Inteligentni sustavi za modeliranje poslovnih procesa u razvoju informacijskog sustava

Francesca Gržinić Kuljanac
Grad Opatija
Maršala Tita 3, 51410 Opatija, Hrvatska
francesca.grzinic@opatija.hr

Sažetak – Razvoj informacijskog sustava je dugotrajan proces koji zahtjeva niz aktivnosti u kojoj je jedna od važnijih faza analize. Čest problem u razvoju informacijskog sustava je prekoračenje planiranoga budžeta i vremena. U fazi analize ključni zadatak je prikupiti korisničke zahtjeve i temeljem njih izraditi prijedlog sustava. No ponekad korisnik ne zna točno izraziti zahtjeve ili projektant dobro ne poznaje poslovni sustav te krivo shvaća korisniče zahtjeve. Ovo unosi grešku u daljnje faze razvoja informacijskog sustava i time povećava rizik za neuspjehom projekta. Problem predstavlja i dvosmislenost i neodređenost prirodnoga jezika kojim se korisnici koriste kako bi opisali svoje zahtjeve. Prevođenje korisničkih zahtjeva pisanih ograničenim prirodnim jezikom u posebni formalni jezik za iskaz modela procesa predstavlja prijedlog za rješenje navedenoga problema. Namjera je navedeno implementirati u novom inteligentnom sustavu za modeliranje poslovnih procesa. Kao uvod u istraživanje koje za cilj ima razvoj navedenoga inteligentnog sustava provedena je analiza postojećih mehanizama automatizacije aktivnosti koje se provode u razvoju informacijskog sustava.

Ključne riječi: informacijski sustav, inteligentni sustav korisnički zahtjev, modeliranje procesa.

I. UVOD

Proizvodnja, trgovina, uslužne djelatnosti, telekomunikacije, bankarstvo, osiguranje, zdravstvene usluge, školstvo – bilo kojom poslovnom djelatnošću da se tvrtka bavi, ona se u provedbi svojih poslovnih aktivnosti direktno ili indirektno oslanja na informacije. Pored sirovina, energije i radne snage, danas je informacija jedan od glavnih resursa koji tvrtki daju konkurentsku prednost. Ako bismo usporedili poslovnu organizaciju sa živim organizmom, onda bismo mogli reći da su informacije ono što kola njenim krvotokom.

Informacije se danas stvaraju i distribuiraju brže i jeftinije no ikad. Jedan od faktora koji su doprinijeli tome je brzi razvoj informacijske tehnologije, te njezina laka dostupnost. [1]

Za uobičajeno odvijanje izvršnih i upravljačkih poslovnih procesa, treba prikupiti, obraditi, čuvati i dostavljati veliki broj informacija o stanju poslovnog sustava, što premašuje ljudske mogućnosti. Stoga se ljudi služe računalno podržanim informacijskim sustavom – posebnim dijelom poslovnoga sustava koji iz svoga okruženja prima ulazne informacije, upisuje ih, pohranjuje i obrađuje, te ih prosljeđuje svome okruženju kao izlazne informacije. Informacijski sustav je skup dizajniranih informacijskih objekata koji predočuju koncepte modela organizacijskog sustava, i to njihovu prošlost i sadašnjost, implementiranih na resurse koji su ili nositelji informacija ili izvode procese nad podatcima, a informacijski objekti razmjenom informacija komuniciraju (povezani su) međusobno i s okolinom. [2]

I dok razvoj informacijske tehnologije napreduje, jedno ostaje konstanta – razumijevanje onoga što korisnici doista trebaju od programskog proizvoda, a što verbaliziraju u svojim zahtjevima, jest izazov svakog sudionika u razvoju informacijskog sustava i time ga je poželjno riješiti. [3] Zanemarivanje ranih faza razvoja informacijskog sustava često je uzrokovano željom da se tijekom razvoja što prije dobiju opipljivi rezultati za krajnjega korisnika. Međutim upravo zanemarivanje važnosti tih faza može dovesti do značajnoga povećanja troškova razvoja, njegova kašnjenja, pa u konačnici i dovesti u pitanje uspjeh razvoja informacijskog sustava. Stoga faza u kojoj se prikupljaju korisnički zahtjevi i faza analize slove kao najkritičnijima u procesu razvoja informacijskog sustava. [4]

