformalna metodologija pa je tvegana za organizacijo, saj je popolnoma odvisna od njenih uporabnikov. Potreba po formalizaciji metodologije izhaja že iz dejstva, da je razvoj programske opreme sistematičen proces, ki mora biti ustrezno zasnovan in dokumentiran, da lahko učinkovito usmerja posameznike in skupine k dobrim rezultatom. Če je metodologija opredeljena zgolj na neformalni ravni, je razvoj težko standardizirati, postopek pa postane nepregleden in neobvladljiv. Kako podrobno naj bo torej metodologija dokumentirana? Kateri elementi metodologije so najbolj stabilni in zato smiselni za formalizacijo? Kako zagotoviti, da bo metodologija enostavna in prilagodljiva, po drugi strani pa dovolj rigorozna, da bo od uporabnikov zahtevala sistematiko in urejenost pri njihovem delu? Kako metodologijo prilagoditi projektu, če vemo, da je način dela odvisen od številnih dejavnikov?

Takšna in podobna vprašanja so narekovala nastanek novih pristopov na področju metodologij razvoja informacijskih sistemov. Z iskanjem ustreznega ravnovesja med formalno in neformalno metodologijo skušajo doseči ustrezno prilagodljivost, ki omogoča ad hoc nastavitev metodologije, tako da ta čimbolj ustreza konkretnemu primeru. V nadaljevanju si najprej poglejmo ozadje nastanka novih trendov ter nekaj splošnih načel in priporočil agilnih pristopov.

3 AGILNE METODOLOGIJE

3.1 Ozadje

Temelji agilnih metodologij so bili postavljeni decembra 2001, ko se je sestala skupina sedemnajstih *gurujev*, tako ali drugače znanih s področja *lahkih* metodologij (*Adaptive Software Development, XP-Extreme Programming, Feature-Driven Development, Crystal, Scrum, Dynamic System Development Method* idr.). Namen sestanka je bil ugotoviti, kaj imajo »njihove« metodologije skupnega in na osnovi tega postaviti skupne metodološke osnove. Čeprav trdijo, da njihov namen ni enotna lahka metodologija – ULM *Unified Light Methodology* (spomnimo se, da je pred desetimi leti šlo za podobno situacijo, ko so se zaradi poplave objektno usmerjenih metod pomembni posamezniki združili in dali temelje jeziku UML *Unified Modeling Language*), pa gre za podoben primer. Tokrat so pod drobnogledom lahke metodologije, katerih število je v zadnjih letih močno naraslo. Njihova temeljna lastnost je učinkovitost in prilagodljivost, s čimer je skupina sedemnajstih, združena v zvezo *Agile Alliance (1)*, opisala tudi nov trend agilnih metodologij.

3.2 Temeljna načela agilnih metodologij

V skupni izjavi so člani zveze kot svoj cilj zapisali iskanje boljših pristopov razvoja programske opreme, tako da pri tem sami sodelujejo in pomagajo drugim. Iz tega so izpeljali štiri načela:

- Posamezniki in njihova komunikacija so pomembnejši kot sam proces in orodja,
- Delujoča programska oprema je pomembnejša kot popolna dokumentacija,
- Vključevanje (sodelovanje) uporabnika je pomembnejše kot pogajanje na podlagi pogodb
- Upoštevanje sprememb je pomembnejše od sledenja planu.

Proces, orodja, dokumentacija, pogodbe in plani so vsekakor pomembni elementi metodologije, vendar pa je v primeru, ko je treba stvari postaviti na tehtnico, pomembneje poskrbeti za posameznike, sodelovanje, delujočo programsko opremo, vključevanje uporabnika in reagiranje na spremembe.

Poglejmo vsako izmed načel podrobneje.

3.2.1 Posamezniki in komunikacija vs. proces in orodja

Več pozornosti je treba nameniti članom projekta, njihovemu znanju in tudi željam. Toga porazdelitev vlog, ki jih določa proces, med člane projekta, ni smiselna, če ni ustreznih ljudi. Načelo poudarja tudi pomen komunikacije, ki je ključna za uspešnost projekta. Boljše rezultate dosežemo, če sta komunikacija in sodelovanje med člani projekta dobra,

čeprav se ne držijo predpisanega procesa, kot pa v primeru, ko je proces predpisan, komunikacija med člani pa slaba.

3.2.2 Delujoča programska oprema vs. popolna dokumentacija

Delujoč program je največ, kar lahko dobi končni uporabnik. Dokumentirane zahteve, model statičnih elementov sistema, model interakcij in druga dokumentacija o problemski domeni in sistemu so koristni, vendar drugotnega pomena. Naročnik ne bo zadovoljen s kupom dokumentacije, če ne bo najprej videl delujočega sistema. Ena večjih kritik *slapovnega* procesa je, da naročnik do konca projekta čaka na sleherno komponento programske opreme. Vse, kar lahko medtem dobi, je dokumentacija. Pod drugi strani pa je dokumentacija temeljnega pomena, saj olajša vzdrževanje sistema, komunikacijo med izvajalcem in naročnikom, preučevanje kakovosti dela projektne skupine itn. Zato je pomembno, da pripravljamo ustrezno dokumentacijo. Vsak del dokumentacije mora biti utemeljen.

3.2.3 Sodelovanje uporabnika vs. pogajanje na podlagi pogodb

S tretjim načelom je pozornost posvečena razmerju med naročnikom in izvajalcem, ki je pogosto preveč formalen in strog. Za uspešnost projekta je ključnega pomena, da se naročnik in izvajalec dobro ujameta. Naročnik najbolje ve, kaj potrebuje, čeprav tega pogosto ne zna razložiti. V agilno usmerjenih metodologijah je zato vloga naročnika postavljena na pomembno mesto. Medtem ko lahko dober odnos med naročnikom in izvajalcem olajša tehnološko še tako zahteven projekt, lahko strog pogodbeni odnos in nerazumevanje med naročnikom in izvajalcem zamaje tudi preproste projekte.

3.2.4 Reagiranje na spremembe vs. sledenje planu

Spremembe so eden od razlogov, ki ga pogosto povezujemo z neuspešnimi projekti. Številni avtorji ugotavljajo pomen obvladovanja sprememb in temu sledijo tudi moderne metodologije. Dinamika poslovnih okolij botruje številnim spremembam, zato je naivno pričakovati, da bomo lahko v začetnih fazah projekta zajeli vse zahteve. Projektni plani so koristen element, vendar morajo omogočati spremembe, vsaj v smislu preureditve prioritet znotraj dogovorjenega okvirja. Planiranje in sledenje planu je koristno, vendar le dokler se to ne razlikuje preveč od dejanskega stanja. Najslabše je slediti planu, ki je zastarel.

3.3 Priporočila

Iz temeljnih načel agilnih metodologij so izpeljana številna priporočila v pomoč pri gradnji in vrednotenju metodologij. Vse metodologije, ki se upravičeno deklarirajo kot agilne, jih upoštevajo. Naj jih naštejemo le nekaj:

- Najvišja prioriteta je zadovoljstvo stranke, zato je priporočeno zgodnje in pogosto izdajanje komponent preizkušene in delujoče programske opreme. To je temelj prilagodljivega razvoja, saj na podlagi medprojektnih izdaj pridobimo koristne povratne informacije, tveganje, da bo izdelek neustrezen, pa zmanjšamo.
- Delujoče komponente programske opreme je treba izdajati pogosto, v ciklu, ki ni daljši od štirih mesecev. Če je naročnik zmožen sprejemati izdelke v krajših razmakih in je tudi razvojna ekipa sposobna upoštevati vse sprotne spremembe, so krajši cikli še boljši. Iterativni pristop je nujen.
- Spremembe v zahtevah naj bodo dobrodošle tudi v poznih fazah razvoja. Eno izmed načel agilnih pristopov je dober odnos med naročnikom in izvajalcem. Spremembe, ki nastanejo med projektom, in ki smo jih pripravljeni upoštevati, lahko prinesejo naročniku pomembno konkurenčno prednost, zato se jih ne smemo otepati. Iterativni pristop je tudi tu primeren.
- Projekti naj vključujejo motivirane posameznike, ki delajo v ustreznem delovnem okolju. Zaupati jim je treba, da bodo delo dobro opravili. Za projekt so koristnejši motivirani posamezniki, ki med seboj dobro komunicirajo, čeprav ne sledijo predpisanemu procesu, kot pa nemotivirani ljudje. Posamezniki lahko s svojim odnosom do dela zelo vplivajo na uspeh projekta.
- Najboljši način prenosa informacij do članov projekta in med njimi je komunikacija »iz oči v oči«. Metodologija XP je ena prvih priporočala, da morajo člani projekta sedeti v isti sobi, saj je s tem omogočeno pogosto in učinkovito komuniciranje. Vsaka druga vrsta komunikacije (elektronska pošta, video sestanki, telefonski pogovori) je manj učinkovita (10, 14). Če med člane vključimo tudi predstavnike naročnika, je za odgovor potrebno malo časa.

- Kakovost programske kode in arhitekture je potrebno stalno preverjati in izboljševati. S tem povečamo prilagodljivost. Pristop, kjer najprej izpeljemo poglobljeno analizo, izdelamo dober načrt in nato razvijemo programsko kodo, je priporočljiv, vendar si ga pogosto ne moremo privoščiti. Zaradi potrebe po hitrih rezultatih je kakovost zapostavljena, zato je pomembno, da si v vsakem ciklu prizadevamo za izboljšave.
- Enostavnost procesa je ključ do uspeha. Kako razviti kakovostno programsko opremo, pri tem pa opraviti čim manj stranskih nalog in izdelkov? Proces mora biti ravno dovolj zahteven. Raje malo manj kot pa preveč. To ne pomeni, da moramo aktivnosti izvajati površno ali pomanjkljivo. Nasprotno, delo mora biti opravljeno kvalitetno, paziti pa moramo, da ni po nepotrebnem kompliciranо. Pot do enostavnosti je težka, saj je včasih preprosteje pustiti stvari kompleksne, kot jih pa poenostaviti.
- Po vsakem intervalu (iteraciji) moramo preveriti proces, ki smo ga uporabljali. Glede na ugotovitve proces prilagodimo in izboljšamo za prihodnje iteracije.

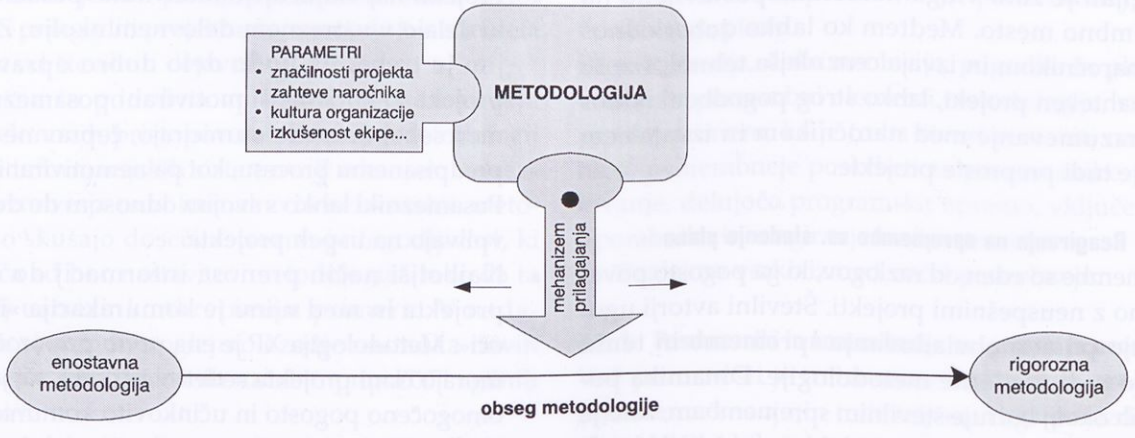
4 IZKUŠNJE PRI ZASNOVI IN UVEDBI AGILNE METODOLOGIJE

4.1 Razumevanje koncepta agilne metodologije

Načela in priporočila, opisana v prejšnjem razdelku, zajemajo le nekaj skupnih smernic, ki so se oblikovale v krogu zagovornikov agilnih pristopov. V precej širšem kontekstu obravnavata nove trende Cockburn in Highsmith v svojih knjigah (5, 7), kjer se ukvarjata z metodologijo razvoja kot celoto in podajata številna zanimiva razmišljanja, ki potrjujejo potrebo po prilagodljivem načinu razvoja programske opreme. Za razliko od njiju se Ambler in nekateri drugi avtorji v svojih delih posvečajo predvsem modeliranju kot ključnemu elementu za dosego učinkovitega razvoja programske opreme (npr. 2, 3, 4, 6). Po njihovem mnenju je modeliranje kritični element metodologije razvoja programske opreme, ki zahteva največjo prilagodljivost.

Iz raznih virov prihajajo tudi številne druge ideje, ki včasih odražajo subjektivna mnenja. To seveda povzroča določene nejasnosti, kar se kaže tudi v številu metodologij, ki jih danes avtorji deklarirajo kot agilne. Še posebej velja poudariti, da agilne metodologije niso nujno lahke metodologije in obratno, niso vse lahke metodologije tudi agilne. Prilagodljivost je pravzaprav popolnoma neodvisna od teže metodologije[1]. Tako lahka kot težka metodologija je lahko agilna, če omogoča ad hoc prilagajanje obsega metodologije dejanskim potrebam projekta.

Na spodnji sliki je koncept agilne metodologije predstavljen grafično. Agilna metodologija se s pomočjo mehanizma za prilagajanje v odvisnosti od parametrov, kot so značilnosti projekta, zahteve naročnika, kultura organizacije, izkušenost ekipe ipd., pozicionira na ustrezno mesto na skali, ki opredeljuje obseg in s tem tudi težo metodologije.



Slika 2: **Koncept agilne metodologije**

[1] *V (1) je teža metodologije opredeljena kot zmnožek obsega in gostote metodologije. Obseg metodologije pove, koliko različnih elementov, tj. aktivnosti, vlog, izdelkov, priporočil itd. zajema metodologija. Gostota pa se nanaša na raven podrobnosti, s katero so elementi opisani. Metodologija, kot je RUP, velja za težko metodologijo, saj zajema in podrobno opisuje številne elemente. Med tipične predstavnike lahkih metodologij sodijo Extreme Programming, Feature-Driven Development, Crystal Clear ipd.*

V nadaljevanju podajamo izkušnje, ki smo jih pridobili z izdelavo in uvajanjem agilne metodologije razvoja programske opreme v enem od slovenskih računalniških podjetij.

4.2 Prilagoditev referenčnega procesa

Podjetje, s katerim smo sodelovali, se je za formalizacijo procesa razvoja odločilo predvsem zaradi možnosti uvajanja novih članov, zaradi lažjega pridobivanja naročnikov ter zaradi potrebe po standardizaciji dokumentacije. Metodologija, ki se je uporabljala v podjetju, je bila precej stihijska in prepuščena posameznim izvajalcem, nova metodologija pa naj bi temeljila na dokumentiranem procesu, ki bi bil enostaven, učinkovit in prilagodljiv potrebam posameznih projektov.

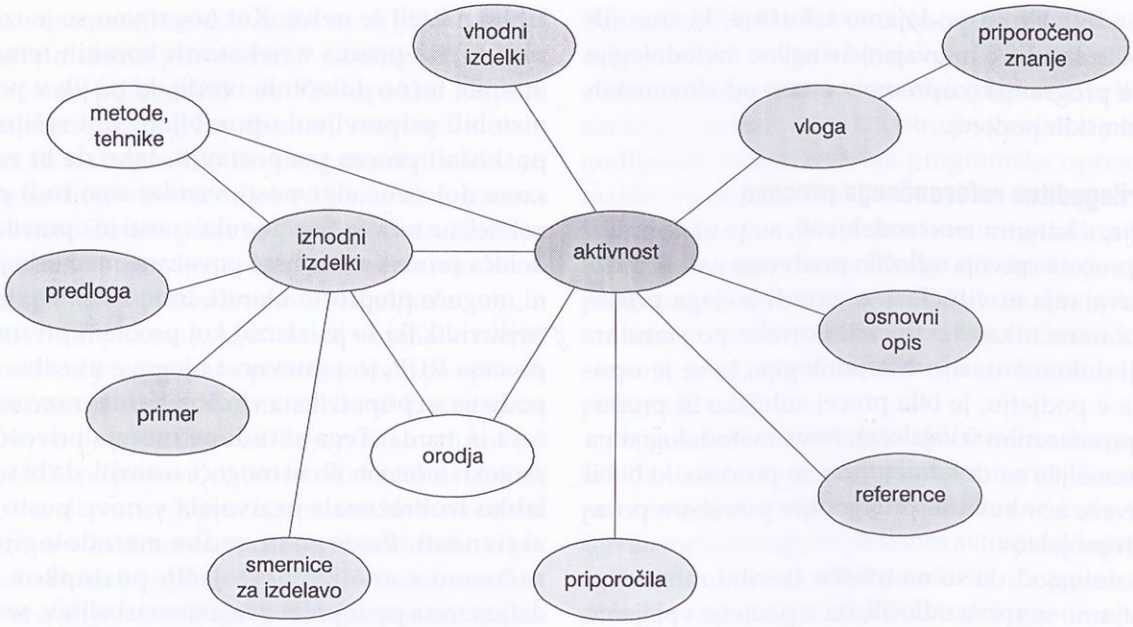
Upoštevajoč, da so na tržišču številni referenčni procesi, smo se sprva odločili, da v podjetje vpeljemo enega izmed preizkušenih modelov, nato pa ga prilagodimo, tako da bo ustrezal zahtevam podjetja. Pri izbiri ustreznih referenčnih procesov smo si pomagali z modelom, ki ga razvijamo v laboratoriju za informatiko v sklopu raziskovalne naloge (15, 16). Model temelji na predpostavkah o korelacijah med karakteristikami projektov in karakteristikami metodologij. Tako smo upoštevali karakteristike tipičnih projektov, ki jih podjetje izvaja, preferenčne želje podjetja (objektno usmerjen razvoj, iterativni življenjski cikel, lahka metodologija idr.) ter znanja in izkušnje posameznikov s področja razvoja programske opreme. Po izvedeni analizi rezultatov smo se odločili za vpeljavo procesa RUP (Rational Unified Process (8)), ki se je tudi v okviru uporabe modela za izbiro metodologije izkazal kot eden primernejših procesov s področja objektnega razvoja.

V prvem koraku smo izvedli izobraževanje. Izpeljali smo več tečajev, s katerimi smo uporabnikom predstavili podrobnosti procesa, obenem pa jih skušali naučiti, kako lahko proces uporabljajo pri svojem delu. Čeprav so uporabniki proces sprejeli kot dober, ga niso bili pripravljeni uporabljati v praksi, saj se jim je zdel preveč zahteven, pa tudi nepotreben. Eden takih korakov je na primer izdelava analize in načrta arhitekture, ki je po RUP precej zahtevna naloga. Arhitekturno ogrodje, ki so ga v podjetju razvili na osnovi večletnih izkušenj, namreč že samo po sebi določa postopek opredelitve arhitekture, kar izključuje številne podrobnosti, ki jih RUP predpisuje v sklopu aktivnosti za arhitekte. In takih korakov bi lahko našteli še nekaj. Kot negativno se je izkazalo tudi to, da proces v nekaterih korakih temelji na uporabi točno določenih orodij, ki pa jih v podjetju niso bili pripravljeni uporabljati. Kot rešitev smo poskušali proces poenostaviti, tako da bi zajemal samo določene aktivnosti, vendar smo tudi pri tem naleteli na težave. Številne aktivnosti in opravila, ki jih določa proces, so namreč povezane med seboj, in jih ni mogoče preprosto ukiniti. In morda najpomembnejši vidik, ki se je izkazal kot problem pri uvajanju procesa RUP, je zahtevnost njegove uvedbe. Redka podjetja so pripravljena vložiti v proces razvoja toliko časa in truda. Tega si tudi ne morejo privoščiti, saj projekti tečejo in jih ni mogoče ustaviti, da bi se ekipa lahko izobraževala in uvajala v nove postopke in aktivnosti. Postopna uvedba metodologije, kjer začnemo z analizo obstoječih postopkov in nadaljujemo s postopnim uvajanjem izboljšav, se nam je zdela primernejša.

4.3 Zajem osnovnega procesa z analizo obstoječe metodologije

Iz pogovorov smo kasneje ugotovili, da so zaposleni zadovoljni z mnogimi aktivnostmi, ki so jih izvajali pri razvoju programske opreme in da ne čutijo potrebe, da bi jih morali ukini ali spremeniti. Zato se je zdelo smiselno, da najprej preučimo obstoječo metodologijo in določimo njene osnovne elemente. Od uporabnikov smo skušali izvedeti, kaj radi počnejo in kaj se jim zdi odveč, katere dele dokumentacije izdelajo in zakaj, v katerih korakih imajo največ težav s komunikacijo, kje čutijo pomanjkljivosti v procesu, kaj bi spremenili, kaj obdržali itn. Pregledali smo dokumentacijo, ki nastaja pri projektih, preverili orodja ipd. Na podlagi ugotovitev smo nato dokumentirali osnovne korake metodologije. Pri opisu smo se držali priporočil agilnih pristopov in med osnovne elemente metodologije zajeli le tiste, ki so se zdeli dovolj stabilni in zato smiselni za formalizacijo. Izbrane elemente prikazuje slika 3.