Kako bi se dobio pregled šireg područja primjene inteligentnih sustava u modeliranju procesa unutar razvoja informacijskog sustava, pretraživala se i analizirala relevantna međunarodna literatura i znanstveni članci dostupni kroz baze podataka kao što su Science Direct, IEEE, ACM Digital Library, Springer Link te Google Scholar. Tijekom pretrage su se koristili sljedeći ključni pojmovi: requirements specification, formal specification, formal methods, software development, data flow diagram, natural language processing, natural language requirements. Osim pretrage po ključnim pojmovima, relevantna literatura se pronalazila i kroz popise literature navedene unutar već pronađenih članaka.

Ključni pojmovi i problemi u domeni primjene inteligentnih sustava za modeliranje poslovnih procesa u razvoju informacijskog sustava opisani su u prvom, drugom i trećem poglavlju. U prvom uvodnom poglavlju je opisan značaj informacijskog sustava u poslovnim organizacijama. Drugo poglavlje opisuje pojam inteligentnog sustava i potrebe za uključivanjem takvih sustava u računalne sustave. Treće poglavlje prikazuje neke osnovne probleme koji se javljaju u procesu razvoja informacijskog sustava, posebno u njegovim ranim fazama. U istom poglavlju opisana je metoda dijagrama toka podataka kao najraširenija način prikaza zahtjeva korisnika. Četvrto poglavlje pruža pregled područja primjene inteligentnih sustava u razvoju informacijskog sustava. Rad završava zaključkom u kojemu su navedeni daljnji pravci istraživanja koji za cilj ima razvoj novog inteligentnog sustava za modeliranje poslovnih procesa u razvoju informacijskog sustava.

II. INTELIGENTNI SUSTAVI

Inteligentnim sustavima se smatraju oni sustavi koji imaju sposobnost prikupljanja i uporabe znanja, kao i sposobnost interakcije, komuniciranja s vanjskim svijetom odnosno korisnicima ili pak s drugim inteligentnim sustavom. To su sustavi koji imaju sposobnost zaključivanja, sposobnost obrade i zamjene znanja, kao i sposobnost objašnjenja svog ponašanja. Umjetna inteligencija je disciplina koja se bavi oblikovanjem inteligentnih sustava koji implementiraju ona svojstva ljudskog ponašanja koja se smatraju inteligentnim. [5] Od inteligentnog sustava se očekuje rješavanje problema iz neke problemske domene na razini i u opsegu kvalitete jednog eksperta.

Potreba za povećanjem kvalitativne razine podrške računalnih sustava različitim područjima ljudskoga djelovanja dovela je do toga da se u njih sve više ugrađuju mehanizmi koji će im omogućiti inteligentnije ponašanje – ponašanje koje se temelji na znanju. Ove mehanizme proučava područje umjetne inteligencije, a njihovom primjenom se razvijaju inteligentni sustavi (sustavi temeljeni na znanju). [6]

Upravo je mogućnost primjene mehanizama umjetne inteligencije u rješavanju već spomenutih problema koji se pojavljuju u ranim fazama razvoja informacijskog sustava osnovna istraživačka motivacija autorice, a krajnji cilj istraživanja je razvoj inteligentnog sustava kao podrške modeliranju poslovnih procesa koji bi bio koristan širem krugu sudionika razvoja informacijskog sustava. Taj inteligentni sustav bi trebao korisnički opis odvijanja poslovnih aktivnosti neke poslovne organizacije iskazan ograničenim prirodnim jezikom prevesti u formalizirani tekstualni prikaz poslovnih procesa metodom dijagrama toka podataka.