Metodologijo smo predstavili grafično ter opisali njene gradnike. Ker je namen metodologije usmerjati sodelujoče na projektu, ki nastopajo v določenih vlogah, smo opise organizirali po vlogah. Osnovne vloge smo določili na podlagi lastnosti tipičnih projektov, ki jih v podjetju izvajajo, in upoštevali znanje in izkušnje zaposlenih. Za vsako vlogo smo zapisali priporočeno znanje. Aktivnosti vlog smo le kratko opisali, saj je treba upoštevati, da se dejanska metodologija pogosto



Slika 3: **Osnovni elementi opisa metodologije**

spreminja, kar otežuje vzdrževanje dolgih in podrobnih opisov. Za potrebe uvajanja novih članov smo dodali reference na podrobnejše opise (knjige, spletni naslovi).

Pomembna opisna elementa metodologije sta vhod in izhod aktivnosti. Vsak izdelek, ki nastane kot stranski ali končni izdelek aktivnosti, smo kratko opisali ter podali primer. Če je šlo za dokument, smo zaradi standardizacije dokumentacije pripravili tudi predlogo zanj, vendar smo pri tem upoštevali, da se uporabnik veliko več nauči iz primera kot iz predloge. Podrobnim opisom metod in tehnik ter orodij, ki naj se uporabljajo za izvedbo aktivnosti oz. za izdelavo izdelka, smo se izogibali, razen v primerih, ko je bilo to potrebno zaradi odvisnosti z drugimi aktivnostmi. Samo poznavanje tehnik in orodij namreč še ne zagotavlja, da bomo znali učinkovito izvesti aktivnost. Poleg tega v podjetju orodja in tehnike pogosto menjajo, odnos posameznikov do uporabe predpisanih tehnik in tudi orodij pa je negativen. Veliko večji pomen smo pripisali izkušnjam, ki so jih imeli posamezniki z izvajanjem določenih aktivnosti. Uporabnike smo izzvali, da smernice za izdelavo posameznih izdelkov ter priporočila za izvedbo aktivnosti zapišejo sami. S tem smo dosegli dvoje: (1) uporabniki se niso čutili ogrožene – uporabniki, še posebej tisti, ki so že izkušeni in prepričani, da svoje delo opravljajo dobro, ne vidijo radi, da se jih uči, (2) zajeli smo pomembno znanje, ki bi sicer ostalo le pri posameznikih.

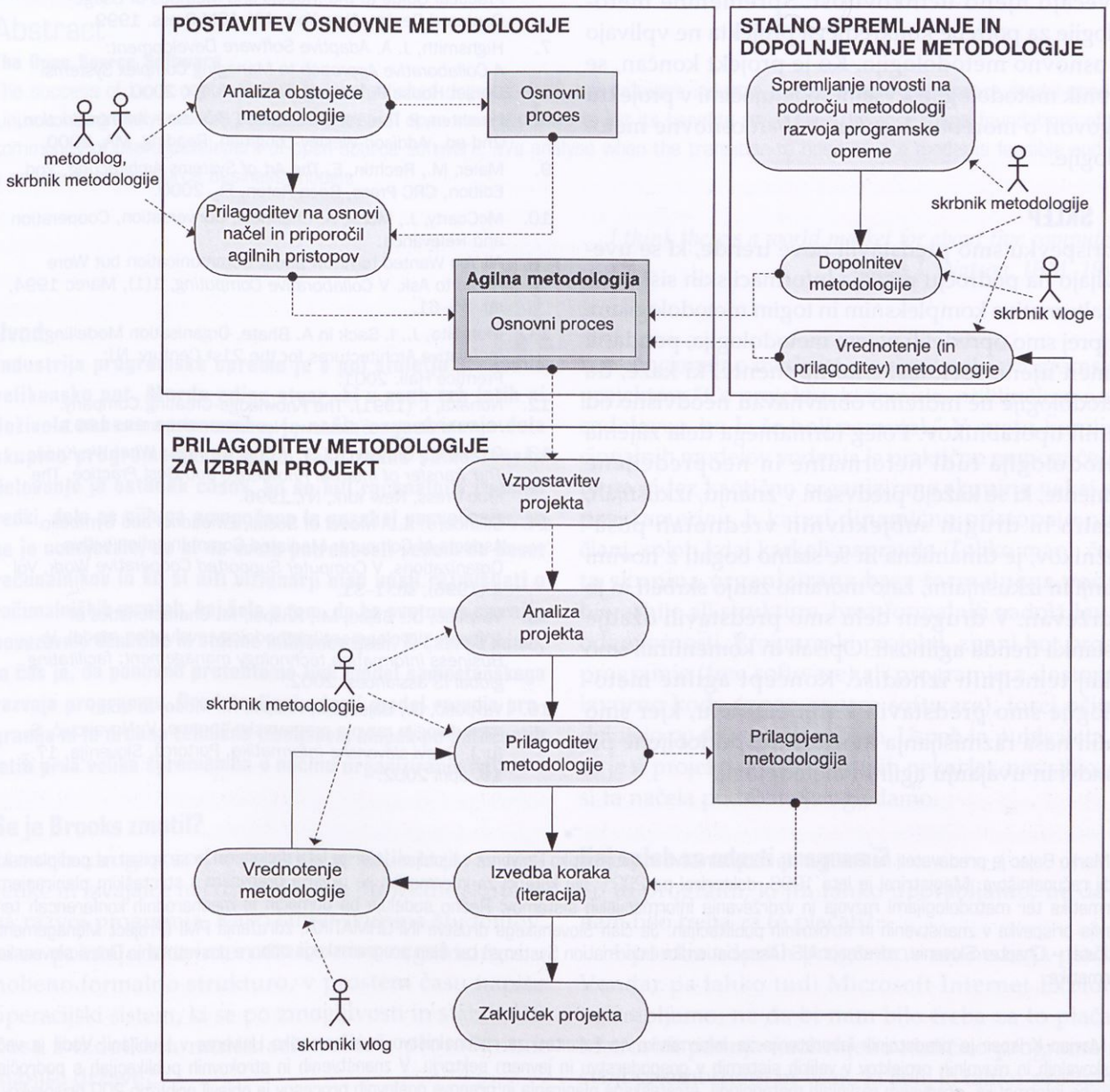
Grafično predstavitev ter opise osnovnih elementov metodologije smo zajeli s preprosto spletno aplikacijo, ki uporabnikom omogoča enostaven dostop (potrebujejo zgolj internetni brskalnik) ter dodajanje smernic in priporočil na podlagi izkušenj. Menimo, da je takšna predstavitev metodologije nujna, če želimo povečati njeno uporabnost. Večinoma gre namreč za kratke vpoglede in ne za učenje metodologije, za kar je priročnik v knjižni obliki boljši. To je še dodaten razlog, ki podpira uporabo referenc namesto dolgih opisov.

4.4 Uporaba, vrednotenje in prilagajanje metodologije

Že pri analizi in opredelitvi osnovnih korakov metodologije smo upoštevali temeljna načela in priporočila agilnih pristopov, s katerimi smo zagotovili preprostost in učinkovitost metodologije. Seveda je ključnega pomena za uporabnost metodologije njena zmožnost, da se prilagaja posameznim projektom. Vsak projekt je nekaj posebnega, zato je težko pričakovati, da bo zapisani proces (formalni del metodologije) dober za vse primere. Agilna metodologija mora biti prilagodljiva. Poleg tega je treba poskrbeti za njeno stalno obnavljanje, kar zajema in formalizira uporabne postopke in druge elemente. V

našem primeru smo predlagali določitev ustreznih vlog ter opredelitev procesa uporabe, vrednotenja in prilagajanja metodologije. Predlagani proces prikazuje slika 4.

Najprej je treba preučiti obstoječo metodologijo ter formalizirati njene osnovne elemente. Pri tem sodelujeta metodolog, ki je navadno zunanji strokovnjak, ter skrbnik metodologije. Z upoštevanjem načel in priporočil agilnih pristopov predlagata prvi osnutek agilne metodologije, ki jo morajo potrditi tudi drugi skrbniki vlog. Ti so posamezniki z največ izkušnjami za posamezno področje. V sodelovanju z drugimi uporabniki, ki nastopajo kot nosilci vloge, dopolnjujejo metodologijo s priporočili in smernicami, pridobljenih ob delu na posameznih projektih. Dopolnjevanje metodologije je dolžnost skrbnika metodologije, ki spremlja novosti na področju metodologij razvoja programske opreme.



Slika 4: **Proces uporabe, vrednotenja in prilagajanja metodologije**

Pri uporabi metodologije na konkretnem projektu skrbnik metodologije najprej preuči lastnosti projekta in določi osnovne korake metodologije. Sem sodi določanje opcijskih aktivnosti, določanje potrebnih vlog, izbira izdelkov, izbira življenjskega cikla (npr. iteracije znotraj faze konstrukcije) itn. Tako prilagojena metodologija je podlaga za izvedbo iteracije. Po zaključku iteracije se skrbnik metodologije in skrbniki vlog pogovorijo o uporabnosti metodologije. Navadno ne gre za korenite spremembe metodologije, temveč zgolj za podrobnosti, ki lahko še povečajo njeno učinkovitost. Spremembe metodologije za potrebe konkretnega projekta ne vplivajo na osnovno metodologijo. Ko je projekt končan, se skrbnik metodologije z vsemi sodelujočimi v projektu pogovori o morebitnih spremembah osnovne metodologije.

5 SKLEP

V prispevku smo predstavili nove trende, ki se uveljavljajo na področju razvoja informacijskih sistemov kot alternativa kompleksnim in togim metodologijam. Najprej smo opredelili pojem metodologija, poudarili pomen njene sociološke komponente, ki kaže, da metodologije ne moremo obravnavati neodvisno od njenih uporabnikov. Poleg formalnega dela zajema metodologija tudi neformalne in neopredeljene elemente, ki se kažejo predvsem v znanju, izkušnjah, idealih in drugih subjektivnih vrednotah posameznikov, je dinamična in se stalno bogati z novimi znanji in izkušnjami, zato moramo zanjo skrbeti in jo vzdrževati. V drugem delu smo predstavili ozadje nastanka trenda agilnosti. Opisali in komentirali smo nekaj temeljnih izhodišč. Koncept agilne metodologije smo predstavili v tretjem delu, kjer smo strnili naša razmišljanja in izkušnje, pridobljene pri zasnovi in uvajanju agilnih metodologij.

6 LITERATURA

1. Agile Alliance. *Manifesto for Agile Software Development:* www.agilealliance.org
2. Ambler W. S. *Agile Modeling: Effective Practices for extreme Programming and the Unified Process*. New York, John Wiley & Sons, Inc. 2002.
3. Ambler W. S. *The Object Primer, Second Edition: The Application Developer's Guide to Object Orientation*. New York, NY: Camridge University Press. 2001.
4. Ambler W. S. *Enterprise Unified Process White Paper*. www.ronin-intl.com/publications/unifiedProcess.htm
5. Cockburn, A. *Agile Software Development*. Pearson Education, Inc. Boston, MA, 2002.
6. Constantine, L. L. in Lockwood, L. A. D. *Software For Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. New York, NY: ACM Press. 1999.
7. Highsmith, J. A. *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. Dorset House Publishing, New York, NY, 2000.
8. Krushten, P. *The Rational Unified Process – An Introduction*, 2nd ed., Addison-Wesley-Longman, Reading, MA, 2000.
9. Maier, M., Rechtin, E. *The Art of Systems Architecting*, 2nd Edition, CRC Press, Boca Raton, FL, 2000.
10. McCcarty, J., Monk, A. Channels, Conversation, Cooperation and Relevance: All You Wanted to Know about Communication but Were Afraid to Ask. V *Collaborative Computing*, 1(1), Marec 1994, str. 35–61.
11. Morabito, J., I. Sack in A. Bhate. Organisation Modelling, Innovative Architectures for the 21st Century. NJ: Prentice Hall. 2001.
12. Nonaka, I. (1991). The Knowledge-Creating Company. *Harvard Business Review*, November–december 1991.
13. O'Dell, C., Grayson, C. Jr. *If Only We Knew What We Know: The Transfer of Internal Knowledge and Best Practice*, The Free Press, New York, NY, 1996.
14. Sillince, J. A. A Model of Social, Emotional and Symbolic Aspects of Computer-Mediated Communication within Organizations. V *Computer Supported Cooperative Work*, Vol 4 (1996), str. 1–31.
15. Vavpotič, D., Bajec, M., Krisper, M. Characteristics of software development methodology evaluation model. V: *Business information technology management: facilitating global IS assiances*. 2002.
16. Vavpotič, D., Bajec, M., Krisper, M. Model za izbiro metodologije razvoja programske opreme. V: Novaković, S. (ur.). *Dnevi slovenske informatike*, Portorož, Slovenija, 17.–19. april 2002.

Dr. Marko Bajec je predavatelj na Fakulteti za računalništvo in informatiko Univerze v Ljubljani, kjer je leta diplomiral in se vpisal na podiplomski študij računalništva. Magistriral je leta 1998, doktoriral pa 2001. Na katedri za informatiko se ukvarja predvsem s strateškim planiranjem informatike ter metodologijami razvoja in vzdrževanja informacijskih sistemov. Redno sodeluje na domačih in mednarodnih konferencah ter objavlja prispevke v znanstvenih in strokovnih publikacijah. Je član Slovenskega društva INFORMATIKA, združenja PMI (Project Management Institute) – Chapter Slovenia, združenja AIS (Association for Information Systems) ter član programskega odbora posvetovanja Dnevi slovenske informatike.

Dr. Marjan Krisper je predstojnik laboratorija za informatiko na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Vodil je več raziskovalnih in razvojnih projektov v velikih sistemih v gospodarstvu in javnem sektorju. V znanstvenih in strokovnih publikacijah s področja poslovne informatike, predvsem razvojnih metodologij, strateškega planiranja in prenove poslovnih procesov je objavil približno 200 prispevkov. Je ustanovitveni član AIS (Association for Information Systems), član Slovenskega društva INFORMATIKA, PMI (Project Management Institute), Društva za umetno inteligenco, INFOSA idr.

Odprto programje

Primož Peterlin
Univerza v Ljubljani, Medicinska fakulteta, Inštitut za biofiziko, Lipičeva 2, 1000 Ljubljana
primoz.peterlin@biofiz.mf.uni-lj.si

Povzetek

Uspeh Linuxa je postavil na glavo nekaj dogem o industriji programske opreme. Analiziramo model odprtega programja, ki ga uporablja Linux. Predstavimo, kaj je odprto programje, katere so njegove prednosti in orišemo sociološke temelje, ki vladajo v skupnosti uporabnikov razvijalcev odprtih programov. Raziščemo, kdaj je prehod na odprt model razvoja smiseln in navedemo nekaj ekonomskih modelov poslovanja v svetu odprtega programja.

Abstract

The Open Source Software

The success of Linux contradicts a few established dogmas about the software industry. The open source software model used by Linux is being analysed. We present the idea of open source software, list its benefits and outline the sociological foundations of the community of users-developers of open source software. We analyse when the transition to open source model is feasible and list a couple of open source business models.

I think there's a world market for about five computers.
(Thomas J. Watson, IBM, 1943)

Uvod

Industrija programske opreme je v pol stoletja prehodila velikansko pot. Morda edina stvar, ki v vseh teh letih ni doživela nobene spremembe, je način organiziranja dela skupine programerjev. Zaprto, tako rekoč samostansko delovanje je ostanek časov, ko so bili računalniki izjemno redki, delo na njih pa omogočeno le posebej posvečenim, ko se je ocenjevalo, da bi na svetu potrebovali sedem do deset računalnikov in ko si niti vizionarji niso upali razmišljati o računalniških mrežah, kaj šele o tem, da bo svetovno omrežje povezovalo desetine in stotine milijonov ljudi. Ti časi so mimo, in čas je, da ponovno pretehtamo tudi model samostanskega razvoja programov. Predstavljeni odprti model razvoja programja ni le drobna tehnična izboljšava, ampak po petdesetih letih prva velika sprememba v načinu organiziranja dela.

Se je Brooks zmotil?

Linux in drugi sorodni projekti so velik izziv za teoretike in izvajalce načrtovanja dela projektne skupine za razvoj programja. Kako lahko skupina študentov, ki se poznajo le z interneta in niso organizirani v nobeno formalno strukturo, v prostem času napiše operacijski sistem, ki se po zmogljivosti in stabilnosti kosa z izdelkom multinacionalke s proračunom manjše države?

Zakaj izziv? Fred Brooks v svoji knjigi The Mythical Man-Month (Brooks 1975) sočno povzema konvencionalno modrost o načrtovanju programskih projektov: "Če projektu, ki zamuja, priključimo nove sodelavce, bo le še bolj zamujal." V svetu konvencionalnih modelov vodenja je praktično nemogoče, da bi na videz kaotično organizirana skupina nekaj sto programerjev, h kateri dinamično pristopajo novi člani, sploh kdaj karkoli napravila. Toliko manj, če je ta skupina organizirana brez formalnega vodje, hierarhije ali strukture, brez formalnih zadolžitev in odgovornosti. Programski projekti, znani kot prosto programje (free software) ali programje z dostopno izvorno kodo (open-source software), torej očitno delujejo po drugačnih načelih. Uspeh in publiciteta, ki so je ti projekti deležni zadnjih nekaj let, nas silijo, da si ta načela podrobneje ogledamo.

Kaj sploh so odprti programi?

So to programi, ki jih lahko uporabljamo, ne da bi nam bilo treba za to plačati?

Uporaba odprtih programov res ne terja plačila. Vendar pa lahko tudi Microsoft Internet Explorer uporabljamo, ne da bi nam bilo treba za to plačati. Microsoft ponuja neokrnjeno različico programa na svoji spletni strani in vsak, ki želi, jo lahko presname na svoj disk in uporablja. Vendar Internet Explorer kljub temu še ni odprt program. Dobimo namreč

samo prevedeno kodo, izvorne pa ne. Zato smo ne glede na to, ali znamo ali ne, prikrajšani za to, da bi odpravili napako, dodali kaj novega k programu ali ga priredili tako, da bi deloval na drugem operacijskem sistemu, in za to, da bi izboljšano izvedbo programa delili z ostalimi. Še več, pravzaprav niti prevedene kode ne smemo deliti, saj nam kot uporabniku pogodba dovoljuje le, da ga shranimo na disk in uporabljamo.

Prostost ni stvar cene. Nič nenavadnega se nam ne sme zdeti, da moramo za odprte programe plačati. Del denarja bo pri tem ostal v trgovini, del ga založnik porabi za kritje svojih stroškov, del pa ga, če da kaj nase, nameni tudi avtorjem programja na CD-ju.

Dobimo z odprtimi programi tudi izvorno kodo?

Razpoložljivost izvorne kode je res nujen pogoj, da je program odprt, sama po sebi pa še ne pomeni odprtosti programov. Znan tak primer, ki ga verjetno poznajo mnogi študenti naravoslovno-tehničnih fakultet, je zbirka podprogramov iz učbenika Numerical Recipes. Res je dostopna z izvorno kodo (in njeno razlago), a tudi z nadvse restriktivnim dovoljenjem za uporabo. To nam sicer dovoljuje vgradnjo podprogramov iz zbirke v svoje programe in razširjanje le-teh, a le, če ne objavimo izvorne kode programa. Kljub temu, da so dostopni z izvorno kodo vred in da za njihovo uporabo ni treba posebej plačevati, ti podprogrami niso prosti, saj je njihova uporaba dovoljena le tistim, ki so kupili učbenik.

Kaj so torej odprti programi?

Na kratko: odprti programi so tisti programi, ki uporabniku dovoljujejo uporabo, razmnoževanje, razširjanje, razumevanje delovanja, spreminjanje in izboljševanje programa. Natančneje pa lahko kot odprto programje opredelimo vse programe, ki izpolnjujejo naslednje pogoje:

- *Dovoljenje za uporabo brez dodatnih omejitev.* Dovoljenje za uporabo mora zadostovati za uporabo programa, ne da bi moral uporabnik pristajati na dodatne omejujoče pogoje. Če je program del paketa, dovoljenje za uporabo ne sme omejevati pravic do uporabe programa zunaj tega paketa.
- *Prosto razširjanje.* Dovoljenje za uporabo programa ne sme preprečevati nadaljnjega razširjanja programa. Uporabnik sme program deliti brezplačno ali ga prodajati, ga vključevati v zbirke ali distribucije programov in za to ni dolžan plačevati odškodnine avtorju programa.
- *Dostopnost izvorne kode.* Program mora biti dostopen v izvorni kodi, sme se ga razširjati tako v izvorni kot v prevedeni obliki. Če se razširja le v prevedeni obliki, mora biti jasno označena možnost brezplačnega dostopa do izvorne kode. Izvorne kode tudi ni dovoljeno pisati namenoma nepregledno.
- *Izvedena dela in integriteta avtorjeve kode.* Program se sme spreminjati, izvedena dela pa se smejo razširjati pod enakimi pogoji kot izvirnik. Avtor lahko zahteva, da se spremembe in dopolnitve razširjajo kot popravki izvorne kode ločeno od izvirnika in jih uporabnik vključi pred prevajanjem programa.
- *Enakopravnost uporabnikov in načinov uporabe.* Dovoljenje za uporabo programa ne sme razlikovati uporabnikov ali skupin uporabnikov niti ne sme omejevati načinov uporabe programa.

Zakaj tako? Dovoljenje za uporabo odprtega programja ne varuje avtorja, temveč proces nastajanja odprtega programja. Zagotavlja, da bo programje dostopno za neodvisni zunanji pregled in nadaljnji razvoj. Moč odprtega programja je v kreativnem potencialu najširše baze uporabnikov razvijalcev, ki je s prvotnim avtorjem izenačena glede dostopa do izvorne kode programa, in spodbujana k spremembam in dopolnitvam programa.

Kaj pomeni Open Source, Freeware, GPL, Copyleft, Public domain, Shareware?

Najbolj preprosto avtor spravi svoj program v obtok tako, da ga izroči v javno last (angl. public domain). Z zahtevo, da se ga navaja kot avtorja programa, obdrži moralne pravice, lahko pa se navajanju tudi odpove. To omogoča, da program kdorkoli uporablja in razširja s svojimi izboljšavami vred. Program v javni lasti nima vgrajene varovalke pred tem, da si program kdo prilasti – ga spremeni ali dopolni in ga razširja naprej pod drugačnimi, bolj omejujočimi pogoji. Kdor prejme program v takšni spremenjeni obliki, z njim ne prejme tudi svoboščin, ki mu jih je namenil prvotni avtor, te mu je odvzel posrednik.

Ob zagonu projekta GNU sredi osemdesetih let je zato fundacija za prosto programje (Free Software Foundation; FSF) poskusila nadomestiti pomanjkljivo pravno varstvo programja v javni lasti z bolj zavezujočo pogodbo. Temeljno vodilo je bila zagotovitev enakih pogojev za vse uporabnike. Zasnovo so poimenovali copyleft (besedna igra z izrazom copyright); po njej mora vsak, ki razširja programje v spremenjeni ali

nespremenjeni obliki, z njim razširjati tudi pravice in svoboščine, ki jih je prejel, vključno s pravico do nadaljnjega razširjanja in spreminjanja. FSF je za programje, ki se razširja pod takimi pogoji, skovala izraz prosto programje (angl. free software). Slovenski izraz je na srečo jasnejši od angleškega in bralcem ni treba posebej pojasnjevati, da nosi v tej zvezi "free" enak pomen kot v "free speech", ne pa kot v "free beer" (Williams 2002).

Copyleft je splošni koncept, dovoljenje za razširjanje in razmnoževanje je GNU General Public License (Free Software Foundation, 1991) ali na kratko GPL. Vsakemu programu iz projekta GNU je priložen izvod tega dokumenta.

Copyleft omogoča, da bo program, izdan pod temi pogoji, ostal vselej dostopen pod enakimi pogoji. Če smo na primer v urejevalnik GNU Emacs dodali možnost vključevanja slik v besedilo, tako spremenjenega programa ne moremo ne brezplačno in ne proti plačilu zapreti in ponujati samo prevedene kode. Lahko ga le ponudimo naprej pod enakimi pogoji, kot smo ga sami dobili, z izvorno kodo vred. Rezultat je to, za kar si prizadeva FSF: širitev baze prostega programja.

Copyleft in GPL nista edina načina za objavo odprtega programja. Nekateri drugi avtorji svoje sicer odprto programje izdajajo pod ohlapnejšimi pogoji, ki dopuščajo tudi komercialno izrabo. Verjetno najbolj znana takšna primera sta operacijski sistem BSD Unix in okensko okolje X Window System. Enega in drugega lahko dobite kot odprto verzijo, obstajajo pa tudi komercialne. Na nekaterih delovnih postajah in grafičnih karticah tečejo izključno komercialne izvedbe X Window System. Če smo lastnik ene takih, potem za nas X Window System ni odprto programje.

Proti radikalni načelnosti fundacije za prosto programje se je v začetku 1998 postavila bolj pragmatična iniciativa za odprto programje (Open Source Initiative 1998), ki je v svojih vrstah zbrala nekaj vidnejših akterjev iz skupnosti Linuxa. Skupini imata iste cilje, vendar uporabljata drugačno retoriko. Medtem ko fundacija za prosto programje poudarja načelne vidike, se je iniciativa za odprto programje osredotočila na rezultate, kar se je pri komunikaciji s poslovno sfero izkazalo kot uspešnejše.

Vse zgoraj naštete zvrsti programja spadajo med odprto programje: programi v javni lasti, prosti programi, izdani pod pogoji copylefta (med njimi so vsi programi, izdani pod pogoji GNU GPL), in odprti programi, izdani pod drugačnimi pogoji. Skupina programov, ki ne ustreza prej zapisanim petim pogojem za odprto programje, sodi v skupino zaprtega (angl. proprietary) programja. Mednje spada večina komercialnega programja ter programov na pokušino (angl. shareware). Slednji sicer navadno dovoljujejo prosto razširjanje, praviloma pa niso dostopni v izvorni kodi in avtorji za njihovo uporabo zahtevajo plačilo.

Pregled licenc

Nekaj licenc, ki jih je iniciativa za odprto programje odobrila kot primerne:

- **GNU General Public License (GPL)**. Restriktivna in "dedna" – prenaša se na izvedena dela. Če v svoj program vgradimo kos programja, izdanega pod GPL, pade tudi naš program pod to licenco.
- **GNU Lesser General Public License (LGPL)**. Nekdaj imenovana GNU Library General Public License. Dopušča povezovanje knjižnic, izdanih pod LGPL, z drugimi programi, izdanimi pod komercialnimi licencami.
- **BSD License**. Permisivna, vztraja samo pri navedbi avtorstva.
- **X Consortium License**. Permisivna, glavni namen je popularizirati paket.
- **Artistic License**. Permisivna, pod določenimi pogoji dovoljuje tudi razširjanje le prevedenih programov.
- **Mozilla Public License (MPL)**. Podobna kot GPL, vendar ni "dedna" – omogoča dodajanje ne-prostega programja. Ni isto kot Netscape Public License, ki postavlja razvijalce v neenakopraven položaj glede na Netscape Communications Corp.
- **Q Public License**. Troll Tech izdaja knjižnico Qt pod dvojno licenco – QPL izključno za nepridobitno uporabo in komercialno licenco za pridobitno rabo.

Še druge licence, ki jih iniciativa za odprto programje priznava kot primerne, so: Academic Free License, Apache Software License, Apple Public Source License, Attribution Assurance Licenses, Common Public License, Eiffel Forum License, IBM Public License, Intel Open Source License, Jabber Open Source License, MITRE Collaborative Virtual Workspace License (CVW License), Motosoto License, Nethack General Public License, Nokia Open Source License, OCLC Research Public License, Open Group Test Suite License, Open Software License, Python license (CNRI Python License), Python Software Foundation License, Ricoh Source Code Public License, Sleepycat

License, Sun Industry Standards Source License (SISSL), Sun Public License, Sybase Open Watcom Public License, University of Illinois/NCSA Open Source License, Vovida Software License v. 1.0, W3C License, wxWindows Library License, X.Net License, Zope Public License in zlib/libpng license.

Kdo piše odprte programe? So to hekerji?

Da in ne, odvisno od tega, kaj razumemo pod izrazom heker. Večinoma se ta izraz povezuje z računalniškim kriminalom in vdiranjem v računalniške sisteme. Vdiralci v sisteme ne pišejo odprtih programov in navadno tudi nobenih drugih ne. Z redkimi častnimi izjemami namreč večina vlomilcev v računalniške sisteme pravzaprav sploh ne zna posebej dobro programirati in za vlamljanje večinoma uporabljajo programe, ki jih je napisal kdo drug.

V prvotnem pomenu izraz heker označuje zanesenjaka, ki živi in diha za programiranje in računalnike (Levy 2001, Raymond 1996, Himanen 2001). Odličnost pri pisanju programov je edini način za doseganje ugleda v tej skupnosti; položaj, naziv, starost in spol so drugotnega pomena, če sploh kaj veljajo. Hekerji so anarhistično svobodnjaška druščina, stara skoraj toliko kot računalniki sami. Izhodišče hekerstva je, da je pravica do programiranja neodtujljiva človekova pravica. Računalniki so koristni in dostop do njih mora biti neomejen za vse, ki si tega želijo. Vse potrebne informacije, ki jih potrebujemo za programiranje — izvorna koda, dokumentacije in podobno — morajo biti dostopne. Če kot hekerje razumemo te druge, potem drži, da odprte programe pišejo hekerji.

Prednosti odprtih programov

Odprti programi imajo glede na klasičen model razvoja programov nekaj prednosti:

- *Eksponentna rast.* Odprti programi so zares zaživeli šele z internetom, ki je povezal nadkritično maso ljudi. Njihovo število je dovolj veliko, da imajo realno možnost za uspeh tudi specializirani projekti, ki sicer zanimajo manj ljudi. Res veliki projekti pa v svoji kategoriji pritegnejo večino kreativnega potenciala.
- *Dolgoročna kredibilnost.* Če ima na milijone ljudi izvorno kodo določenega programa, je manj verjetno, da bo program čez noč izginil. Na primer: manj verjetno je, da bo nenadno izginotje doletelo spletni strežnik Apache kot pa urejevalnik WordPerfect. Tudi če bi celotna ekipa, ki vodi razvoj strežnika, čez noč iz neznanih razlogov izgubila zanimanje zanj in ga zapustila, se bi prej ali slej našel kdo, ki bo vzel v roke izvorno kodo in nadaljeval, kjer je bilo njeno delo končano. Če razvijalec opusti zaprt projekt, je z njim bolj ali manj konec.
- *Vzporedno odpravljanje napak.* Raymond je poimenoval to Linusov zakon: "Pri dovolj opazovalcih so vse napake v programu očitne." Ali kot je formuliral Jeff Dutky: "Odpravljanje napak je moč paralelizirati." Ta stopnja je morda ključna, zakaj model odprtega programja kljubuje Brooksovemu zakonu. Množičnost pomeni, da ni nujno, da tisti, ki napako najde, ve tudi, kako jo odpraviti, vendar se potem, ko je napaka diagnosticirana, prej ali slej najde kdo, ki to zna. Množičnost zagotavlja tudi, da je čas med objavo odkritja napake in predlogom za njeno rešitev bistveno krajši kot pri komercialnih ponudnikih programja.
- *Vzporedni razvoj.* V množici programerjev na internetu, ki samo čakajo na primeren izziv, je razvoj programov takorekoč zastonj, ekonomsko je upravičen hkratni razvoj več prototipnih rešitev. Tako trenutno na primer poteka razvoj namizij KDE in GNOME, dveh zvočnih podsistemov za Linux (OSS in ALSA) itn.
- *Odprto programje je recenzirano.* V znanosti so objave v vseh pomembnih revijah recenzirane. Preden gre članek v objavo, ga navadno pregleda več neodvisnih strokovnjakov ter poskuša v njem najti napake v razmišljanju in opozori avtorja na mogoče izboljšave. Šele po tem postopku gredo članki v objavo. Dostopnost izvorne kode odprtih programov omogoča enak postopek – kdorkoli se lahko poglobi v ustroj programa ter opazi morebitne napake ali pa predlaga izboljšave. Internet omogoča, da je med vsemi, ki si lahko ogledajo kodo, tudi dovolj strokovnjakov. Programi, katerih izvorna koda je dostopna dovolj dolgo, lahko tako doseže zavidljivo raven zanesljivosti. Primer za to sta stavni program TeX in pripadajoči program za izdelavo pisav Metafont, ki izvirata iz druge polovice sedemdesetih let. Njun avtor, profesor Donald Knuth, vodi tudi zanimiv način nagrajevanja poročil o napakah. Opis doslej še neodkrite napake v programu je prinesel najditelju napake ček profesorja Knutha za sprva simboličen 1 cent. Sorazmerno z dozorevanjem kode se je podvo-

jevala tudi nagrada. Za dele programov TeX in Metafont, napisane pred letom 1989, trenutno znaša 327 dolarjev in 68 centov.

Eden pogostih strahov v zvezi s projekti odprtega programja je nenadomestljivost avtorja: kaj se bo zgodilo z Linuxom, če se ga Linus Torvalds nekega dne naveliča? Kot podkrepitev navajajo obdobje v letu 1998, ko je zaradi Torvaldsove preobremenjenosti, osebnih razlogov in pomanjkanja časa razvoj Linuxa za krajši čas nekoliko zastal. Strah ni povsem odveč, ni pa vezan le na odprte projekte: kaj se bo zgodilo, če David Cutler, idejni oče Windows NT, po petnajstih letih spet zamenja delodajalca? Z izkušnjami, pridobljenimi v kriznem obdobju, je skupnost razvijalcev Linuxa razvila nekoliko stabilnejši način vodenja projekta. Novi koncept je uspešno prestal ognjeni krst jeseni 2001, ko je Alana Coxa kot človeka na poziciji številka dve brez večjih pretresov nadomestil Marcelo Tosatti. Na čelu večine večjih odprtih projektov so skupine, kar zmanjšuje pomen posameznika. Za manjše projekte to ne velja, vendar pa je te zaradi manjše kompleksnosti tudi lažje "posvojiti", če jih prvotni avtorji zapustijo.

Drugi strah je strah pred fragmentacijo, h kateri navidez nezadržno vodijo odprte licence. Model odprtega programja je za cepljenja in drobljenja odpornejši, kot se zdi. Linux s popolnoma odprtim razvojem lahko v tem pogledu primerjamo z drugim odprtim Unixom – BSD, ki je sicer tudi prosto dosegljiv v izvorni kodi, a ga razvijajo zaprte skupine. Linux je po skoraj desetih letih ostal enoten (imamo sicer nepreštevne "distribucije", vendar gre pri tem samo za garniranje istega skupnega jedra). BSD se je razcepil na 386BSD, BSDI, FreeBSD, NetBSD in OpenBSD. Prvi je medtem že izumrl, razvoj drugih pa poteka ločeno.

Kdaj lahko uspe projekt odprtega programja?

Redki odprti projekti zrastejo do razsežnosti, kot je na primer Linux. Navadno en (npr. Samba), dva (npr. KDE in GNOME) ali kvečjemu nekaj programov zasede določeno nišo, velika večina projektov pa zamre. Katere pogoje mora torej izpolnjevati projekt, da zaživi?

- Vsebina mora biti obenem dovolj dostopna ter dovolj zanimiva za razvijalce. Programi za krmiljenje manevrirnih raket ali nadzor jedrskih central ne zadoščajo prvemu pogoju, ker običajno dijaki in študenti nimajo doma ne enega ne drugega, da bi se lahko sami poizkušali v programiranju. Na drugi strani pa program za vodenje saldokontov ne predstavlja dovolj velikega izziva, da bi pritegnil širše množice. Za razliko od teh dveh pa npr. prevajalnik za Javo ali program za zapis MP3 zadovoljujeta oba pogoja.
- Pomembna je komunikacija z uporabniki. Pri uspešnih projektih vodje obravnavajo uporabnike kot sorazvijalce. Po elektronski pošti vse, ki so karkoli prispevali k projektu ali izrazili zanimanje zanj, obveščajo o kasnejših razvojnih različicah. Prednost odprtih programov je tudi njihova osebna nota – navadno se lahko obrnemo neposredno na avtorja namesto na posrednike. Potencialnega razvijalca v slabo voljo nič ne spravi tako kot to, da ne dobi nikakršnega odziva na svojo pošto s popravkom, ampak le-ta brez sledu ponikne na oddelku za stike s strankami. Tudi vnaprej pripravljeni odgovori v smislu "Spoštovani XY, prejeli smo popravke, ki ste jih poslali dne tega in tega in jih posredovali našemu razvojnemu oddelku. Hvala lepa in na svidenje" so komaj kaj boljši.
- Pogosto izdajanje novih, izpopolnjenih različic. Linux se razvoja z dinamiko, ki jo z zavistjo spremljajo komercialni proizvajalci programske opreme. Ključni prispevek k takšni dinamiki je dal avtor. V najbolj dinamičnih časih razvoja je tudi nekajkrat na teden izdal novo različico jedra, v kateri je upošteval do takrat prejete popravke in dopolnitve. S tem je določal in obdržal dinamiko razvoja.
- Uporabniki razvijalci na začetku potrebujejo nekaj, s čimer se da že kaj početi. Projekta ne moremo začeti tako, da v novičarski skupini razpredamo o tem, kakšen program bi radi napisali. Ljudem moramo ponuditi kaj, kar jih prepriča po eni strani o resnosti naših načrtov in po drugi strani o potencialih zastavljenega projekta, v katerih morajo videti tudi možnost za sodelovanje. Ena izmed glavnih kritik za začetne težave ob odprtju brskalnika Mozilla je bila ta, da je bila ob izidu zadeva neuporabna in se ni niti prevedla brez napak.
- Neposesivnost glede kode. Pomembno je, da si vodja projekta ne lasti kode. To pomeni, da zna prepoznati in sprejeti boljše rešitve, četudi na škodo svojih in da bo znal najti sposobnega naslednika, če prvotni avtor izgubi zanimanje za projekt ali ga ne bo več zmožen voditi.

Specifika programja kot ekonomske dobrine

Pojdimo od sociologije k ekonomiji in k tretjemu Raymondovemu članku (Raymond 1999). Enako kot drugi izdelki imajo programi dve ločeni zvrsti ekonomske vrednosti: prva je njihova tržna (vrednost izdelka kot končne dobrine), druga pa uporabna vrednost (vmesna dobrina). Zgledovanje po drugih industrijskih panogah pri analizi ekonomike pisanja programov zapelje v dve predpostavki:

- Programerji so večinoma plačani za ustvarjanje tržne vrednosti.
- Tržna vrednost programa je premosorazmerna razvojnim stroškom in njegovi uporabni vrednosti.

Nobena ni točna. Prva ne drži, ker so programi za nadaljnjo prodajo le zelo majhen del vsega napisanega programja – Raymond navaja oceno 5 % (Raymond 1999). Večino svojih delovnih ur prebijejo programerji ob pisanju in vzdrževanju programov, ki nikoli ne ugledajo luči sveta, ampak so zgolj za interno rabo.

Druga predpostavka v resnici velja za mnoge druge dobrine. Opazovanje obnašanja kupcev nas uči, da je pri programih drugače. Za primer vzemimo programski izdelek, katerega proizvajalec je propadel, ali pa manj dramatično – izdelek, ki ga je proizvajalec opustil. Vloženi razvojni stroški se s tem niso nič spremenili, niti se ni spremenila teoretična uporabna vrednost izdelka. Kljub temu zdrsne tržna vrednost takega izdelka proti nič. Kako to? Pri nakupu vrtne škropilnice ali šampona za lase kupci navadno niso posebej pozorni na to, ali ta model izdelovalec še izdeluje. Mnogi izdelki po propadu izdelovalca med zbiratelji dosežejo celo višje cene. Programi nikoli.

Kupci večinoma kupujejo program zaradi pričakovane zagotovitve vzdrževanja, kar ni obsega le tehnične pomoči, temveč tudi izdajo popravkov, gonilnikov za novejše naprave ipd. Industrija programske opreme je torej večinoma storitvena dejavnost, ki živi v iluziji, da je proizvodna dejavnost.

Če vzamemo ustaljeno vrednost, da je v tipičnem življenjskem ciklu programskega izdelka 75 % vseh stroškov vzdrževanje, odprava napak in izdelava razširitev, to pomeni, da je običajna tržna shema z visoko začetno ceno programa in brezplačno podporo slaba za obe strani: za kupca, saj v centrih za pomoč uporabnikov, ki v tej shemi niso dobičkonosni, praviloma pristanejo manj sposobni strokovnjaki, ki se jim da na voljo le najnujnejše vire.

V večini primerov je slaba tudi za proizvajalca. Financiranje podpore iz enkratnega zneska začetne cene je finančno smiselno samo pri hitro razvijajočem se trgu. Ko se trg ustali, je neogibno krčenje stroškov na račun podpore. Rezultat je enak najsi se to zgodi neposredno – z opustitvijo izdelka – ali posredno – tako, da je težko dobiti podporo za izdelek. Ker smo s tem izdelku uničili pričakovano prihodnjo vrednost, kupci uidejo drugam. Nekaj časa lahko sicer kupcem popravke predstavljamo kot nov izdelek, trajna rešitev pa je samo ena: pridobiti si moramo monopolni položaj na tržišču. V tem primeru kupci nimajo kam uiti. Zgodovina dovolj zgovorno postreže s primeri izdelkov, ki so bili kljub začetnemu solidnemu drugemu mestu izrinjeni iz niše: DR-DOS, WordStar, Quattro Pro ipd.