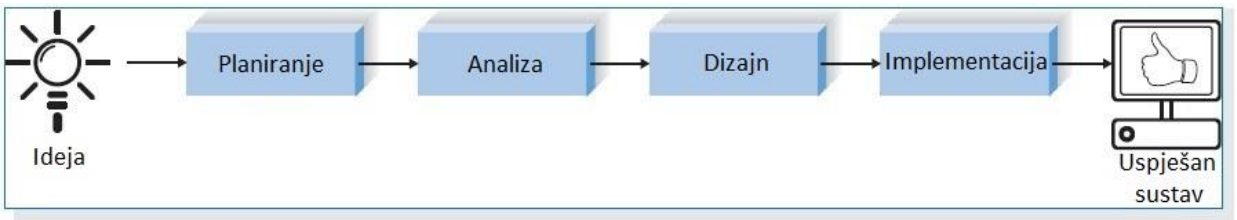
III. RAZVOJ INFORMACIJSKOG SUSTAVA

Proces razvoja informacijskog sustava prati opće faze razvoja sustava: planiranje, analiza, oblikovanje i implementacija (Slika 1.). [7]

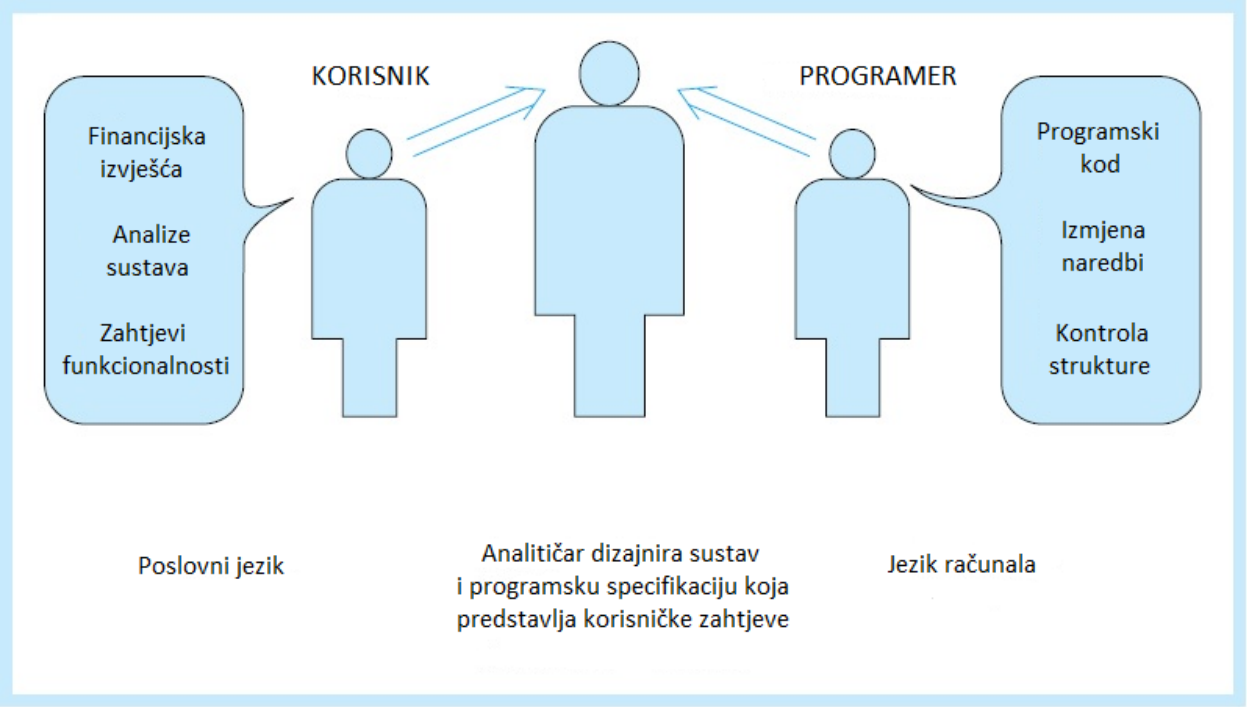
Svaka od navedenih faza ima svoj specifičan zadatak. Tako faza analize odgovara na pitanja tko, gdje i kada se koristi sustavom, te koja je svrha sustava. U procesu razvoja informacijskog sustava se tijekom ove faze prikupljaju korisnički zahtjevi primjenom različitih tehnika - intervju s korisnicima, radni sastanci čitavih timova, analiza postojećeg sustava. No cilj faze analize je samo jedan - doći do valjane specifikacije korisničkih zahtjeva. Analiza prikupljenih podataka, uz naputke pokrovitelja projekta, dovodi do razvoja koncepta novog sustava. Koncept se potom koristi kao osnova za razvoj niza modela koji opisuju kako će se posao obavljati ukoliko se novi sustav implementira. Jedan od tih modela je uobičajeno i model koji prikazuje procese i podatke nužne za poslovni sustav. [8]

Dakle na samom početku razvoja informacijskog sustava, u fazi analize, potrebno je prikupiti zahtjeve ključnih korisnika i njih formalizirati, te temeljem njih izraditi prijedlog sustava. Ova faza razvoja informacijskog sustava nije imuna na pogreške. Neki od razloga pojave pogrešaka su:

- većina korisnika ima snažno izrađenu vizualnu ideju o promjenama koje bi htjeli uvesti, međutim imaju problema s verbaliziranjem takve ideje [8]
- pokrovitelji projekta uključeni u proces razvoja sustava često zahtjeve izražavaju „svojim“ stručnim jezikom, međutim rijetko su u stanju formalizirati te koncepte na razumljiv, nedvosmislen način [9]



Slika 1. Životni ciklus razvoja sustava [16]



Slika 2. Uloga analitičara [1]

- pokrovitelji projekta često podrazumijevaju da analitičari znaju za određene zahtjeve, bez da ih sami izričito definiraju [4]
- projektant nije dovoljno dobro upoznat s poslovnim sustavom.

Neuočene i neispravljene pogreške koje se jave u početnim fazama razvoja informacijskog sustava dalje se propagiraju u naredne faze uzrokujući time dobro poznate probleme - troškovi i vrijeme razvoja informacijskog sustava daleko premašuju zadane okvire, prosječno prekoračenje troškova projekata razvoja programskih proizvoda od 189%, prosječno prekoračenje rokova izrade od 222%, samo 16.2% projekata dovršeno na vrijeme, u okviru predviđenih sredstava i sa svim predviđenim funkcijama. [10], [11] Gotovo 60% pogrešaka koje se pojave tijekom razvoja informacijskog sustava uzrokovane su unutar faze prikupljanja zahtjeva korisnika. [27] Autori G. Curtis i D. Cobham u svojoj knjizi "Business Information Systems" [12] opisuju kako nastaje potreba za informacijskim sustavom, te prikazuju ulogu faze analize, odnosno analitičara. Dakle, uobičajeno je da se projekti pokreću zato što postoji problem te se uvidio način kojim bi se on mogao riješiti ili se pak prepoznala mogućnost kojom bi se postojeći sustav poboljšao. U svakom slučaju ključan je korisnik sustava koji prepoznaje mogućnosti. On je taj koji može pružiti informacije o postojećem sustavu, ali i specificirati koje zahtjeve on postavlja na novi sustav.

Razvojni programeri su pak odgovorni za pretvaranje tih zahtjeva u program, bilo pisanjem programskog koda od početka, pomoću programskom generatora ili pak primjenom razvojnog alata kao što je CASE. U svakom slučaju programer kontrolira ponašanje programa i prilagođava ga korisniku, ali na način da vidi probleme u računalnim terminima, odnosno programer i korisnici pričaju drukčijim jezicima i tu nastaje komunikacijski jaz. Kako bi se taj jaz prevazišao ulogu preuzimaju analitičari poslovnih sustava, osobe koje imaju sposobnost komunicirati s korisnicima i razumjeti njihove zahtjeve s jedne strane, a opet imaju stručno znanje o računalima što im daje sposobnost da te zahtjeve prenesu programerima na način da ih oni mogu razumjeti. U osnovi analitičari prevode znanje iskazano korisničkim jezikom u znanje iskazano nekim inženjerskim formalnim jezikom (analitičkim, dizajnerskim, programerskim itd.) Važno je da analitičar bude dobar komunikator koji može razmišljati i komunicirati iz obje perspektive. (Slika 2.)