Trajnostni razvoj

Ali je mogoč trajnostni razvoj odprtega programja? Ali obstaja nevarnost, da bodo iz skupnega fonda odprtega programja želeli vsi samo črpati, nihče pa ne bo prispeval?

Nekateri si fond odprtega programja predstavljajo kot skupni pašnik, kjer vaščani pasejo živino. Paša mora biti zmerna, sicer pašnik zmendrajo v blaten dol, ki se le počasi spet zaraste. Tu imamo klasičen primer zapornikove dileme (Axelrod 1985): za vaščane kot celoto je seveda najbolje, če so pri izkoriščanju pašnika zmerni, medtem ko je odločitev posameznika ravno nasprotna: čimprej mora čimveč živine poslati na pašnik, dokler ta še ni zmendran v blato. Če je ne bo sam, jo bo sosed, ki razmišlja enako.

Opažanja kažejo, da izbrana ilustracija zgreši bistvo. Prvič, priliv odprtih programov se ves čas povečuje in ne zmanjšuje. In drugič, programov z rabo ne obrabimo. Ravno nasprotno, če jih odpremo, s tem spodbudimo druge uporabnike razvijalce, da prispevajo svoje popravke in dopolnitve. V zgornji prispodobi bi to pomenilo, da bolj ko pasemo, bolj sočna in zelena je trava.

Pa vendar, denimo, da smo napisali popravek, s katerim odpravimo nadvse nadležno napako v programu, ki ga uporablja veliko ljudi. Mnogi od njih se strinjajo, da ima popravek določeno tržno vrednost. Kako pobrati denar? Težava ni le v tem, da imajo vsi obstoječi sistemi prenosa denarja dosti prevelike lastne stroške, da bi bila takšna mikroplačila izvedljiva. Težava je, da je pravzaprav zelo težko zares oceniti vrednost takega dela.

Kaj lahko torej storimo? Popravek lahko ljubosumno hranimo zase ali pa ga zastonj prispevamo v

skupni sklad odprtega programja. V prvem primeru ne pridobimo nič. V drugem primeru morda tudi ne, morda pa bomo s svojim dejanjem spodbudili še koga drugega, da bo prispeval kak drug popravek, ki nas muči. Teorija iger pokaže, da je v takem idealiziranem primeru druga možnost, čeprav navidez altruistična, v resnici optimalna.

Ker se programi ne obrabljajo, je v tovrstnem sodelovanju pravzaprav pomembno le število tistih, ki prispevajo k projektu. Sorazmerno narašča tudi zahtevnost vodenja projekta. Naraščajoče število tistih, ki iz skupnega fonda samo črpajo, resda povečuje delež neprimernih vprašanj na dopisnih listah ali v novičarskih skupinah, kar pa lahko odpravimo tako, da sestavimo spisek pogosto zastavljanih vprašanj z odgovori in ignoriramo tiste, ki so vprašali, ne da bi ga prebrali. Pozitivni prispevek prvih je torej linearen, (negativni) prispevek drugih pa konstanten.

Kaj varujemo z zaprto kodo?

Denimo, da imamo napisan računovodski programski paket. V računovodstvu ni nobenih posebnih algoritmov, ki bi jih kazalo skrivati. Možna odgovora, kaj varujemo z zaprto kodo, sta torej dva: s tem varujemo njegovo tržno vrednost, ali pa ne želimo, da pride izvorna koda v roke konkurenci.

Prvi argument je veljaven za tisti manjšinski delež programov, ki so v resnici namenjeni trgu, ne pa za programe, ki jih uporabljamo sami.

Drugi argument je težji. Konkurenca res dobi izvorno kodo našega programa, namesto da bi jo morala razviti sama, vendar pa lahko na ta način pridobimo tudi nove zunanje sodelavce, ki nam bodo morda pomagali s popravki in dopolnitvami. Premisliti moramo torej, ali dobre plati odtehtajo slabe.

Kot tretji razlog mnogi navajajo tudi varnost, posebej če gre za kriptografske metode. Razmišljanje "obskurnost pomeni varnost" je tiščanje glave v pesek. Če kje, potem je prav pri kriptografskih metodah pomemben neodvisen zunanji strokovni pregled uporabljenega postopka, če naj se ne slepimo z utvaro o varnosti.

Posredna tržna vrednost odprtega programja

Raymond navaja nekaj zgledov, kako lahko s programi z dostopno izvorno kodo ustvarjamo posredno tržno vrednost.

- *Ustvarjanje ali držanje tržne niše.* Če nastopamo v dveh soodvisnih tržnih segmentih, lahko zapustimo manj donosnega in se osredotočimo na bolj donosnega. Če na primer prodajamo brskalnike in strežnike, odpremo brskalnike in se osredotočimo na donosnejši del: strežnike, oglaševanje na internetni vstopni točki ali naročnino. Za tak korak se je spomladi 1998 odločila družba Netscape Communications.
- *Gonilniki.* Nujno zlo za vse proizvajalce naprav je pisanje gonilnikov. Trg zahteva, da jih pišejo in vzdržujejo, kljub temu pa to ni vir dohodka, saj živijo od prodaje strojne opreme. Z odprtjem izvorne kode gonilnikov lahko proizvajalec pridobi širšo bazo sodelavcev, kar pomeni hitrejše prilagajanje, zaradi neodvisnega zunanjega pregleda pa tudi večjo zanesljivost. Zadržek, da s tem izdamo pomembne informacije o zgradbi svoje strojne opreme, je s skrajšanjem razvojnih ciklov strojne opreme tudi odpadel. Konkurenca lahko iz izvorne kode morda res razbere zgradbo opreme, vendar pa gre čas, ko so se ukvarjali z analizo našega izdelka, na račun razvoja njihovega. Plagiatorstvo je past. Glede na kratko življenjsko dobo strojne opreme je model dostopne izvorne kode zanjo še posebej primeren. Ko izdelovalec napravo opusti, so kupci prepuščeni sami sebi. Če so gonilniki dostopni, je nekaj več možnosti za odpravo napak tudi potem, ko proizvajalec izdelek preneha vzdrževati, kar bo morda pritegnilo kakega kupca ... Tako odločitev je marca 1999 sprejela družba Apple Computer.
- *Objavimo recept, odpremo restavracijo.* V tem primeru odprto programje drži odprto nišo za storitve. Tako obratujejo hiše, kot so Red Hat, Caldera ali SuSE, ki garnirajo odprto programje v enostavno uporabne pakete.
- *Potrebščine.* Ena spremljevalnih dobičkonosnih dejavnosti ob odprtem programju je tudi prodaja potrebščin: od majic, vrčkov za kavo in kap do profesionalno urejene dokumentacije. Primer slednje je založba O'Reilly.
- *Sprostimo prihodnost, prodajamo sedanjost.* Program izdamo skupaj z izvorno kodo pod zaprto licenco, vendar časovno omejeno, denimo takšno, ki dopušča le nepridobitno izrabo, a obenem zagotavlja, da program zapade pod GPL eno leto po izidu, ali pa, če ga proizvajalec opusti. V tem primeru so stranke lahko prepričane, da je program karseda prilagodljiv njihovim potrebam, saj imajo izvorno kodo. Izdelek nosi tudi določena zagotovila glede

prihodnosti. Slednje lahko vliva strankam več zaupanja v izdelek, kot bi ga, če bi bil ta izdan pod zaprto licenco, odprtost izvorne kode pa seveda prinese tudi neodvisni zunanji pregled in s tem večjo zanesljivost. Tak pristop uporablja Aladdin Enterprises pri programu Ghostscript.

- *Sprostimo programje, tržimo blagovno znamko.* Model je zaenkrat špekulativen. V njem odpremo program, vendar zadržimo testni nabor in uporabnikom na njihovo željo prodamo pravico do uporabe blagovne znamke, ki zagotavlja, da je njihov izdelek združljiv z vsemi, ki uporabljajo isto blagovno znamko. Na ta način bi Sun Microsystems lahko tržil Javo in Jini.
- *Sprostimo programje, tržimo vsebino.* Nepreverjeni model, primeren za večje ponudnike internetskih storitev, kot je na primer AOL, ki je razvil tudi lastni spletni brskalnik. Pri njem vrednost ni v kodi brskalnikov ali strežnikov, temveč v vsebini. Kodo brskalnika bi lahko odprli in prodajali izključno naročnino na vsebino. Z neodvisnim prenosom brskalnika v novo strojno okolje seveda samo pridobimo, ker se razširi krog potencialnih strank.

Kaj pri programju sploh prodajamo?

V modelu zaprtega programja je stvar jasna. Zavestno se odrečemo neodvisnemu zunanjemu vpogledu v kodo in pobiramo dividendo na algoritme in ideje, skrite v programu. Do pred nekaj leti je prevladovalo mnenje, da je to tudi edini način dobička v industriji programske opreme. Vzpon odprtega programja nas sili v to, da začnemo razmišljati tudi o bolj subtilnem in posrednem donosu odprtega programja.

Model odprtega programja prek postopka neodvisnega zunanjega pregleda izvorne kode zagotavlja visoko raven kakovosti in zanesljivosti ter se dobro prilagaja velikosti. Če je izpad dohodka iz trženja programskih skrivnosti mogoče nadomestiti s prihodkom od storitev, dodatkov in postranskih trgov, vezanih na programje, je verjetno prehod na model odprtega programja pametna odločitev. Odprto programje je tudi eden od načinov za podporo neodvisnim odprtim standardom, kot je na primer TCP/IP. Hitro rast interneta je brez dvoma omogočilo tudi dejstvo, da je odprt, ne pa lastniški standard. Vsekakor pa je dogajanje, ki je sledilo odprtju brskalnika Netscape spomladi 1998, ovrglo mnenje, da od odprtega programja ni moč pričakovati nobenega donosa.

V prid modelu odprtega programja govorijo naslednji dejavniki:

- Zanesljivost/stabilnost/skalabilnost so kritični parametri.
- Pravilnosti zasnove in izvedbe ni mogoče enostavno verificirati drugače kot z neodvisno recenzijo. Vsak količkaj zahteven program zadošča temu kriteriju.
- Programje je bistveno za nadzor nad uporabnikovim poslom. Ali drugače, uporabnik si ne more privoščiti popolne odvisnosti od proizvajalca programske opreme.
- Programje vzpostavlja ali omogoča enotno računsko ali komunikacijsko infrastrukturo.
- Ključni postopki ali njihovi funkcionalni ekvivalenti so del splošnega inženirskega znanja.

Linux ali Apache sta vzorčna primera, ki zadoščata vsem petim kriterijem. Poučen je tudi razvoj rabe omrežnih protokolov. Po nekaj neuspešnih poskusih izgradnje imperija s protokoli DECNET, XNS, IPX in podobnimi je sredi devetdesetih prevladal najstarejši med njimi – TCP/IP – ki edini temelji na odprti kodi.

Kot nasprotje temu navaja Raymond primer družbe, ki trži program za lesne obrate – optimizacija razreza hlodov. Ta primer približno zadošča le tretjemu kriteriju, zato ni verjetno, da bi z odprtjem bistveno pridobil.

Če naj verjamemo današnjim trendom, kaže, da bo eden večjih izzivov projektnega vodenja v naslednjih letih in desetletjih primerna izbira trenutka, ko se odločimo za odprtje kode. Če to storimo prezgodaj, zavržemo del prihodka, ki bi ga imeli s trženjem programskih skrivnosti. Če to storimo prepozno, je lahko prav tako slabo. Če nas pri tem v naši tržni niši z odprtjem kode prehiti drug proizvajalec, bo na svojo stran pritegnil najboljše razvijalce in z njimi tudi večino prednosti, ki jih prinaša model odprtega programja.

Nevarnosti, ki grozijo odprtemu programju

Oglejmo si še nekaj potencialnih nevarnosti, ki grozijo modelu odprtega programja. Večino smo ovrgli že sproti, kot denimo strah pred fragmentacijo Linuxa. Dve nevarnosti pa vendarle obstajata. V svojem poročilu jih je navedel Valloppillil (Valloppillil 1998) kot možnosti, s katerimi si lahko monopolisti zagotovijo svoj položaj še naprej.

- *Prisvajanje protokolov.* Linux in internet sta bila tako uspešna, ker temeljita na javno dostopnih protokolih (TCP/IP, SMTP, HTTP, POP3, IMAP, NFS

idr.). Tu so interesi monopolistov in skupnosti, ki podpira odprto programje (pa tudi vseh manjših proizvajalcev programske opreme, ki niso v monopolnem položaju), v konfliktu. V interesu monopolista je zamenjava javnih protokolov z lastniškimi, nad katerimi bdi sam namesto IETF.

- *Patenti nad programi.* Evropska unija za razliko od ZDA ne priznava patentov nad programi. Ti so celo v ZDA predmet žolčnih polemik, saj varujejo izključno največje proizvajalce, ki si lahko privoščijo vzajemno licenciranje množice patentov. Celo nekateri večji ameriški proizvajalci, kot sta Adobe in Oracle, so izrazili mnenje, da patenti na programsko opremo inovativnosti bolj škodijo kot koristijo (Freepatents.org). Kljub temu skupine, kot je denimo BSE, izvajajo na Evropsko patentno organizacijo (EPO) velik pritisk, da se evropska zakonodaja uskladi z ameriško in spremeni člen 52.2 Evropske patentne konvencije (EPC), ki računalniške programe izvzema iz skupine izdelkov, ki jih je moč patentirati. Oktobra 2001 je tako EPO že delno popustila in sprejela določilo, s katerim je dovolila odobritev patentov, ki izrecno navajajo programsko kodo ali podatkovne strukture.

Življenje po revoluciji

Ideja odprtega programja ima pridih revolucionarnega in uporniškega. Ali pa lahko ideja preživi tudi na daljši rok? Ni razlogov, da ne. Obenem pa seveda tudi ne smemo pričakovati odprtja čisto vsega programja.

Programe lahko razvrstimo v nekaj skupin. V prvi je obče programje, ki vzpostavlja osnovno tehnično infrastrukturo (operacijski sistemi, omrežne komunikacije) in uporablja javno dostopne protokole. Pričakujemo lahko, da bo ta skupina programja ostala in še v večji meri postala odprto programje, katere razvoj bodo vodili konzorciji uporabnikov in/ali založniško/storitvena podjetja, kot je na primer Red Hat.

Antipod tej skupini so namenski programi, npr. urejevalniki besedil, ki delujejo v svetu neobstoječih ali šibkih tehničnih standardov. V tej skupini je največ programov, v kateri dodatna zanesljivost, pridobljena z neodvisnim zunanjim vpogledom v kodo, morda ne bo odtehtala vrednosti tajnih algoritmov ali tehnik.

Značilen predstavnik vmesne skupine so programi za obdelavo podatkovnih zbirk potem, ko je uporaba SQL odlepila vmesnike od strežniškega dela. Odprtje tega segmenta je odvisno od cene nezanesljivosti. Višja ko je, večji bo pritisk trga po odprtju.

Prehod na odprto programje seveda ne pomeni konca industrije programske opreme, kot jo poznamo danes, pomeni pa njeno prestrukturiranje in zasedbo novih niš, ki jih prinaša sprememba. S stališča uporabnika programja pa je seveda tak razvoj ugoden, saj s tem, ko bo vrsta programov živela naprej, namesto da bi bila opuščena, omogoča večjo izbiro.

Literatura

Axelrod, R. (1985): The Evolution of Cooperation, Basic Books, New York.

Brooks, F. P. Jr. (1975): The Mythical Man-Month: Essays on Software Engineering, Addison-Wesley, Reading.

Free Software Foundation (1991): GNU General Public License, 2. izdaja. http://www.gnu.org/copyleft/gpl.html.

Freepatents.org – Protecting Competition Against the Abuse of Software Patents, http://www.freepatents.org/.

Himanen, P. (2001): The Hacker Ethics and the Spirit of the Information Age, Secker & Warburg, London.

Levy, S. (2001): Hackers: Heroes of the Computer Revolution, 2. izdaja, Penguin USA, New York.

Open Source Initiative (1998): The Open Source Page, http://www.opensource.org/.

Raymond, E. S., ur. (1996): The New Hacker's Dictionary, 3. izdaja, MIT Press, Cambridge.

Raymond, E. S. (1997): The Cathedral and the Bazaar, http://www.tuxedo.org/~esr/writings/cathedral-bazaar/.

Raymond, E. S. (1998): Homesteading the Noosphere, http://www.tuxedo.org/~esr/writings/homesteading/.

Raymond, E. S. (1999): The Magic Cauldron, http://www.tuxedo.org/~esr/writings/magic-cauldron.

Raymond, E. S. (2001): The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, 2. izdaja, O'Reilly & Associates, Sebastopol.

Valloppillil, V. (1998): Open Source Software – A (New?) Development Methodology", http://www.opensource.org/halloween/halloween1.html.

Williams, S. (2002): Free as in Freedom: Richard Stallman's Crusade for Free Software, O'Reilly & Associates, Sebastopol. http://www.oreilly.com/openbook/freedom/.

Primož Peterlin je med uporabniki odprtega programja znan predvsem kot avtor navodil za prilagoditev operacijskega sistema Linux slovenskemu jezikovnemu okolju. Sam ali s sodelavci je v domačih in tujih revijah objavil več člankov o Linuxu, skupaj s sodelavci pa tudi dva priročnika za delo z njim. Po izobrazbi je doktor fizike in se ukvarja z biofizikalnimi raziskavami mehanskih lastnosti celičnih membran.

Informatizacija procesa upravljanja človeških virov

Miroslav Ribič, Marjan Lončarič, Andrej Kovačič

Povzetek

Učinkovito upravljanje človeških virov temelji na spoznanju, da je človek najpomembnejši kritični dejavnik uspeha pri doseganju ciljev organizacije. Predmet obravnave so procesi, ki sodijo v sklop upravljanja človeških virov in so potrebni za njihovo pridobivanje, ohranjanje in motiviranje, ter informatizacija, ki omogoča njihovo učinkovito izvajanje in vodi k večjim koristim tako za posameznika kot za organizacijo in družbo.

Abstract

Informatization of Human Resource Management

Effective human resource management reflects the recognition of the importance of human labour, which represents the most important success factor to attain organization's goals. In this article we present functions and activities, that are needed to attract qualified job applicants, to retain desirable employees and to motivate them. We will also indicate, how to implement information system, that supports such functions and enables that human resources are used effectively.

1 Proces upravljanja človeških virov

Upravljanje človeških virov temelji na spoznanju, da je človek najpomembnejši kritični dejavnik uspeha pri doseganju ciljev organizacije. Glavni namen je zagotoviti ustrezno število ustrezno kvalificiranih delavcev, ki so nujno potrebni za realizacijo poslovnih ciljev organizacije. Sestavljajo ga naslednji procesi: planiranje, ocenjevanje, pridobivanje in razvoj kadrov, nagrajevanje, izboljševanje kakovosti življenja v organizaciji ter merjenje izvajanja strategije upravljanja s človeškimi viri, ki s povratnimi informacijami za proces planiranja kadrov zaključi zanko v procesu upravljanja človeških virov.

1.1 Planiranje kadrov

Strateško planiranje v organizaciji lahko opredelimo kot neprekinjeno vrednotenje in izkoriščanje priložnosti za doseganje konkurenčne prednosti organizacije. Ker je realizacija strategij v celoti odvisna od človeških virov, morajo biti ti kvantitativno in kvalitativno vedno na voljo. Planiranje kadrov sestavljajo tri faze, in sicer: napoved povpraševanja in ponudbe človeških virov, oblikovanje ciljev in strategij upravljanja človeških virov, s katerimi je mogoče odpraviti odmike med napovedjo povpraševanja in ponudbe, ter načrtovanje proračuna za realizacijo oblikovanih strategij. Oblikovane strategije praviloma določajo izvajanje naslednjih procesov: ocenjevanje človeških virov, pridobivanje (oz. odpuščanje) kadrov, nagrajevanje, razvoj kadrov ter izboljševanje kakovosti življenja v organizaciji.

1.2 Ocenjevanje človeških virov

Povečevanje produktivnosti je cilj vsake organizacije. Produktivnost ni samo funkcija uporabe sodobne tehnologije, optimizacije poslovnih procesov ipd., ampak je soodvisna od razpoložljivosti človeških virov. Delavčevo dejavnost je mogoče izmeriti in ovrednotiti, še zlasti njegovo delovno uspešnost (učinkovitost), odnos do dela, vodenje in izvajanje dela, inovativnost ipd. Zato je smiselno oblikovati ocenjevanje človeških virov, katerega namen je merjenje in vrednotenje. To je podlaga za planiranje in pridobivanje kadrov, njihovo nagrajevanje in izobraževanje, hkrati pa predstavlja učinkovito orodje za odkrivanje potencialov, ki jih je vredno pripraviti na večje odgovornosti.

1.3 Pridobivanje kadrov

Pridobivanje kadrov opredelimo kot aktivnost, ki je sestavljena iz iskanja, selekcije in razporeditve izbranih kandidatov. Podlaga za izvajanje te aktivnosti je plan kadrov oz. sistemizacija delovnih mest, ki temelji na njem. Namen pridobivanja kadrov je zagotoviti ustrezne delavce, potrebne za realizacijo ciljev organizacije.

1.4 Nagrajevanje

Nagrajevanje je orodje menedžerjev za upravljanje delavčeve aktivnosti. Pravimo, da je plača uravnotežena takrat, ko omogoča delavcu normalno življenje in vpliva na njegovo prizadevnost, ga motivira za delo. Organizacija na podlagi načrtovanih ciljev in veljavnih predpisov oblikuje strukturo plač. Z vrednotenjem zahtevnosti dela vsakega delovnega mesta dobi oceno, ki jo ustrezno vgradi v osnovno plačo. Organizacije, katerih struktura plače vsebuje le osnovno plačo, ne priznava individualnega prispevka in daje prednost podpovprečnim delavcem. Naprednejše nagrajevanje pozna poleg osnovne plače še dodatke, stimulacije, ugodnosti iz zdravstvenega in drugega zavarovanja, delnice organizacije in druge nagrade iz delovnega razmerja, npr. nagrajevanje po učinku, ki nagrajuje sposobne delavce, nagrajevanje po stažu, ki nagrajuje zvestobo in nagrajevanje po potrebi, ki nagrajuje delavce z visokimi življenjskimi stroški. Nobena od navedenih oblik ni popolna, na podlagi ciljev organizacije in veljavnih predpisov je treba najti ustrezno kombinacijo. Nagrajevanje je ponavadi vgrajeno v sistemizacijo delovnih mest, v ocenjevanje človeških virov in v obračun plač.

1.5 Razvoj kadrov

Proces razvoja kadrov razdelimo na dva podprocesa, in sicer načrtovanje kariere in izobraževanje. Kariero lahko opredelimo kot načrtovano ali nenačrtovano zaporedje dela ali aktivnosti, ki vključuje elemente napredovanja, samouresničevanja in osebnega razvoja v določenem času. Načrtovanje kariere je pomembno tako za posameznika v organizaciji kot za organizacijo. Karierni načrt omogoča posamezniku osebnostno in strokovno rast in razvoj ter ga usmerja pri odločanju o nadaljnjem izobraževanju. Za njegovo kariero je zainteresirana tudi organizacija, saj so zaposleni in njihove zmožnosti njen ključni konkurenčni element, ki ga lahko načrtno oblikuje in spreminja z načrtovanjem karier. Tako se ustvarjajo vzajemne koristi za organizacijo in posameznika, saj drug drugemu omogočata preživetje.

Proces izobraževanja lahko razumemo kot omilitev trenutnih in preprečevanje pričakovanih neučinkovitosti kadrov. Te so lahko posledica pomanjkanja splošnih in tehničnih znanj, nesposobnosti komuniciranja in razvijanja medsebojnih odnosov ali pa pomanjkljivih konceptualnih in povezovalnih znanj, kot so strateško in operativno planiranje, oblikovanje organizacije itn. Proces izobraževanja se prične z raziskovanjem izobraževalnih potreb in interesov ter s pripravo načrta izobraževanja.

1.6 Izboljševanje kakovosti življenja v organizaciji

Namen izboljševanja kakovosti življenja v organizaciji je organizirati delo tako, da bo vplivalo na učinkovito uporabo sposobnosti zaposlenih in na čimbolj učinkovito doseganje ciljev organizacije. Taka organizacija dela vodi k večjemu interesu za odločanje vseh zaposlenih in k večji samostojnosti pri delu. To lahko dosežemo z upoštevanjem smiselnih predlogov, pridobljenih s posebnimi vprašalniki, s sodelovanjem vsakega posameznika pri postavljanju delovnih ciljev in s skupnim oblikovanjem sistemizacije delovnih mest (slika 1).

Tako se poveča raznolikost sposobnosti, ki jih zahteva delo, zaposleni se laže poistovetijo s svojim delom, opravljajo ga bolj skrbno in kvalitetno, poveča se občutek pomembnosti dela, vse to pa vodi k višji stopnji motivacije. Na produktivnost vplivata poleg motiviranosti tudi varnost pri delu in zdravje zaposlenih. Zato je treba upoštevati pogostost in resnost nesreč in poklicnih bolezni, ki se pojavljajo pri določenem delu. Skladno s tem je treba pripraviti pravilnik o varstvu pri delu, popraviti sistemizacijo delovnih mest in urediti delovno okolje.

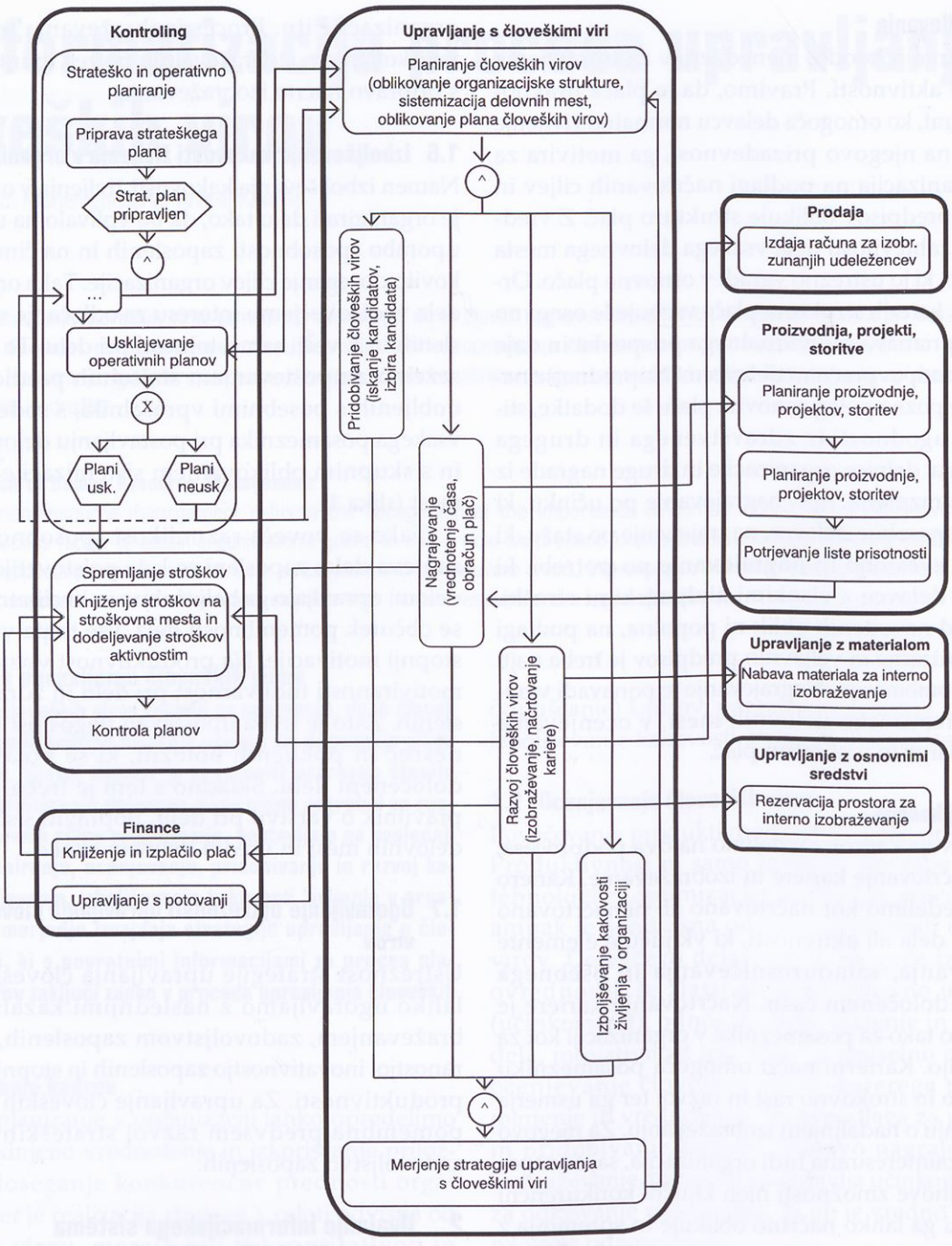
1.7 Ugotavljanje ustreznosti upravljanja človeških virov

Ustreznost strategije upravljanja človeških virov lahko ugotavljamo z naslednjimi kazalniki: izobraževanjem, zadovoljstvom zaposlenih, informiranostjo, inovativnostjo zaposlenih in stopnjo njihove produktivnosti. Za upravljanje človeških virov sta pomembna predvsem razvoj strateških znanj in zadovoljstvo zaposlenih.

2 Uvajanje informacijskega sistema za upravljanje človeških virov

Informacijski sistem za upravljanje človeških virov je treba obravnavati tako, da bo omogočena integracija vseh poslovnih procesov. Pri uvajanju je treba načrtovati zgradbo informacijskega sistema, ki vpliva na povezovanje z drugimi informacijskimi podsistemi in komponentami, ki podpirajo delovanje poslovnih procesov (slika 2).

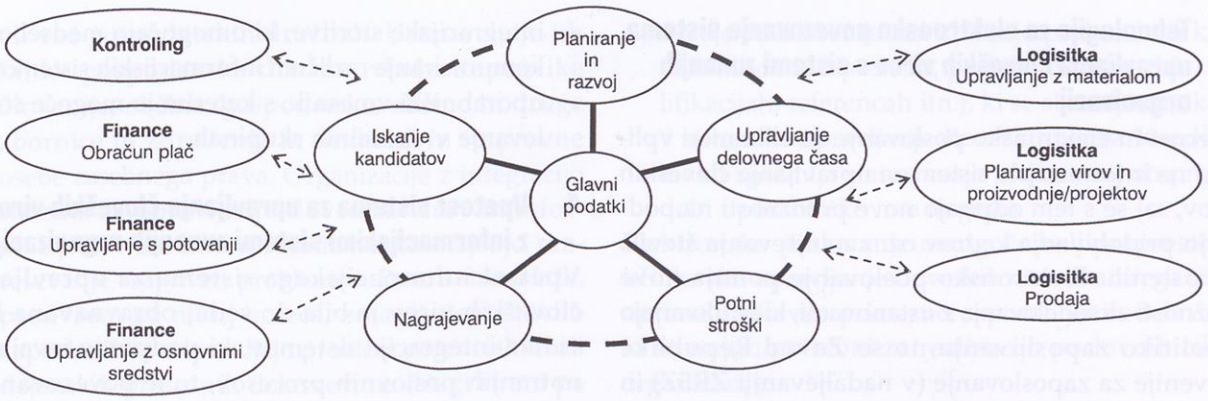
Integracija sistemov za upravljanje z materialnimi, finančnimi in človeškimi viri je zahtevna. Poznamo jo



Slika 1: **Povezovanje procesa upravljanja človeških virov z drugimi procesi organizacije**

kot celovit poslovno-informacijski sistem ERP (angl. enterprise resource planning). Zgradba sistema ERP mora biti vsaj trinivojska, saj je le tako mogoče ločeno obravnavati kompleksnost poslovnih procesov in oblikovanje podatkovne baze ter uporabniških vmesnikov.

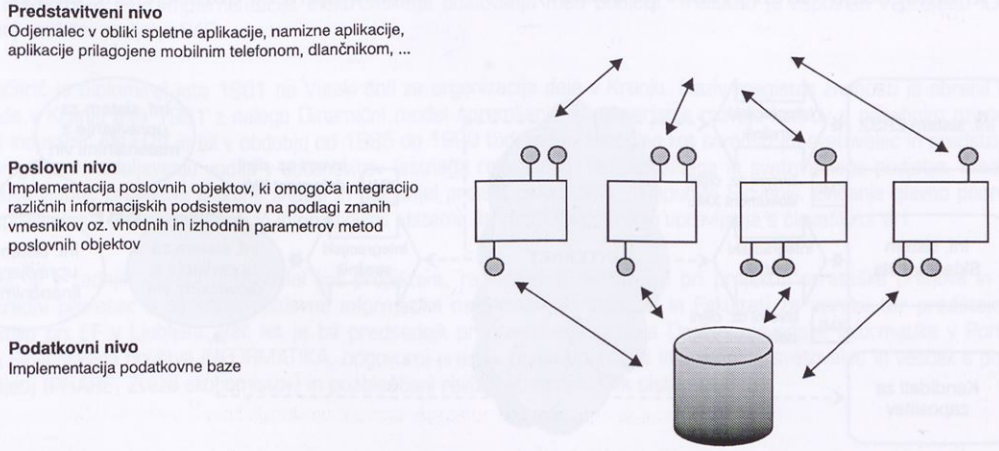
Odjemalec mora biti tako v celoti neodvisen od poslovne logike in strukture podatkovne baze. Vsebovati mora le oblikovanje obrazca, ki ga pri svojem delu uporablja uporabnik, ter sklice na metode ustreznih poslovnih objektov. Primer delovanja trinivojske zgradbe je tako naslednji. Če želi



Slika 2: **Integracija sistema za upravljanje človeških virov z drugimi informacijskimi podsistemi**

uporabnik popraviti podatke o zaposlenem delavcu, je treba te najprej vpisati na obrazec za podatke. Uporabnik iz seznama zaposlenih izbere zapis o delavcu, za katerega želi spremeniti podatke. Pri tem se izvede klic metode »Prikaz« poslovnega objekta »Delavec« z vhodnim parametrom »Šifra delavca«. Metoda »Prikaz« izvede poizvedbo v podatkovni bazi (stavek »SELECT«) in tako pridobi podatke o delavcu ter jih posreduje odjemalcu, ki podatke razporedi na ustrezna vnosna polja. Sedaj lahko uporabnik popravi podatke o izbranem delavcu. Ko se odloči, da bo popravljene podatke shranil, se izvede klic metode »Popravi« poslovnega objekta »Delavec«. Kot vhodne parametre je treba posredovati popravljene podatke o delavcu, na podlagi le-teh pa se izvedejo ustrezne spremembe v podatkovni bazi (stavek »UPDATE«).

Bistvo trinivojskega načrtovanja informacijskih sistemov je ločitev poslovne logike od uporabniškega vmesnika. Poslovna logika je upoštevana s pomočjo poslovnih objektov, ki so temeljne prvine poslovnih procesov, to so npr. delavec, naročilo, plačilo, material, kupec, dobavitelj itn. Za vsakega izmed njih lahko uporabimo vsaj štiri funkcije (kreiranje, popravljanje, brisanje in prikaz). Objekti praviloma nimajo lastnosti, saj jih je zaradi učinkovitejšega delovanja sistema bolje pretvoriti v vhodne oz. izhodne paramatre metod poslovnih objektov. Povezovanje informacijskih podsistemov je tako relativno preprosto. Sistemi se povezujejo tako, da z ustreznimi vhodnimi parametri uporabijo metode poslovnih objektov. Iz slike 1 je razvidno, da mora biti sistem za upravljanje človeških virov prek poslovnih objektov povezan s proizvodnim sistemom oz. sistemom za vodenje projektov, za finance in kontroling, za upravljanje z materialom in osnovnimi sredstvi ter s prodajnim sistemom (ko gre za izobraževanje zunanjih udeležencev).



Slika 3: **Zgradba informacijskega sistema ERP**

3 Tehnologije za elektronsko povezovanje sistema upravljanja človeških virov s sistemi zunanjih organizacij

Internet in elektronsko poslovanje v veliki meri vplivata na informacijski sistem za upravljanje človeških virov, saj se s tem odpirajo nove priložnosti na področju pridobivanja kadrov oz. zmanjševanja števila zaposlenih. Elektronsko poslovanje ponuja nove možnosti za sodelovanje z ustanovami, ki se ukvarjajo s politiko zaposlovanja, to so Zavod Republike Slovenije za zaposlovanje (v nadaljevanju ZRSZ) in skladi dela, ter tako podjetjem omogoča pregled nad trgom delovne sile (Lončarič, Antauer, Ribič, 2000). Prav tako lahko organizacija učinkovito izvaja strategijo zaposlovanja, saj relativno preprosto in poceni pride do podatkov o potencialnih kandidatih, ki se prijavijo na razpis za prosta delovna mesta prek spletnih strani, ki so del sistema za upravljanje človeških virov.

Prehod na elektronsko poslovanje ne pomeni zgolj podpreti obstoječe poslovne procese s spletnimi aplikacijami. Poslovne procese je treba temeljito prenoviti. Njihovo prenovo razumemo kot preverjanje procesov, postopkov in aktivnosti ter njihovo spremembo, ki jo sprožimo z namenom doseganja pozitivnih rezultatov pri zniževanju stroškov, povečanju kakovosti storitev, skrajševanju časovnih ciklov ipd. (Kovačič, 1998).

Uporaba elektronskega poslovanja za izvajanje aktivnosti za razreševanje presežnih delavcev zahteva:

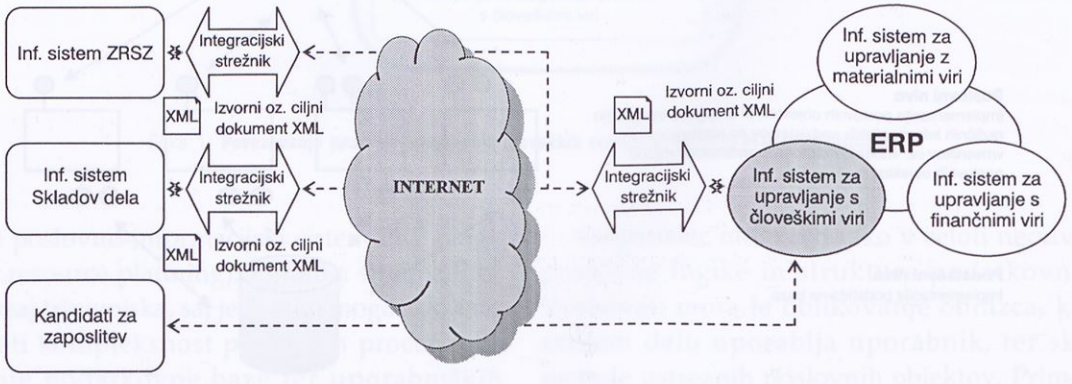
- omrežje, ki poveže informacijske sisteme,
- standard za opis in zapis podatkov,
- storitve, ki omogočajo poleg različnih funkcionalnosti tudi prenos podatkov,
- storitve, ki omogočajo varen prenos podatkov,
- integracijske storitve, ki omogočajo medsebojno komuniciranje različnih informacijskih sistemov in
- uporabniški vmesnik, s katerim je mogoče sodelovanje v virtualnih skupinah.

4 Vpetost sistema za upravljanje človeških virov z informacijskimi sistemi zunanjih organizacij

Vpetost informacijskega sistema za upravljanje človeških virov je bila do sedaj obravnavana le z vidika integracije sistemov, ki podpirajo izvajanje notranjih poslovnih procesov, to je povezovanje s sistemom za upravljanje z materialnimi viri in s sistemom za upravljanje s finančnimi viri. Vsi trije tvorijo celovit poslovno-informacijski sistem, s katerim je mogoče upravljati vse vire organizacije.

Z uporabo predstavljenih informacijskih tehnologij je mogoče sistem za upravljanje človeških virov učinkovito povezati z informacijskimi sistemi zunanjih organizacij. Vsebinsko so tu pomembne tri vrste povezav, in sicer:

1. *Povezava z informacijskim sistemom ZRSZ.* ZRSZ deluje kot organ v sestavi Ministrstva za delo, družino in socialne zadeve in je eden izmed nosilcev in izvajalcev aktivne politike zaposlovanja v Republiki Sloveniji. Elektronsko povezovanje z informacijskim sistemom ZRSZ daje celovit vpogled na trg dela, hkrati pa omogoča učinkovitejše izvajanje operativnih aktivnosti, kot so npr. prijava potrebe po delavcu, prijava in odjava delavca, prijava družinskih članov idr., za kar pa je treba informacijski sistem ZRSZ ustrezno posodobiti in nadgraditi.
2. *Povezava z informacijskim sistemom skladov dela.* Skladi dela so ustanove za izvajanje ukrepov aktivne politike zaposlovanja v Republiki Sloveniji, ki so ustanovljeni za območje ene ali več občin, pri eni



Slika 4: **Vpetost informacijskega sistema za upravljanje človeških virov**

ali več gospodarskih družbah. Ustanovljajo jih lahko gospodarske družbe, vlada Republike Slovenije, občine, gospodarske, obrtne in druge zbornice in združenja in sindikati in so pravne osebe zasebnega prava. Organizacije z integracijo sistema za upravljanje človeških virov in informacijskim sistemom skladov dela olajšajo razreševanje vsaj dveh problemov. Po eni strani laže, predvsem pa manj boleče, zmanjšujejo število zaposlenih in odpravljajo problem potencialnih ali dejanskih presežnih delavcev, na drugi strani pa imajo organizacije dovolj časa za kadrovsko prestrukturiranje in sanacijo, pridobivajo si ustrezno kvalificirano delovno silo, časovna finančna razbremenitev pri odpuščanju delavcev je porazdeljena, izboljšuje se likvidnostni položaj organizacije ipd. Obstajajo tudi prednosti za državo, saj se prek informacijskega sistema skladov dela in ZRSZ optimalno uporabljajo ukrepi aktivne politike zaposlovanja, ohranja se socialni mir, trajno se povečuje bruto družbeni proizvod, preprečuje se rast brezposelnosti in dolgoročno se manjša poraba sredstev iz proračunske postavke zavarovanja za primer brezposelnosti.