Autori G. Curtis i D. Cobham napominju kako prevođenje ovih zahtjeva nije jednostavan proces i on se ne može gledati kao prevođenje jednoga prirodnoga jezika u drugi (npr. prevođenje s hrvatskog na engleski jezik), već je taj proces bliže aspektu arhitekture i građevine. Klijent (korisnik) je taj koji iskazuje svoje razumijevanje i želje o tome kako bi zgrada trebala izgledati i koje funkcionalnosti bi trebala pružati. Arhitekt (analitičar) uzima te karakteristike (korisničku izjavu o zahtjevima sustava) i prevodi ih u opću skicu sustava koja će zadovoljiti te zahtjeve (logički model). Nakon usuglašavanja s klijentom, arhitekt izrađuje detaljni plan (detaljna specifikacija programa) temeljem kojega graditelj (programer) može raditi. Poput znanja o građevini potrebnog arhitektu da bi kvalitetno obavio zadatak, tako je i analitičaru potrebno znanje o računalima.

A. Modeliranje poslovnih procesa metodom dijagrama toka podataka

U fazi analize se mogu koristiti razne metode za stvaranje modela, no među popularnije spadaju one koje potiču dekompoziciju po funkcionalnostima [13]. Jedna od takvih metoda je i metoda dijagram toka podataka (engl. Data Flow Diagram - DFD) kojom se modeliraju funkcionalnosti i tok podataka jedne funkcionalnosti prema drugoj – odnosno modeliraju se poslovni procesi

koji implementiraju pojedinu funkcionalnost. Dijagram toka podataka je raširen način prikaza zahtjeva korisnika i njegova prednost je u tome što je jednostavan za razumijevanje. Njime nastaje model procesa čija je pouzdanost ključni element za unaprjeđenje performansi sljedećih faza u razvoju poslovnih sustava. [14]

Jezik dijagrama toka podataka sastoji se od minimalnog broja koncepata bliskih čovjeku, odnosno grafičkih simbola, standarda imenovanja i sintaksnih pravila. Sastoji se od četiri koncepta od kojih je svaki prikazan korištenjem različitih grafičkih simbola. Unutar dijagrama toka podataka raspoznajemo i dvije grafičke inačice prikaza koncepata – simboli po Chris Gane i Trish Sarson, te drugi simboli po Tom DeMarco i Ed Yourdon (Slika 3).

Elementi dijagrama toka podataka	Gane i Sarson simboli	DeMarco i Yourdon simboli
Svaki *proces* sadrži broj naziv (glagolska fraza) opis najmanje jedan ulazni tok podatka najmanje jedan izlazni tok podatka	1 Naziv	Naziv
Svaki *tok podatka* sadrži naziv (imenica) opis jednu ili više poveznica s procesom/ima	Naziv →	Naziv →
Svako *spremište* sadrži broj ime (imenica) opis jedan ili više izaza toka podatka jedan ili više ulaza toka podatka	D1 Naziv	D1 Naziv
Svaki *vanjski sustav* sadrži naziv (imenica) opis	Naziv	Naziv

Slika 3. Koncepti dijagrama toka podataka [16]

Dijagram toka podataka je logički model koji prikazuje što sustav radi, odnosno model koji grafički prikazuje logičku povezanost tokova podataka kroz sustav, granicu sustava, procese i podatke. Sastoji se iz četiri koncepta: [15]