3. *Povezovanje s potencialnimi kandidati za zaposlitev.* Potencialni kandidati lahko na spletnih straneh organizacije iščejo priložnosti za novo zaposlitev. Spletne strani se sestavljajo dinamično na podlagi podatkov iz sistemizacije delovnih mest. Podobno deluje tudi obrazec za zbiranje podatkov o kandidatu (osebni podatki, podatki o izobrazbi, kvalifikacijah, referencah itn.), ki se shranijo direktno v sistem za upravljanje človeških virov. Kandidat lahko spremlja status svoje prijave, saj ob prijavi dobi uporabniško ime in geslo.

Elektronsko poslovanje omogoča bolj preprosto in učinkovito upravljanje človeških virov, katerega cilj je optimalno število primerno usposobljenih delavcev, ki so nujno potrebni za realizacijo poslovnih ciljev organizacije.

5 Literatura in viri

[1] Kaplan, Robert, Norton, David: Uravnoteženi sistem kazalnikov. Ljubljana: Gospodarski vestnik, 2000. 343 str.

[2] Kovačič, Andrej: Informatizacija poslovanja. Ljubljana: Ekonomska fakulteta, 1998. 214 str.

[3] Lončarič Marjan, Antauer, Igor, Ribič, Miroslav: Skladi dela v Republiki Sloveniji: sistemsko dokumentacijsko gradivo. Ljubljana: Zavod Republike Slovenije za zaposlovanje, 2000. 357 str.

[4] Lipičnik, Bogdan: Človeški viri in ravnanje z njimi. Ljubljana: Ekonomska fakulteta, 1996. 326 str.

[5] Možina, Stane: Osnove vodenja. Ljubljana: Ekonomska fakulteta, 1994. 287 str.

[6] Shuler, Randall, Beutell, Nicholas, Youngblood, Stuart: Effective Personnel Management. St.Paul: West Pub.Co., 1989. 680 str.

[7] Zakon o zaposlovanju in zavarovanju za primer brezposelnosti iz leta 1991 (Uradni list Republike Slovenije, v nadaljevanju Ur. l. RS, št. 5/91, 12/92, 71/93, 38/94 , 69/98 in 67/02).

Mag. Miroslav Ribič je po končani Srednji šoli za računalništvo nadaljeval izobraževanje na Ekonomski fakulteti v Ljubljani, kjer se je usmeril v študij informatike. Med študijem se je ukvarjal tudi z izgradnjo informacijskih sistemov, ki podpirajo upravljanje s človeškimi viri, aktivno pa je sodeloval pri informatizaciji skladov dela. Dodiplomski študij je sklenil z delom Informacijski sistem spremljanja in usmerjanja presežnih delavcev v RS. Kot imetnik Microsoftovih licenc MCP in MCSD se aktivno ukvarja s proučevanjem internetnih tehnologij. S tega področja je tudi uspešno ubranil magistrsko delo Implementacija elektronskega poslovanja med podjetji. Trenutno je zaposlen v podjetju IDS Scheer, kjer sodeluje pri uvajanju rešitev podjetja SAP.

Mag. Marjan Lončarič je diplomiral leta 1981 na Visoki šoli za organizacijo dela v Kranju. Naziv magistra znanosti je obranil na Fakulteti za organizacijske vede v Kranju leta 1991 z nalogo Dinamični model spremljanja in usmerjanja razvoja kadrov s posebnim ozirom na presežke zaposlenih v lesni industriji. Na FOV je bil v obdobju od 1985 do 1990 tudi redno zaposlen kot samostojni svetovalec in predstojnik konzultantskega centra, kar ga je v nadaljevanju vodilo v ustanovitev lastnega razvojnega, raziskovalnega in svetovalnega podjetja. Med drugim je kot ekspertni svetovalec vladne projektne skupine snoval in udejanjal projekt Skladi dela v Republiki Sloveniji. Sedanje glavno področje njegovega dela je razvoj in upravljanje z organizacijskimi in informacijski sistemi, še zlasti na področju upravljanja s človeškimi viri.

Dr. Andrej Kovačič je v zadnjih desetih letih delal kot projektant, razvijalec in svetovalec pri projektih strateške prenove in informatizacije poslovanja. Je izredni profesor s področja poslovne informatike na Ekonomski fakulteti in Fakulteti za upravo ter predstojnik Inštituta za poslovno informatiko pri EF v Ljubljani. Več let je bil predsednik programskega odbora Dnevov slovenske informatike v Portorožu, je član izvršnega odbora Slovenskega društva INFORMATIKA, odgovorni urednik revije Uporabna informatika, svetovalec in veščak s področja vodenja in upravljanja podjetij (PHARE, Zveza ekonomistov) in pooblaščeni revizor informacijskih sistemov.

Aktivno arhiviranje

Stane Ravnik
Pris, d. o. o. Domžale

Izvleček

Eno od tveganj v procesu izvajanja informacijskega sistema je naraščanje baze podatkov. Upočasni se odzivanje sistema, vzdrževanje baze postane zahtevnejše. Klasično rešujemo problem s povečevanjem računalniških kapacitet, kar je drago in s čemer pogosto ne rešimo problema. Druga možnost je aktivno arhiviranje, ko neaktualne podatke umaknemo iz baze podatkov in jih zapišemo v arhivske datoteke, tako da do njih še vedno lahko dostopamo preprosto. V prispevku predlagamo metodologijo in orodja za aktivno arhiviranje.

Abstract

Active Archiving

Database growth is one of the risks for effective information system use. The performance of applications deteriorates and response times increase. A classical solution for database growth is to upgrade hardware and storage capacity. This is a very expensive solution and the database still continues to grow. We propose the methodology and tools for active archiving. The solution keeps performance critical data optimized and preserves older data for potential reuse.

1 Naraščanje baze podatkov – tveganje učinkovitosti informacijskega sistema

1.1 Oris problema

Tveganje pri razvijanju računalniške rešitve je možnost, da uporabnostna računalniška rešitev ne bo skladna s pričakovanji končnih uporabnikov, da ne bo pravočasno na voljo oziroma da stroški, ki jih bo povzročila, ne bodo skladni z načrtovanimi (Turk I., Pojmovnik uporabniške informatike, Ljubljana, Slovenski inštitut za revizijo, 2002).

Delovanje informacijskega sistema je združeno s tveganjem, da računalniške rešitve ne bodo skladne s pričakovanji končnih uporabnikov. Celotno tveganje je odvisno od niza sistemskih in tehničnih tveganj v aktivnostih načrtovanja, razvoja in vzdrževanja sistema. Če ne nadzorujemo dejavnikov, ki povzročajo ta tveganja, bo ob uresničitvi tveganja naš projekt graditve sistema neuspešen ali pa bo delovanje operativnega sistema neučinkovito.

Eno od tveganj v procesu izvajanja informacijskega sistema je naraščanje baze podatkov. V metodologiji prototipne graditve sistema to tveganje uvrščamo v projektno fazo nadzora kakovosti sistema. Z nadzorovanjem kakovosti med drugim dosežemo, da so uporabniške rešitve operativno učinkovite, njihovo učinkovitost pa ocenjujemo po odzivnih časih in varnosti podatkov.

Informacijski sistem postaja s časom vse manj učinkovit zaradi hitrega naraščanja baze podatkov. Upočasni se odzivanje sistema, vzdrževanje baze podatkov pa postane zahtevnejše. Čim obsežnejša je baza podatkov, tem več časa potrebuje sistem za iskanje in obdelavo podatkov, aplikativne rešitve so vse počasnejše. Izkušnje kažejo, da se količina podatkov v enem letu podvoji ali celo potroji. Če ima podjetje podatkovno intenziven informacijski sistem, nenadzorovana rast baze podatkov lahko neposredno vpliva na poslovanje zaradi nezadovoljnih uporabnikov in slabših možnosti izdelave informacij za odločanje.

Svetovalna organizacija Giga Information Group predvideva celoten strošek vodenja baze podatkov 5.000 $ na gigabyte na leto. Ta strošek vključuje: stroške računalniške opreme (strežnik, diski, mreža) – 33 %, stroške strokovnjakov za informacijski sistem – 32 %, stroške operacijskega sistema (licence za bazo podatkov, aplikacije in povezan software) – 17 %, stroške infrastrukture (telefonske povezave, elektrika, prostor) – 18 %.

Problem naraščanja baze podatkov rešujemo na dva načina – z nakupom dodatnih računalniških kapacitet ali z arhiviranjem podatkov in njihovim brisanjem iz operativne baze podatkov. V nadaljevanju opisujemo oba pristopa in se po kritiki prvega načina – nakupa dodatnih kapacitet – osredotočimo na rešitev problema z arhiviranjem.

Arhiviranje podatkov je v okolju relacijske baze podatkov zahtevno opravilo. Običajno kopiramo

izbrane, medsebojno skladne nize podatkov. Paziti moramo, da ne umikamo podatkov, ki so potrebni aplikacijam v operativnem okolju, ali pokvarimo operativne baze podatkov. Rešitev mora obvladovati kompleksne podatkovne relacije in zagotavljati njihovo medsebojno konsistentnost. Kadar je podatkovni model enostaven, je zagotavljanje konsistentnosti relativno enostavno. Vendar večina današnjih transakcijsko intenzivnih aplikacij temelji na kompleksnih podatkovnih modelih. Največkrat je baza podatkov sestavljena iz več sto medsebojno povezanih tabel.

Zaradi teh razlogov se podjetja redko odločajo za umik starih podatkov iz produkcije, saj je arhiviranje povezano z nevarnostjo izgube podatkov. Tudi kadar so podatki varno arhivirani, sta njihovi uporaba in ponovna vzpostavitev problematični.

1.2 Klasični način reševanja problema – nakup dodatnih kapacitet

Problem rasti baze podatkov rešujemo na klasičen način s povečevanjem računalniških kapacitet in spominskih enot. Hitrejši, močnejši procesorji pospešijo dostop do podatkov, na dodatne spominske kapacitete zapisujemo vse več podatkov. V bazi podatkov imamo podatke poslovanja tekočega leta in vseh preteklih let. Vendar je ta rešitev slaba zaradi naslednjih glavnih razlogov.

1.2.1 Povečevanje stroškov

Nakup dodatnih računalniških kapacitet lahko pomeni začasno rešitev, vendar moramo z rastjo baze te nakupe ponavljati. Rezultat je ta, da porabimo veliko denarja za računalniško opremo in dodatne licence operacijskega sistema ter orodij. V preteklosti so strokovnjaki za informacijski sistem predvidevali linearno povečevanje baze podatkov pri izračunu potrebnih računalniških kapacitet za normalno delovanje informacijskega sistema, z nastopom intenzivne obravnave podatkov, elektronskega poslovanja in interneta pa so zahteve postale večje kot kdajkoli.

1.2.2 Povečevanje zahtevnosti vzdrževanja podatkov

Z vzdrževanjem baze podatkov lahko dosežemo boljšo učinkovitost sistema, in sicer tako, da bazo podatkov pogosteje reorganiziramo, dodajamo indekse, bazo razdelimo na delne baze, jo denormaliziramo ipd. Vendar z nadaljnjo rastjo postaja to delo vse manj učinkovito, saj vse manj vpliva na učinkovitost informacijskega sistema.

1.2.3 Neučinkovitost ukrepanja ob prekinitvah delovanja sistema

Ob prekinitvah delovanja sistema zaradi kakršnihkoli razlogov je treba najprej vzpostaviti ponovno delovanje baze podatkov. Ker je ta obsežna, lahko takšna vzpostavitev sistema poteka več ur ali celo več dni.

1.3 Cilj prispevka

Na podlagi opisanih problemov lahko ugotovimo, da klasični način obvladovanja rasti baze podatkov ni učinkovit, rešitev pa je kratkoročna. Potrebujemo metodologijo za nadzorovano arhiviranje relacijske baze podatkov, s katero bomo rešili problem tveganja informacijskega sistema glede naraščanja baze podatkov. Za okolje relacijske baze smo se odločili, ker je informacijski sistem z relacijskim krmilnim sistemom v naših podjetjih najpogostejši in ker imamo z njegovim upravljanjem dolgoletne praktične izkušnje. Pričujoči prispevek je zamišljen kot uvod v daljši raziskovalni ciklus, ki ga sestavljata splet različnih raziskovalnih nalog v okviru skupnega projekta izdelave in uporabe programskih rešitev za arhiviranje baze podatkov. V njem nameravamo razviti ustrezno metodologijo in računalniške rešitve arhiviranja podatkov.

Razviti želimo metodologijo za arhiviranje podatkov v okolju relacijskega krmilnega sistema, ki bo ustrezala naslednjim lastnostim:

- zmanjšati mora stroške investicij v računalniško opremo in licence programske opreme,
- poenostaviti mora vzdrževanje baze podatkov,
- ukrepanje ob prekinitvah delovanja sistema mora biti učinkovito,
- omogočati mora arhiviranje podatkov na cenenih medijih,
- zmanjšati mora čas, potreben za reorganizacijo baze podatkov,
- omogočati mora neposreden dostop do arhivskih datotek na takšen način, da uporabniki ne zaznajo razlike med aktivnimi in arhiviranimi podatki.

2 Opredelitev aktivnega arhiviranja

Po splošni definiciji je arhiviranje urejeno in varno shranjevanje datotek, praviloma za določeno obdobje. V procesu arhiviranja kreiramo arhiv podatkov, ki je statična, neodvisna kopija povezanega niza datotek

za vmesno ali trajno hranjenje, potrebna za računovodsko poročanje, revidiranje, izpolnjevanje državnih zahtev itn.; po arhiviranju se lahko izvirne datoteke shranijo ali brišejo (Turk I., Pojmovnik uporabniške informatike, Ljubljana, Slovenski inštitut za revizijo, 2002).

Splošno definicijo razširimo oziroma podrobneje določimo s tremi zahtevami:

- Selektivnost. V procesu arhiviranja obravnavamo natančno določeno množico podatkov, ki jo določimo glede na pogostost uporabe in količino zapisov v bazi podatkov. To pomeni selektivnost v transakcijah arhiviranja, brisanja, pregledovanja in restavriranja podatkov.
- Enostavnost. Način zapisa podatkov v arhivu mora biti takšen, da do podatkov dostopamo na način, ki je skladen z relacijskim modelom podatkov, to je po istih pravilih, kot dostopamo v bazo podatkov.
- Dinamičnost. Informacijski sistem je dinamičen, med drugimi njegovimi elementi se spreminja tudi model podatkov in s tem oblika baze podatkov. Zato zahtevamo dinamičnost arhiva, kar pomeni omogočanje dostopa do podatkov takrat, ko so povezave med podatki v arhivu drugačne od tistih v trenutnem operativnem informacijskem sistemu.

Tako razširjeno definicijo arhiviranja imenujemo aktivno arhiviranje.

Aktivno arhiviranje je skrčenje baze podatkov z umikom podatkov, ki jih ne potrebujemo pri vsakodnevnih transakcijah, in njihovim hranjenjem v aktivnih arhivskih datotekah. Operativna baza podatkov produkcijskega okolja vsebuje samo aktualne podatke, ko potrebujemo arhivirane, jih dobimo neposredno iz arhivskih datotek.

Proženje arhiviranja brisanja in restavriranja izvedemo na ravni aplikacijskih rešitev in ne le v obliki administrativnih opravil. V informacijski sistem ga integriramo z uporabniškimi aplikacijami za dostopanje končnih uporabnikov do arhivskih podatkov. Ker so arhivirani podatki zapisani v obliki enostavnih datotek in ne v bazi podatkov, zgradimo dostop aplikacij do podatkov z uporabo vmesnikov ODBC.

Za ilustracijo aktivnega arhiviranja navedimo primer podjetja za vodenje kreditnih kartic. Podjetje ima zaradi narave dela podatkovno intenziven informacijski sistem. Obsežno bazo podatkov lahko ločimo z vidika potrebnosti podatkov v transakcijah. Stalno so potrebni podatki o kupcih – ime, naslov, številka računa, stanje na računu ipd. Ko podjetje kupcem izdaja račune, nastanejo podatki o računih, ki pa jih po izdaji in finančnoračunovodski obravnavi ne potrebujemo več v dnevnih transakcijah. Zato jih zapišemo v arhiv in brišemo iz operativne baze podatkov, kjer bi njihova velika količina povzročala upočasnjeno delovanje informacijskega sistema.

Izdani račun potrebujemo le v primeru reklamacij ali revidiranja. Proces aktivnega arhiviranja omogoča hiter dostop do zahtevanega računa z običajnimi aplikativnimi rešitvami. Podatke o računu lahko selektivno restavriramo, kar pomeni, da prepišemo v operativno okolje samo podatke zahtevanega računa in ne npr. vseh arhiviranih računov. Dostop do računa je možen tudi v primeru, če se je med izdajo računa in med zahtevo po njegovem restavriranju spremenila oblika modela podatkov, npr. struktura atributov računa.

2.1 Arhiviranje in brisanje podatkov

Podatke arhiviramo in brišemo selektivno, se pravi samo tiste, ki ustrezajo vnaprej določenim kriterijem, definiranim v načrtu arhiviranja. Za ta namen definiramo enostavne ali kompleksne pogoje, skladne z modelom podatkov. Z enostavnostjo dostopa smo v definiciji aktivnega arhiviranja zahtevali dostop do arhiviranih podatkov na enak način kot do tistih v operativni bazi podatkov. Zato morajo arhivirani podatki v okolju relacijskega krmilnega sistema zadržati vse kompleksne povezave relacijskega modela podatkov.

Za varen in učinkovit umik podatkov iz relacijske baze sistem aktivnega arhiviranja zagotavlja naslednje:

- obravnavanje kompleksnih relacijskih povezav z upoštevanjem referenčne integritete,
- obravnavanje metapodatkov, ki zagotavlja sledljivost oblike podatkov in njihove relacijske povezanosti,
- obravnavanje podatkov po delnih in združenih atributih ter datumsko povezovanje,
- prenos relacij iz podrejenih na različne nadrejene relacije ipd.

Transakciji arhiviranja in brisanja podatkov nista medsebojno povezani, kar pomeni, da arhiviranje podatkov ne pomeni nujno tudi njihovega hkratnega brisanja iz operativne baze podatkov. Aktivno arhiviranje omogoča naslednje vrste brisanja podatkov:

- Takojšne brisanje po arhiviranju. Odločimo se lahko za takojšne brisanje podatkov po arhiviranju. Za dodatno varnost moramo imeti na razpolago pregled podatkov, s katerim se pred brisanjem prepričamo, da brišemo prave podatke.
- Brisanje z zamikom. Brisanje podatkov izvajamo potem, ko je bilo arhiviranje že uspešno izvedeno. Brisanje podatkov z zamikom omogoča preverjanje posameznih zapisov podatkov pred brisanjem, lahko pa proces brisanja vključimo tudi v svoje vzdrževalne procedure.
- Selektivno brisanje. S selektivnim brisanjem brišemo vse ali samo dele arhiviranih podatkov (npr. arhiviramo vsa naročila in plačila kupcev, neaktivnih zadnjih pet let, zadržati pa želimo identifikacijske podatke o kupcih, torej njihove nazive in naslove). Arhiv kreiramo z vsemi podatki vsakega kupca, selektivno brisanje pa briše le naročila in plačila.

2.2 Dostop do podatkov in restavriranje

Arhiviranje je popolno oziroma smiselno šele takrat, ko omogoča tudi branje in restavriranje podatkov, ko je to potrebno. Do arhiviranih podatkov dostopamo iz različnih razlogov: pri finančnoračunovodskem pregledu starih finančnih podatkov, usklajevanju podatkov s partnerji, pri zahtevah zunanjih institucij za vpogled v podatke ali pri ponovni vzpostavitvi pokvarjene baze podatkov. Podatke pregledujemo ali restavriramo skladno z relacijskimi pravili, določenimi v modelu podatkov. Delo izvajamo z vgrajenimi uporabniškimi rešitvami, v katerih določimo kriterije in tako vzpostavimo samo nekaj zapisov iz arhiva, ki ustrezajo tem kriterijem. Kriteriji restavriranja so lahko drugačni od tistih, ki smo jih uporabili pri arhiviranju, npr. naročila in plačila kupcev so arhivirana na podlagi meseca obračuna, po poslovnem pravilu, da je zadnja transakcija starejša od dveh let. Čez sedem let želimo restavrirati podatke kupcev iz Ljubljane. Aktivno arhiviranje omogoči hitro identifikacijo in restavriranje transakcijskih podatkov samo za partnerje iz Ljubljane. V tem primeru smo podatke arhivirali na podlagi kriterija datuma in restavrirali z uporabo kriterija mesta partnerja.

Podatke restavriramo v produkcijsko bazo podatkov ali bolj pogosto v posebno bazo podatkov, kjer jih pregledujemo in izpisujemo.

Glavne značilnosti dostopa do podatkov v procesu aktivnega arhiviranja so:

- možnost selektivne vzpostavitve arhiviranih podatkov,
- možnost pregleda arhiviranih podatkov brez njihove ponovne vzpostavitve v operativno okolje,
- hiter dostop do arhiviranih podatkov.

3 Metodologija izvedbe aktivnega arhiviranja

V nadaljevanju bomo izdelali zasnovo metodologije izvedbe aktivnega arhiviranja, kar je glavni cilj prispevka. Uporabili bomo zahteve, ki smo jih definirali v prvem delu, in naše izkušnje pri projektiranju informacijskih sistemov.

Projekt definiramo v dveh delih: z načrtovanjem in prototipno izvedbo aktivnega arhiviranja. V načrtu arhiviranja izhajamo iz ciljev, ki jih definiramo na podlagi kritičnih dejavnikov, povezanih z obsežnostjo in naraščanjem operativne baze podatkov. Namen dejavnikov je opredelitev učinkovitosti informacijskega sistema in vzpostavitev načina arhiviranja, ki bo k temu pripomogel. Ob analizi kritičnih dejavnikov uspešnosti razmišljamo o obsegu posameznih relacijskih tabel, odzivnih časih, potrebnosti starih podatkov v tekočih aplikacijah ipd. Z načrtom aktivnega arhiviranja vzpostavimo splošno arhitekturo arhiviranja, ki jo realiziramo med poznejšo graditvijo.

Sistem aktivnega arhiviranja zgradimo na podlagi načrta na prototipni način, ki zagotavlja postopno graditev arhivskih datotek in uporabniških rešitev s tekočim preverjanjem ustreznosti zgrajenega sistema.

3.1 Načrtovanje aktivnega arhiviranja

Sistem arhiviranja začnemo graditi z njegovim načrtovanjem. Najprej moramo spoznati strukturo baze podatkov in delovanje informacijskega sistema, šele potem lahko določimo cilje, probleme in kritične dejavnike, povezane z obsegom in naraščanjem baze podatkov. Cilji načrta aktivnega arhiviranja so:

- opisati informacijski sistem z vidika modela podatkov in z obsežnostjo pripadajočih relacijskih tabel,
- opredeliti cilje in kritične dejavnike uspešnosti, povezane z obsegom in naraščanjem baze podatkov,
- opredeliti sistem aktivnega arhiviranja glede na cilje in kritične dejavnike uspešnosti,
- opredeliti prednostna področja v razvoju sistema arhiviranja.

3.1.1 Opis informacijskega sistema

Z opisom informacijskega sistema ponazorimo njegovo delovanje in strukturo ter opredelimo poslovne funkcije in entitete. Na ta način ugotovimo, kje se določene funkcije izvajajo, katere entitete pri tem nastopajo in kakšen je obseg posameznih, entitetam pripadajočih relacijskih tabel. Rezultate zapišemo v obliki strateškega modela podatkov. Dopolnimo ga s podatki o obsegu in predvidenem naraščanju pripadajočih relacijskih tabel.

3.1.2 Analiza problemov in ciljev informacijskega sistema

Analiza problemov in ciljev je neposredno povezana s problematiko delovanja informacijskega sistema. Analizo izvedemo po posameznih poslovnih funkcijah. Med analiziranjem ugotovimo tiste probleme, ki so v zvezi z obsegom baze podatkov. Za probleme, ki so rešljivi z umikom podatkov in s tem skrčenjem baze podatkov, določimo vsebinske zahteve sistema arhiviranja. Rezultat te aktivnosti je prikaz odnosov med problemi in zahtevami aktivnem arhiviranju za rešitev.

Pri definiranju ciljev uporabimo tehniko analize kritičnih dejavnikov uspešnosti informacijskega sistema. Z njimi opredelimo najpomembnejša področja za uspeh informacijskega in posledično poslovnega sistema. Zgraditi moramo takšen sistem arhiviranja, ki bo zagotavljal učinkovitost informacijskega sistema predvsem na teh področjih. Kritične dejavnike uspešnosti ugotovimo med pogovori z nosilci posameznih aplikativnih področij informacijskega sistema in z nosilci upravljanja ključnih poslovnih funkcij.

3.1.3 Opredelitev načrta aktivnega arhiviranja

Na podlagi ciljev izdelamo načrt arhiviranja v obliki prikaza arhivskih datotek in pripadajočih entitet sistema. Pri opisu relacij med arhivskimi datotekami za zagotovitev referenčne integritete izhajamo iz tovrstnih relacij med pripadajočimi entitetami informacijskega sistema.

V načrtu opredelimo tudi politiko arhiviranja, kjer določimo načine brisanja podatkov ob arhiviranju in kriterije selekcije ob arhiviranju in dostopanju do arhiva. Prikažemo odnose med arhivskimi datotekami in računalniškimi rešitvami informacijskega sistema ter jih opišemo z vidika načina pregledovanja in restavriranja.

3.1.4 Opredelitev prednostnih nalog v graditvi sistema arhiviranja

Načrt arhiviranja ugotavlja določena kritična in problematična področja informacijskega sistema z vidika obsežnosti baze podatkov. Projekte za graditev sistema arhiviranja moramo obravnavati z vidika donosnosti naložbe in potrebnih vložkov. Na tej podlagi določimo prednostna področja razvoja sistema arhiviranja in terminski načrt graditve. Prednostna področja opredelimo po merilih nujnosti poslovnih funkcij in problemov informacijskega sistema ter subjektivnih merilih vodstva podjetja.

3.2 Prototipna graditev sistema aktivnega arhiviranja

Za učinkovito in varno izdelavo sistema aktivnega arhiviranja je posebno pomembna uporaba prototipnega načina graditve, saj s tem tekoče preverjamo dobljene rezultate v arhivskih datotekah in jih primerjamo z želenimi rezultati. Na ta način se tudi izognemo nevarnosti izgube podatkov po njihovem brisanju iz operativne baze podatkov.

Prototipni način pomeni izdelavo operativno funkcionalnega sistema, tako da lahko opazujemo in preizkušamo nastajajočo rešitev pred začetkom njegovega rednega obratovanja. Prototip je sistem v fazi razvoja, ki nam pomaga razumeti in testirati rešitev. Poudariti je treba, da prototip ni le skupina računalniških programov, temveč organizacijska enota vseh medsebojno povezanih prvin v razvoju, torej tudi pisnih rezultatov načrtovanja in graditve sistema.

Prototip uporabljamo za vrednotenje idej in prikazovanje zasnove sistema. Ko je vrednotenje končano, prototip prevedemo v delujoč sistem. Razvoj temelji na vrednotenju ustreznosti sistema glede na zahteve načrta arhiviranja, v fazi uvedbe prototipa pa ugotavljamo učinkovitost sistema, ki opredeljuje njegovo ustreznost glede na realno okolje. Prednost ločevanja ustreznosti in učinkovitosti sistema je v tem, da ga lahko razvijamo ne glede na realno okolje, po drugi strani pa lahko njegovo operativno učinkovitost neodvisno optimiziramo.

3.2.1 Razvoj prototipa

Prototip razvijamo po naslednjih postopkih:

- razvoj transakcij za izdelavo arhivskih datotek,
- razvoj transakcij za uporabo in restavriranje arhiva,
- prikaz rezultatov razvoja prototipa kot podlage za njegov nadaljnji razvoj.

V fazi razvoja prototipa podrobneje razčlenjujemo rezultate, ki smo jih navedli v fazi načrtovanja arhiviranja.

Izdelava transakcij pomeni prevedbo funkcij v instrukcije računalniškega orodja, s katerim bomo izvajali arhiviranje. Funkcije prevajamo v transakcije v okviru priprav na sestanke hevristične analize ali pa neposredno na sestankih. Preverjamo ustreznost arhivskih datotek in transakcij.

Problemi v zvezi z ustreznostjo sistema arhiviranja, s katerimi se ukvarjamo na sestankih, so predvsem:

- skladnost transakcij arhiviranja, brisanja in restavriranja s funkcijami, določenimi v načrtu arhiviranja,
- skladnost arhivskih datotek s tabelami baze podatkov,
- skladnost referenčne integritete arhivskih datotek z referenčno integriteto baze podatkov,
- skladnost zapisov v arhivskih datotekah in bazi podatkov z zahtevami selektivnih kriterijev.

Prototip je končan takrat, ko je sistem arhivskih datotek stabiliziran, informacijski sistem pa deluje z uporabo arhivskih datotek. Sistem arhivskih datotek je stabiliziran, ko lahko uvedemo naključne nove funkcije informacijskega sistema, povezane s podatki arhiva, ne da bi bilo treba spremeniti obliko arhiva.

3.2.2 Uvedba prototipa

Po končanem razvoju prototipa in potem, ko ugotovimo, da je le-ta stabiliziran, ga uvedemo v operativno tehnološko okolje. Preveriti moramo tudi njegovo učinkovitost v povezavi z realno količino podatkov.

V opredeljevanju tehnološkega okolja delujočega sistema definiramo tehnologijo, na podlagi katere uvedemo sistem, ki smo ga razvili na prototipni način. Izberemo računalniško opremo, pri čemer ocenjujemo ustroj zunanjih pomnilnikov. Le-ta je odvisen od hitrosti odziva uporabe arhivskih datotek, cene in primernosti shranjevanja.

Učinkovitost delujočega sistema merimo glede na zadovoljevanje uporabniških potreb, pri katerih so odzivni časi operativne baze najpomembnejši element. Prototip aktivnega arhiviranja je v celoti uveden, ko smo v operativnem tehnološkem okolju dosegli optimalno učinkovitost celotnega operativnega informacijskega sistema.

3.2.3 Nadzor kakovosti

V tem okviru pregledujemo nove zahteve uporabnikov in na njihovi podlagi dograjujemo sistem arhiviranja. Zahteve po spremembah sistema se najpogosteje pojavljajo zaradi spremenjenega modela podatkov. Če se to zgodi, ponovimo funkcije razvoja in uvedbe prototipa. V primeru zahtev po boljši operativni učinkovitosti ponovimo samo fazo uvedbe prototipa. Najbolj zahtevne so spremembe zaradi novih poslovnih pravil ali kritičnih dejavnikov uspešnosti. V tem primeru moramo izdelati nov projekt z vsemi fazami prototipnega razvoja.

Preverimo doseganje ciljev, ki je definirano v načrtu arhiviranja. Ugotovimo, ali smo odpravili ovire kritičnih dejavnikov uspešnosti in ali je operativni informacijski sistem z uvedenim sistemom arhiviranja učinkovitejši od prejšnjega.

Z nadzorovanjem kakovosti dosežemo, da so rešitve arhiviranja operativno učinkovite. Njihovo učinkovitost ocenjujemo po odzivnih časih informacijskega sistema in varnosti baze podatkov in arhivskih datotek.

3.3 Računalniška orodja in rešitve

Za izdelavo načrta, se pravi oblikovanje modela podatkov z arhivskimi datotekami, uporabimo orodje Oracle Designer, ki nam omogoča zapisovanje in pregled nad vsemi metapodatki o bazi podatkov. Poleg entitet določimo tudi njihovo medsebojno referenčno integriteto, obseg podatkov in njihovo predvideno naraščanje. S povratnim inženiringom baze podatkov izdelamo model podatkov. V tega dodamo arhivske datoteke, ki jih povežemo z osnovnimi entitetami. Tudi med arhivskimi podatki določimo referenčno integriteto.

Za izdelavo sistema arhiviranja lahko razvijemo svojo aplikativno rešitev, ki ustreza zahtevam načrta arhiviranja. Takšna rešitev bi bila zelo kompleksna, ker moramo poleg arhivskih datotek obravnavati tudi njihove metapodatke zaradi sledenja spremembe baze podatkov v prihodnosti, arhivske datoteke pa je treba opremiti še z referenčno integriteto.

Zato za izdelavo arhivskih datotek uporabimo orodje Archive for Servers podjetja Princeton Softech. V orodju določimo zahteve iz načrta arhiviranja, arhivske datoteke z metapodatki in referenčnimi integritetami se zapišejo v interni obliki orodja. Za uporabo orodja Archive for Servers izdelamo uporabniško prijazno računalniško rešitev njegove vključitve v osnovni meni informacijskega sistema, s katero prožimo arhiviranje. Predvsem pa je treba izdelati obravnavo arhivskih datotek v aplikacijah ali pri

restavriranju. Ker so arhivske datoteke zapisane v interni obliki orodja, lahko do njih dostopamo z ODBC vmesniki sistema za krmiljenje baze podatkov. Vmesnike uporabimo neposredno v aplikacijah ali pa izdelamo pomožne tabele, v katere polnimo podatke iz arhiva in jih nato obravnavamo s standardnimi aplikacijami.

Literatura

Karolak, D. W.: Software engineering Risk Management. Los Alamitos, IEEE Computer Society Press, 1996

Oracle Corporation: Oracle Designer (Dokumentacija orodja). USA, Oracle Corporation, 2000

Princeton Softech: Archiving Complex Relational Databases (White Paper). Princeton, Princeton Softech, 2002

Ravnik, S.: Podatkovno zasnovani informacijski sistemi in njihova graditev. Maribor, VEKŠ, 1989

Turk, I., Pojmovnik uporabniške informatike, Ljubljana, Slovenski inštitut za revizijo, 2002

Vonk, R.: Prototyping. Englewood Cliffs, Prentice Hall, 1990

Mag. Stane Ravnik ima prek 20 let delovnih izkušenj na področju načrtovanja in graditve informacijskih sistemov – najprej v podjetju Produktivnost Ljubljana, od leta 1987 pa v PRIS, d. o. o. Osebne reference so načrtovalni in izvedbeni projekti v večjih podjetjih – Tosama Domžale, LTH Škofja Loka, Merkur Kranj, Semenarna Ljubljana, Croatia Osiguranje Zagreb, Vele Domžale. Na teoretičnem področju se je ukvarjal z razvojem metodologije graditve informacijskega sistema in je zagovornik podatkovno vodenega prototipnega pristopa. Raziskuje zakonitosti tveganja pri projektih informacijskih sistemov.

Udeleženci posvetovanja Dnevi slovenske informatike 2003,
ki je bilo od 16. do 18. aprila 2003 v Portorožu,
smo z namenom, da bi ocenili vlogo, pomen in cilje posvetovanja ter priporočili usmeritve
za prihodnja posvetovanja, sprejeli naslednjo

DEKLARACIJO 10. JUBILEJNEGA POSVETOVANJA DNEVI SLOVENSKE INFORMATIKE 2003,

s katero želimo sporočiti javnosti svoje videnje o poteku, dosežkih in pomenu dogodka in podati usmeritve za delovanje Slovenskega društva INFORMATIKA v času do prihodnjega posvetovanja.

1. Udeleženci ugotavljamo, da je bila vodilna misel – slovenska informatika v družbi najrazvitejših za tretje tisočletje – izbrana primerno, saj so najvišji predstavniki slovenske države podpisovali pridružitev Slovenije Evropski uniji prav v času, ko se je odvijalo posvetovanje. Vsebina, kakovost in število prispevkov na posvetovanju so dokaz, da slovenski informatiki vstopajo v nove evropske povezave kot enakovredna skupina, ki bo na strokovnem in znanstvenem področju lahko enakopravno sodelovala s strokovnjaki vseh držav Evropske unije. Predsednik Republike Slovenije dr. Janez Drnovšek je v svojem pozdravnem nagovoru poudaril pomen informatike za še intenzivnejši razvoj slovenskega gospodarstva in slovenske družbe ter vlogo države pri nadaljnjem razvoju uporabe sredstev informacijske tehnologije. Mednarodni ugled posvetovanja je potrdil tudi predsednik IFIP dr. Klaus Brunnstein, ki je v pozdravnem nagovoru izrazil prepričanje, da je vloga slovenskih informatikov pomembna za razvoj informacijske družbe v regiji, katere del je Slovenija.
2. Ob širjenju vsebin in področij obravnave je bil izbran posrečen kompromis med številom obravnavanih referatov in sekcij, v katerih so bili prispevki predstavljeni. Manj sekcij je pripomoglo k temu, da so udeleženci laže obiskali zanje zanimiva predavanja. Posvetovanje je ob tem ohranilo tudi druge načine predstavitev; aktualne teme so bile širše obravnavane na okroglih mizah, delavnice pa so omogočile vpogled v uporabo praktičnih rešitev. Trajna sled posvetovanja je zbornik, v katerem so objavljeni vsi prispevki in poudariti moramo, da so v posebnem poglavju zbornika objavljeni tudi referati dodiplomskih študentov. Prispevki tujih strokovnjakov, ki so se radi odzvali na vabilo, dokazujejo tudi mednarodni ugled posvetovanja, ki je sicer namenjeno predvsem domači publiki.
3. Udeleženci izražamo prepričanje, da lahko informatika kot znanost in stroka, ki je postala neločljiv del skoraj vseh drugih področij, pripomore k še večjemu blagostanju državljanov, kar pričakujemo od članstva v Evropski uniji in obenem pomaga ohranjati kulturno samobitnost. Posebna razsežnost posvetovanja je, da s svojo vsebino in možnostjo spremljanja dosežkov in rešitev v informatiki ter z možnostjo primerjave domačih dosežkov s tujimi širi pogled strokovnjakov. Od njih se pričakuje ne le poznavanje lastnega področja, temveč tudi to, da zmorejo realno oceniti predstavljene prispevke. To pa poleg poglobljenega znanja na določenem področju omogoča tudi širina znanja in strokovna odličnost, kar oboje intenzivno podpira Slovensko društvo INFORMATIKA. Obenem se tudi zaveda, da se odličnost začenja že v šoli in zato bo posvetovanje odprto najboljšim študentom tudi v prihodnje.
4. Udeleženci posvetovanja z zadovoljstvom ugotavljamo, da so tudi Dnevi slovenske informatike 2003 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 srečamo in spoznamo med seboj, da izmenjamo mnenja, izkušnje, informacije in spoznanja ter jih posredujemo javnosti. Razen tega je posvetovanje namenjeno še poslovnemu sodelovanju in tudi neformalnemu druženju. Ocenjujemo, da je bil letošnji program izredno zanimiv in kakovosten. Obsegal je prek sto prispevkov, katerih avtorji so strokovnjaki iz slovenskih podjetij in ustanov. Na posvetovanju so nastopili vabljeni predavatelji iz Slovenije in tujine, obiskalo pa ga je skupaj več kot štiristo udeležencev. Organizirani sta bili dve okrogli mizi, na katerih so sodelovali številni ugledni strokovnjaki, udeleženci pa so z njimi in med seboj lahko izmenjali svoja mnenja o aktualnih vprašanjih. V program posvetovanja smo uvrstili tudi delavnico in študentski forum. Med odmori so si udeleženci lahko ogledali razstavo, na kateri so se predstavljala nekatera podjetja. Organizirani so bili tudi že tradicionalni družabni dogodki, ki so bili tudi tokrat odlično sprejeti.
5. Na otvoritvi smo lahko poslušali tudi izredno zanimivi vabljeni predavanji. Prvi predavatelj dr. Marjan Krisper je predstavil nov pristop k razvoju informacijskih sistemov, agilnost. Gre za uporabo metodologij, ki so »ravno dovolj« opredeljene ter hitro in enostavno prilagodljive. Drugi vabljeni predavatelj Charles Abrams iz ameriškega podjetja Gartner je govoril o integraciji aplikacij, pri čemer bodo spletne storitve igrale pomembno vlogo. V programu posvetovanja so nastopili še vabljeni predavatelji z Japonske, Madžarske, iz Nemčije in Srbije.
6. Otvoritvi posvetovanja je sledila okrogla miza, ki je obravnavala aktualno problematiko slovenske informatike pred vstopom v Evropsko unijo. Okroglo mizo je vodil minister za informacijsko družbo dr. Pavel Gantar, kot panelisti pa so nastopili mag. Marin Silič, Rudi Bric, Matjaž Čadež in Grega Kukec. Govora je bilo predvsem o konkurenčnosti slovenskih podjetij s področja informatike pred vstopom v EU, sposobnosti nastopanja v širšem evropskem prostoru, povezovanju podjetij za doseganje kritične mase, oblikovanju grozdov ter o tem, kaj lahko stori država za spodbujanje razvoja. Dotaknili so se tudi aktualnih tem s področja e-poslovanja javne uprave, kot je na primer e-dohodnina. Pomembno sporočilo te okrogle mize je, da je sodelovanje države s civilno družbo možno in zaželeno v korist obeh.
7. Druga okrogla miza je skušala odgovoriti na dilemo o vplivu informatike na uspešnost poslovanja. Vodil jo je dr. Andrej Kovačič, panelisti pa so bili Rado Faleskini, Igor Kosem, dr. Marjan Krisper in Dušan Zupančič, ki so podali svoje poglede na vnaprej

predstavljene teze. Obdelali so jih z vidika upravljanja, izobraževanja, standardizacije ter vloge države in uvajanja najboljše prakse in celovitih (ERP) rešitev. Udeleženci okrogle mize so zastavljeno problematiko obravnavali z različnih vidikov, vendar so v celoti potrdili izhodiščne teze. Ugotavljajo, da gre vzroke za nezadostni vpliv informatike iskati zlasti v neustrezno in nepravilno usmerjanih projektih, katerih predlagatelji so v večini primerov sicer informatiki, kjer pa so pobude in vpliv poslovnega sistema največkrat obrobni. Vzroke gre iskati tako v pomanjkljivem znanju in izkušnjah ter omejeni poslovni vplivnosti informatikov, predvsem pa slabo izraženem in pomanjkljivem zavedanju lastnikov in menedžerjev o možnostih in priložnostih, ki jih v strateškoposlovnem smislu nudi informatika.