- Proces je skup aktivnosti odnosno funkcija koje na temelju ulaznih tokova podataka generiraju jedan ili više izlaznih tokova. Imenovanje procesa treba biti kratko ali s dovoljno informacija koje će čitatelju omogućiti shvaćanje što proces obavlja. Važno je naglasiti kako proces obavlja po jednu aktivnost, pa stoga sistem analitičari izbjegavaju korištenje veznika "i" u imenima procesa. Procesi se obično označavaju glagolskom imenicom (imenicom i glagolskom radnjom). Svaki proces treba sadržavati barem jedan ulazni tok i barem jedan izlazni tok, te imati jedinstveni identifikacijski broj, ime i opis.
- Tok podatka predstavlja podatak ili skup podataka koji se kreću kroz sustav. To mogu biti izvještaji, dokumenti i sl., a možemo ih podijeliti u dvije vrste – ulazni i izlazni podaci. Tokovi podataka su zapravo poveznica procesa. Ulazni tokovi podataka ulaze u proces, koriste se i mijenjaju tijekom obavljanja procesa, dok izlazni nastaju kao rezultat procesa. Uvijek imaju definiran smjer i započinju ili završavaju unutar procesa. Naziv toka podatka jest jedinstveni oblik imenica ili pridjeva i imenica, a označava točno one podatke koji se prenose tim tokom.
- Spremište podataka je skup podataka koji se pohranjuju unutar sustava. Svako spremište podataka je imenovano imenicom s dodijeljenim identifikacijskim brojem i opisom, a sve aktivnosti nad spremištem obavljaju se isključivo korištenjem procesa. Tokovi koji izlaze iz skladišta podataka ukazuju na to da se informacija dobiva iz skladišta, dok tokovi koji ulaze u skladište označavaju spremanje podataka. Svako skladište mora imati najmanje jedan ulazni tok osim ako je kreiran i održavan od strane nekog drugog informacijskog sustava. Isto tako skladište mora imati bar jedan izlazni tok podatka (ukoliko se ti podaci ne koriste, nema razloga niti za njihovo pohranjivanje).
- Vanjski sustav predstavlja osobu, organizaciju, organizacijsku jedinicu ili drugi sustav koji je vanjski u odnosu na promatrani sustav, ali je u interakciji s njime. Vanjski sustav prima ili šalje podatke i služi za uspostavljanje granice promatranoga sustava. Svaki vanjski sustav ima svoje ime i opis. [16]

IV. INTELIGENTNI SUSTAVI U MODELIRANJU POSLOVNIH PROCESA

Zahtjevi na sustav koje korisnici iznose u prirodnome jeziku često su dvosmisleni i neodređeni. [17] Višestruko složene rečenice, sastavljene od nezavisnih i zavisnih rečenica nisu pogodne za računalo i programe koji zahtijevaju preciznost, formalnost i jednostavnost.

Kao što je ranije navedeno, u razvoju informacijskih sustava sudjeluje veći broj dionika - korisnici, projektni menadžeri, menadžeri kvalitete, eksperti poslovnih sustava, analitičari, dizajneri, razvojni programeri itd. Kvaliteta konačnog proizvoda - informacijskog sustava - ovisit će i o kvaliteti komunikacije, odnosno istovjetnosti zahtjeva tih dionika. Stupanj istovjetnosti zahtjeva često ovisi o sposobnost analitičara da ih usuglasi. Bez visokog stupnja istovjetnosti zahtjeva umanjuje se vjerojatnost razvoja zadovoljavajućeg informacijskog sustava.

U razvoju informacijskog sustava početnu točku obično čine neformalne specifikacije sustava u prirodnom jeziku. No, kako bi se sustav mogao razviti, potrebno je te

specifikacije pretvoriti u formalan zapis, i to čim ranije u procesu razvoja [18].

Dizajn budućih sustava najčešće je izrađen korištenjem grafičkih metoda. I dok se dijagram toka podataka često koristi upravo iz razloga što je široko primjenjiv za vizualnu specifikaciju sustava, kao nedostatak se spominje manjak njegove formalnosti i preciznoga razumijevanja [19]. U tom smislu strukturne i formalne metode spominju se kao komplementarne i poželjne za korištenje. One bi trebale olakšati razumijevanje budućeg sustava svim dionicima u procesu njegova razvoja.

Autor Derek Andrews u svom radu navodi kako je za bolje razumijevanje računalnog programa važna percepcija - što je računalni program, a što je njegova specifikacija. Računalni program može biti izveden od strane nekog mehaničkog ili elektroničkog uređaja, najčešće računala, te je njegovo izvođenje obično ekonomično (u smislu da je jeftino i ne zahtjeva ljudsku intervenciju). S druge strane, specifikacije se mogu izvesti samo od stane nekog oblika inteligentnog uređaja, obično ljudskog uma. Dobra definiranost specifikacije uvjetuje lakoću razumijevanja kompleksnog sustava. No bez obzira na kvalitetu specifikacije, bila ona i matematički opisana, računalo ju teško može izvesti. Slično je i u obrnutoj situaciji. Pokuša li se računalni program izvesti u ljudskom umu, potrebno je dosta intelektualnog napora kako bi se shvatilo što se izvodi. Dakle, iako je specifikacija zapravo verzija programa odnosno sustava, glavna razlika je u uređaju koji ga izvodi.