8. Ker prihaja do vse večje povezanosti med informatiko in telekomunikacijami, je bila v program posvetovanja uvrščena tudi delavnica UMTS in mobilna informacijska prihodnost, ki so jo izvedli člani Laboratorija za telekomunikacije Fakultete za elektrotehniko pod vodstvom dr. Janeza Beštra. Udeležencem so predstavili pot razvoja mobilne telefonije od druge k tretji generaciji. Opisane so bile tehnologije in predstavljene storitve, ki jih posamezne generacije prinašajo uporabnikom. Posebej so bile poudarjene informacijske storitve. Predstavljene so bile tri tehnologije brezžičnih omrežij (sodobni mobilni telefoni z visoko ločljivostjo zaslona, dlančniki in tablični osebni računalniki, povezani v brezžično omrežje) in prikazanih je bilo pet storitev z uporabo teh naprav. Na sodobnih mobilnih telefonih si je bilo mogoče ogledati predvajanje video vsebin, na dlančnikih je bilo prikazano predvajanje televizijskega programa (prek interneta) v realnem času in video nadzor prostora, na tabličnih osebnih računalnikih pa je bila prikazana spletna izobraževalna televizija.
9. Glavnino programa so predstavljali prispevki, razvrščeni v tri sekcije (poslovna informatika in elektronsko poslovanje, informacijske tehnologije in internet, informacijska kultura in družba), katerih avtorji so strokovnjaki iz slovenskih podjetij, univerz in uprave. Letos je bilo v Portorožu predstavljenih 113 referatov in o vseh smo udeleženci lahko tudi razpravljali. V sekciji poslovna informatika in elektronsko poslovanje so bili obravnavani strateški vidiki informatizacije, medorganizacijsko povezovanje in integracija poslovnih procesov ter rešitve. Predstavljen je bil vsebinsko zaokrožen sklop prispevkov na temo modeliranja poslovnih procesov in njihove integracije z izvajalnega in aplikacijskega vidika. Avtorji prispevkov so poudarili problematiko modeliranja poslovnih procesov glede na različne namene (strateško načrtovanje informatike, prenova procesov, izvajanje). V nadaljevanju so bili predstavljeni razlogi za povezovanje programskih rešitev v okviru procesov in kaj zavira tovrstno povezovanje. Pomembna je ugotovitev, da je upravljanje poslovnih procesov pogoj za uspešno in fleksibilno povezovanje programskih rešitev. Prikazane so bile tudi posebnosti integracije v virtualni organizaciji.
10. Zavedanje o možnostih, ki jih ponuja informatika v okoliščinah povečane konkurence, se kaže tudi v vse pogostejšem uvajanju upravljanja odnosov s strankami (Customer Relationship Management – CRM) v organizacije. Pomemben vidik tega se izvaja skozi spletno tržno pot, kar se je pokazalo v prispevkih na temo elektronskega poslovanja. Da gre pri elektronskem poslovanju za veliko več kot za izdelavo spletnih aplikacij, je bilo razvidno iz raznolikosti tem. Avtorji so predstavili problematiko upravljanja vsebin, zakonitosti e-trgovanja in standarde za integracijo poslovnih procesov med organizacijami (ebXML). Problematika uvajanja elektronskega poslovanja v zavarovalništvu kaže, da glavne težave niso tehnološke narave, pač pa nerazumevanje elektronskega poslovanja v primerjavi z ostalim finančnim sektorjem, neprimernosti tradicionalne poslovnofunkcijske organiziranosti in prodajnih poti.
11. V sekciji informacijske tehnologije in internet so prispevki obravnavali predvsem informacijsko arhitekturo, pristope in koncepte, sodobne razvojne tehnologije ter zgradbo, izgradnjo in praktične primere portalov. Poleg zgradbe, načina izgradnje in praktičnih predstavitev portalov so imeli udeleženci možnost poslušati o napakah, virtualnih in drugačnih skupinah ter o novih pristopih dela na področju informacijske tehnologije. Poudarek je bil tudi na varovanju, kjer smo se srečali s praktično rešitvijo v velikem podjetju ter nevarnostmi za majhna in srednje velika podjetja. Obravnavane so bile možnosti uporabe televizorja za dostop do interneta, kjer lahko v prihodnje pričakujemo odpiranje novih možnosti.
12. Sekcija informacijska kultura in družba je bila posvečena vprašanjem širšega družbenega konteksta, v katerega se umešča informatika. Prispevki so se povezovali v tematske sklope, ki obsegajo vpliv informatike na socialni razvoj, informacijsko podporo demokraciji in upravljanju, izobraževanje in kulturo na področju informatike ter skrb za slovenski jezik. Vsebinski poudarki so bili tudi na uporabi informacijske tehnologije za pomoč ljudem s posebnimi potrebami ter na uporabnosti spletnih mest. Ugotovili smo tudi, da lahko informatika pripomore k dvigu kakovosti izobraževanja. Nič manj pomembno ni poenotenje strokovnega izrazja informatike.
13. Tudi letos smo na posvetovanju organizirali poseben dogodek za mlade informatike – študentski forum, na katerem so svoje prispevke predstavili dodiplomski študenti informatike z različnih fakultet. Najboljši študentski prispevek je izbrala posebna komisija. S tem delom posvetovanja želijo organizatorji posvetovanja spodbujati zanimanje za strokovno delo, javno nastopanje in predstavljanje svojih dosežkov kot temelj kasnejšega strokovnega uveljavljanja.
14. Na posvetovanju so bila prisotna vsa pomembnejša slovenska podjetja in ustanove s področja informatike, predstavniki mnogih so bili aktivno udeleženi v programu, drugi pa so se posvetovanja udeležili kot poslušalci. Precej podjetij je posvetovanje tudi finančno podprlo, za kar se vsem najlepše zahvaljujemo, saj brez njihove pomoči posvetovanja ne bi bilo mogoče izpeljati.
15. Udeleženci priporočamo prirediteljem posvetovanja, naj zasleduje uveljavljene vsebinske usmeritve tudi v prihodnje. K sodelovanju naj povabi več predstavnikov poslovne sfere, ker bo s tem posvetovanje pridobilo pomembno vlogo tudi pri navezovanju in utrjevanju poslovnih stikov. Še naprej naj navezujejo, ohranjajo in poglabljajo stike s strokovnjaki slovenskega rodu. Kljub temu, da je posvetovanje nacionalno, naj povabijo več tujih strokovnjakov, mednarodnih organizacij in predvsem sorodnih organizacij bližnjih držav, saj s tem krepijo sodelovanje v regiji. Od organizatorjev pričakujemo, da bodo posvetovanje tudi v prihodnje razvijali v smeri najpomembnejšega slovenskega neodvisnega strokovnega dogodka s področja informatike.

Tehnologije znanja pri predmetu informatika

Alenka Krapež, Vladislav Rajkovič

Zavod Republike Slovenije za šolstvo, 92 str., Ljubljana 2003

Po učnem načrtu je v izbirnem delu predmeta informatika predviden tudi sklop tehnologije znanja. Za njegovo izpeljavo je Zavod RS za šolstvo izdal vodnik za učitelje, ki je namenjen tudi vsem, ki želijo obogatiti vzgojno-izobraževalni proces s sodobnimi informacijskimi tehnologijami.

Vodnik predstavlja vsebine in možne oblike poučevanja tehnologij znanja. Razdeljen je na pet delov. Učitelje opremlja s številnimi napotki, priporočili in navodili; seznanja jih tudi z izkušnjami drugih učiteljev. Slike in tabele lepo dopolnjujejo besedilo.

V prvem delu je natančno opisan model poučevanja tehnologij znanja s cilji, predlaganimi oblikami poučevanja in vsebino. Opisani so človekovi miselni procesi pomnjenja in odločanja, različne tehnologije znanja in posebej sistemi za podporo odločanju od ročne preglednice do metod umetne inteligence.

V drugem delu je objavljen učni načrt z vsebino, operativnimi cilji in s predlogom časovne razporeditve ur. Navedeni so standardi znanj, didaktična priporočila in napotki za načine preverjanja in ocenjevanja znanja.

V tretjem delu so opisana številna gradiva, ki jih učitelji lahko uporabijo kot pripomoček pri pouku; med drugim tudi program DEXi in elektronske prosojnice, vse dostopno na spletnem naslovu http://lopes1.fov.uni-mb.si/DEXi. Podrobno je opisan učni primer Poročilo za izbiro maturitetnega predmeta. Dodana so priporočila za izbor odločitvenega problema, primeri dreves kriterijev itn.

V četrtem, zelo zanimivem delu, predstavljajo svoje izkušnje učitelji, ki so v okviru projekta Uvajanje tehnologij znanja v predmet informatike model poskusno vpeljali v pouk. Dodana je analiza preizkusa učnega modela, v kateri avtorji ugotavljajo, da so dijaki vsebino dobro sprejeli in pridobili številna koristna spoznanja; hkrati pa iz nje izhajajo tudi nekateri predlogi za izboljšanje programa DEXi, učnih pripomočkov in razvoja še učinkovitejših oblik poučevanja predmeta.

Peti del je dodatek z naslovom Izbrana poglavja področij umetne inteligence. Na razumljiv način predstavi zmožnosti in omejitve tehnologije na sedanji stopnji. Brez dvoma nujno temeljno znanje za vse, ki poučujejo predmet informatika v srednji šoli.

Privlačna knjižica bo brez dvoma prispevala k odločanju srednješolskih učiteljev, da sami poglobijo svoje znanje in popestrijo pouk, pa tudi k širjenju splošnega informacijskega znanja, vpogled v zmožnosti in omejitve metodologije ekspertnih sistemov.

Katarina Puc

Predstavitev spletnega terminološkega slovarja v Mariboru

Terminološki slovar informatike http://www.ef.uni-lj.si/terminoloskislovar je na spletu že več kot dve leti. Število obiskov se povečuje iz meseca v mesec, kar nam potrjuje, da se slovar med uporabniki vse bolj uveljavlja.

Sodobna tehnologija ponuja številne možnosti prav v slovaropisju. To programska rešitev našega slovarja tudi lepo izkorišča: spletni slovar je odprt za uporabnike, omogoča jim iskanje na temelju slovenskega ali angleškega izraza pa tudi dodajanje mnenj k obstoječim izrazom, udeležbo v razpravah, vnašanje novih besed in razlag. Tako zdaj sodeluje že sedemdeset avtorjev, kar je tudi v skladu z našimi cilji: pritegniti v oblikovanje slovarja čim več strokovnjakov in tako pridobiti v čim krajšem času zbirko aktualnih izrazov.

Skupini strokovnih sodelavcev IZUM-a smo slovar predstavili 30. maja 2003 v Mariboru. Namen te predstavitve je bil seznanjanje širše javnosti z možnostmi, ki jih ponuja slovar, in pridobivanje novih avtorjev in urednikov slovarja. Po prikazu dela s slovarjem smo razpravljali o izboru izrazov in razlagah. Med drugim smo ugotovili, da uporabniki pogrešajo nekatere specifične, vendar pogoste izraze, npr. harvest, negotiation. To smo s skupnimi močmi v slovarju tudi popravili. Vsekakor pa upamo, da smo med udeleženci predstavitve pridobili še več sodelavcev.

Informatika posega na številne dejavnosti, zato ocenjujemo, da podobne predstavitve in razprave koristno prispevajo k oblikovanju slovarja.

Katarina Puc

DEXi
Računalniški program za večparametrsko odločanje

Uporabniški priročnik

Eva Jereb, Marko Bohanec, Vladislav Rajkovič

Moderna organizacija, 94 str., Kranj 2003

V založbi Moderna organizacija iz Kranja je letos izšel uporabniški priročnik za delo s programom DEXi, ki je zasnovan na metodi večparametrskega odločanja. V predgovoru avtorji pojasnijo namen priročnika, to je pomoč vsem tistim, ki si v procesih odločanja pomagajo s programom DEXi. Program je naslednik za okolje DOS razvitega programa DEX, ki ga poznamo že dlje časa in ki ima po navedbah avtorjev nekaj več funkcijskih možnosti, kar je nekoliko presenetljivo.

Priročnik je pregledno in sistematsko urejen v tri dele. Prvi del je splošen, teoretski in daje ustrezno podlago za razumevanje nadaljevanja. Na kratko opisuje odločitveni proces kot postopek odločanja med več možnostmi in večparametrsko odločanje kot razdelitev odločitvenega problema na podprobleme. Z večini bralcev domačim primerom kupovanja avtomobila pojasni osnovne pojme večparametrskega odločanja, kot so odločitveni model, parametri, funkcija koristnosti in variante. V nadaljevanju so opisane faze odločitvenega procesa, ilustrirane s primerom izbire kandidata za prosto delovno mesto. Sklepni del tega primera je analiza variant, ki je pri odločanju pomemben zaključni del procesa, saj moramo odločitve praviloma pojasniti in obrazložiti.

Drugi del vsebuje navodilo za uporabo programa DEXi. Urejeno je tako, da je primerno skorajda za začetnika za računalnikom, saj se začne z navodili za upravljanje z miško. Tem sledijo navodila za upravljanje z okni in uporabo gumbov, opisi menijev in njihove funkcije, delo z atributi, odločitvena pravila, variante in vrednotenje le-teh vse do prikaza, izpisov in povezave z drugimi programi. Navodilo je bogato in nazorno ilustrirano s slikami ekranov, kar olajšuje razumevanje, kaže pa tudi na obsežno delo, vloženo v razvoj programa in grafičnih vmesnikov.

Tretji del priročnika so primeri uporabe programa DEXi, izbrani tako, da so po vsebini razumljivi bralcem in uporabnikom programa: izbira prenosnega računalnika, ocenjevanje učiteljev in izbira kadrov. Po teoretičnem in opisnem delu so primeri lahko v pomoč pri reševanju odločitvenih problemov, vseeno pa bi bilo preveč pričakovati, da bi se lahko s priročnikom v roki lotili konkretnega zapletenega odločitvenega primera iz prakse. Vedeti moramo, da je priročnik le pomagalo pri delu s programom, ne pa učbenik teorije odločitev in da je treba znanje o tem in o uporabi DEXi pridobiti prej in drugje. Vsekakor pa lahko oceno sklenemo z ugotovitvijo, da so navodila za uporabo programa del urejenega razvoja programskih rešitev, kar je predvsem namen pričujoče knjižice, ki ga le-ta izpolnjuje pregledno in – končno, vendar ne nepomembno – z opazno skrbjo avtorjev za pravilno rabo strokovnega jezika.

Niko Schlamberger

SCI 2003 - 7th World Multi-Conference on Systemics, Cibernetics and Informatics	27.–30. 7. 2003	Orlando, Florida, USA		http://www.iiisci.org/sci2003 icastellano@iiis.org
PISTA '03 – International Conference on Politics and Information Systems: Technologies and Applications	31. 7.–2. 8. 2003	Orlando, Florida, USA		http://www.confinf.org/pista03/WebSite2003/default.asp pista@confinf.org
12th international conference on information systems development (Methods&Tools; Theory&Practice)	25.–27. 8. 2003	Melbourne, Australia		Isd2003@sims.monash.edu.au http://www.monash.edu.au/conference/isd2003
World IT Forum	27.–29. 8. 2003	Vilnius, LT	IFIP, UN/UNESCO/UNU/OECD	renaldas.gudauskas@kf.vu.lt Fax: + 370 2 366 104
2th EGOV Conference From E-Government to E-Governance	1.–5. 9. 2003	Prague, Czech Republic		
9th IFIP TC 13 Intl. Conference on Human – Computer Interaction	1.–5. 9. 2003	Zürich, CH	IFIP TC 13	g.w.m.rauterberg@tue.nl http://www.interact2003.org
MMNS 2003 – 6th IFIP/IEEE International Conference of Management of Multimedia Networks and Services	7.–10. 9. 2003	Belfast, Northern Ireland	IFIP	http://www.ee.qub.ac.uk/dsp/mmns2003/ http://www.ifip.or.at/ dh@ifip.or.at
SOR '03	24.–28. 9. 2003	Podčetrtek	Slovensko dru'tvo INFORMATIKA	http://www.drustvo-informatika.si/sekcije/sor lidija.zadnik@uni-lj.si
PACT'03 – 12th International Conference on Parallel Architectures and Compilation Techniques	27. 9.–1. 10. 2003	New Orleans, LA, US		http://www.pactconf.org pact2003@gup.jku.at
FORTE 2003 – 23rd International Conference on Formal Techniques for Networked and Distributed Systems	29. 9.–2. 10. 2003	Berlin, Germany	IFIP TC6/WG6.1	http://www.forte2003.de.vu
CHARME 2003	21.–24. 10. 2003	L'Aquila, Italy		http://www.di.univaq.it/charme2003
CODES-ISSS MERGED CONFERENCE	1.–3. 10. 2003	Marriot Hotel, Newport Beach, California, USA		codesissss@cs.ucr.edu http://www.ece.uci.edu/codes+isss/
MWCN 2003 – International Conference on Mobile and Wireless Comunications Networks	27.–29. 10. 2003	Singapore		http://www.icr.a-star.edu.sg/mwcn2003/ dh@ifip.or.at
CASES 2003 – Int. Conf. on Compilers, Architecture and Synthesis for Embedded Systems	30. 10.–1. 11. 2003	Doubletree Hotel, San Jose,California, USA		http://www.casesconference.org cases@cs.ucr.edu
MICRO-36 – 36rd IEEE/ACM International Symposium on Microarchitecture	3.–5. 12. 2003	San Diego, CA, USA		http://www.microarch.org/micro36/ micro2k3@cse.wustl.edu
EIS 2004 – ENGINEERING in INTELLIGENT SYSTEMS	29. 2.–3. 3. 2004	Island of Madeira, Portugal	University of Madeira, Portugal University of Maribor, Slovenija ICSC Canada/USA/Europe	http://www.icsc-naiso.org planning@icsc.ab.ca

Včlanite se v Slovensko društvo INFORMATIKA

Pristopna izjava

Želim postati član Slovenskega društva INFORMATIKA

Prosim, da mi pošljete položnico za plačilo članarine SIT 5.200 (kot študentu SIT 2.400) in me sproti obveščate o aktivnostih v društvu.

(ime in priimek, s tiskanimi črkami)

(poklic)

(domači naslov in telefon)

(službeni naslov in telefon)

(elektronska pošta)

Datum: Podpis:

Članarina SIT 5.200,- (plačljiva v dveh obrokih) vključuje tudi naročnino za revijo Uporabna informatika. Študenti imajo posebno ugodnost: plačujejo članarino SIT 2.400,- in za to prejemajo tudi revijo. Izpolnjeno naročilnico ali pristopno izjavo pošljite na naslov:
Slovensko društvo INFORMATIKA, Vožarski pot 12, 1000 Ljubljana.

Lahko pa izpolnite obrazec na domači strani društva: **http://www.drustvo-informatika.si**

Naročilnica na revijo UPORABNA INFORMATIKA

Revijo naročam(o)
- ☐ s plačilom letne naročnine SIT 4.600
- ☐ izvodov po pogojih za podjetja SIT 13.800 za eno letno naročnino in SIT 8.900 za vsako nadaljnjo naročnino
- ☐ po pogojih za študente letno SIT 2.000

(ime in priimek, s tiskanimi črkami)

(podjetje) (davčna številka)

(ulica, hišna številka)

(pošta)

Datum: Podpis:

Naročnino bomo poravnali najkasneje v roku 8 dni po prejemu računa

INTERNET

Vse bralce revije obveščamo, da lahko najdete domačo stran društva na naslovu: **http://www.drustvo-informatika.si**
Obiščite tudi spletne strani mednarodnih organizacij, v katere je včlanjeno naše društvo: IFIP: **www.ifip.or.at,** ECDL: **www.ecdl.com,** CEPIS: **www.cepis.com**

Razprave

Matjaž Jaušovec, Boštjan Brumen, Tatjana Welzer Družovec
Analiza varnostne ogroženosti

Marko Bajec, Marjan Krisper
Agilne metodologije razvoja informacijskih sistemov

Primož Peterlin
Odprto programje

Miroslav Ribič, Marjan Lončarič, Andrej Kovačič
Informatizacija procesa upravljanja človeških virov

Rešitve

Stane Ravnik
Aktivno arhiviranje

ISSN 1318-1882
9 771318 188001