U tom smislu i tim pogledom na specifikaciju postavlja se i osnovni zahtjev na jezik specifikacije – jezik treba biti i neformalan i formalan, u korist svih dionika. [20]

U kasnim sedamdesetima prošlog stoljeća, autor Halstead govori o mogućnosti da teorija programa i kvantitativne analize može počivati na prirodnom jeziku korištenjem kategorizacije operatora i operanada iz prirodnih elemenata jezika. [21]

Osamdesetih godina prošlog stoljeća autor Abbott napisao je članak u kojem govori o vrstama podataka i njihovom korištenju u programskom dizajnu. Autor prezentira tehniku za razvoj programa kroz neformalan ali precizan engleski jezik uz Ada programski dizajn. [22] Konkretno, članak utvrđuje opće imenice u prirodnom jeziku analogne apstraktnim tipovima podataka u programskim jezicima. Njegov rad su nastavili Enomoto, Saeki i Horai, autori koji su predložili postupak za dobivanje formalne specifikacije iz one neformalne pisane u prirodnom jeziku. Govore o procesu koji se sastoji iz dvije glavne aktivnosti - dizajn i obrada. Ovaj rad i istraživanje je temeljeno na objektno orijentiranom modeliranju. [23]

Godine 1990. Carasik, Johnson, Patterson i Von Glahn ističu ograničenja ER modela u definiranju semantike, zalažući se za konceptualno modeliranje i tehnike prikaza znanja koje bi formalno mogle prikazati neko značenje. Međutim i konceptualni modeli mogu biti nerazumljivi korisnicima. [24]

Cordes i Carver su 1992. godine krenuli s prvim pokušajima razvoja automatiziranih alata za analizu zahtjeva te automatskim generiranjem objektno orijentiranih modela iz zahtjeva. Prijevod neke domene znanja u objektno orijentirane modele jest automatiziran, međutim prijevod prvotnih zahtjeva u prikaz znanja prikladan za bazu podataka zahtjeva ljudsku interakciju. Ono što se uvidjelo jest da je prijevod formaliziranog znanja u objektno orijentirane modele izrazito osjetljivo na kvalitetu inicijalnih specifikacija zahtjeva. [25]

Autor J. Börstler u svom radu iz 1996. godine spominje probleme korisnika koji ne razumiju formalne zahtjeve definirane grafičkim metodama za modeliranje procesa i podataka (poput DTPa, EVa i sl.), te predlaže neformalnu notaciju koja ne ograničava korisnike - RECORD (engl. REquirements COllection, Reuse, and documentation). Taj pristup omogućava korisniku aktivno sudjelovanje u procesu razvoja softvera na način da svoje zahtjeve unosi u sustav u obliku obrasca, kojega zatim RECORD sustav pretvara u dijagrame uporabe (engl. Use Case Diagram) koji se zatim transformiraju u objektno orijentirane modele. [26]

Beum-Seuk i Barret su autori koji u svom radu iz 2004. godine predlažu metodologiju za formalan razvoj programskih sustava temeljem zahtjeva korisnika na prirodnom jeziku koja se temelji na teoriji dvorazinskih gramatika (engl. Two-Level Grammar - TLG) i objektno orijentiranog dizajna. Dvorazinska gramatika je gramatika kojom se generira neka druga gramatika. Autori napominju, da iako je prirodni jezik sam po sebi objektno orijentiran i deskriptivan, njegova sintaksa i semantika nisu dovoljno formalne da bi se mogao koristiti direktno kao programski jezik. Kako bi postigli svoj cilj, potrebno je napraviti nekoliko koraka. Prije svega korisničke zahtjeve je potrebno obraditi kako bi bili u odgovarajućoj formi - manualnim procesom se provjerava pravopis, gramatičke pogreške, korištenje istih termina kroz čitav zahtjev i sl. Zatim slijedi automatsko prevođenje u XML format koji se koristi za punjenje baze znanja u kojoj se pohranjuje dokumentacija, sintaksa, semantika i pragmatične informacije. U fazi stvaranja baze znanja, uočavaju se i rješavaju problemi dvosmislenosti – automatski ili to radi programski inženjer. Primjenom dvorazinske gramatike baza znanja se automatski prevodi u formaliziranu specifikaciju koja je međukorak za dobivanje konače specifikacije u formi sukladnoj VDM++ (engl. Vienna Development Model – objektno orijentiranoj metodi razvoja računalnog sustava [40]). [27]

Nekoliko znanstvenika u svojim radovima navodi kako nema formalnih jezika koji se koriste za prikaz dijagrama toka podataka [29], [30], [31]. Pa tako niti u ovom

istraživanju autorica nije pronašla radove koji se temelje na primjeni procesiranja prirodnog jezika u fazi analize za prijevod tekstualne specifikacije u tekstualni dijagram toka podataka, odnosno u njegovu verbalizaciju.

Autor N. Boyd u svojem radu iz 1999. godine govori o tome kako razvoj informacijskog sustava obično uključuje različite interesne skupine korisnika i analitičara. I dok korisnici imaju svoje stavove i mišljenja u odnosu na vrstu problema kojeg programsko rješenje treba zadovoljiti, analitičari imaju zadatak posrednika između zainteresiranih strana kako bi se postigao kompromis o raznim potrebama i prioritetima. Slično kako su to kasnije pisali G. Curtis i D. Cobham u [32], i ovaj autor napominje kako analitičari služe i kao prevoditelji između interesnih skupina korisnika i razvojnih programera iz razloga što oni često govore „različitim jezicima“. Dok korisnici u svojim zahtjevima naglasak stavljaju na poslovne potrebe, ciljeve i funkcionalnosti, razvojni programeri govore u kontekstu modela, dizajna i programiranja. Odgovornost analitičara je razumjeti ograničenje formaliziranih modela i prirodnog jezika, te uloge prevodioca. Prema istraživanju istoga autora, korištenje prirodnog jezika do sada nije imalo velikog utjecaja na razvoj softvera [33].

2006. godine su Ambriola i Gervasi predstavili alat za modeliranje zahtjeva i analizu pod nazivom CIRCE. Sustav raščlanjuje rečenice tekstualnog korisničkog zahtjeva preko posebnoga parsera i korisničkoga rječnika. Potom stručnjak dodaje semantiku u raščlanjeni zahtjev kako bi kreirao konačni model. U ovom sustavu se javlja ograničenje korištenja teksta koji parser može obraditi. [34]

Autor Mu, zajedno sa suradnicima, 2009. godine objavljuje rad u kojem prikazuje raščlanjivanje korisničkih zahtjeva pisanih prirodnim jezikom s ciljem procjene nefunkcionalnih zahtjeva koji se pojavljuju u njima. Postoji standardizirana forma unosa korisničkih zahtjeva, a konačni rezultat obrade nije konceptualni model, već novi format zapisa koji je pogodan za njihove specifične potrebe. [35]

2011. godine autori Yue, Briand i Labiche daju prikaz transformacijskog procesa korisničkih zahtjeva u modele. U svom istraživanju su otkrili kako postoji pet tehnika obrade koje se koriste kombinirano ili pojedinačno: leksička, sintaktička, semantička i pragmatična analiza i kategorizacija. Uočeno je da dobiveni rezultati transformacijskog procesa imaju nisku učinkovitost, te im nedostaje mehanizam za evaluaciju. Njihovo Istraživanje je obuhvatilo 20 radova, kategoriziranih u 16 transformacijskih pristupa. Od navedenih pristupa, sedam ih je automatizirano, dva nisu automatizirana ali ih se može automatizirati, dva pristupa zahtijevaju angažman korisnika kako bi se izvela poluautomatska transformacija, i ostali pristupi se izvršavaju manualno. [36]

Poluautomatsku pretvorbu korisničkoga zahtjeva pisanoga u prirodnom jeziku u model prikazan dijagramom uporabe obrađuju Elbendak i suradnici 2011. godine. Tekstualni zahtjevi u prirodnome jeziku se raščlanjuju parserom ali uz manualnu intervenciju čime se potpomaže proces modeliranja. Definirana su i posebna lingvistička pravila koja korisnika vode kroz stvaranje modela. Rezultat je prošireni ER model. [37]

I dok postoje mnogi alati za izražavanje i vizualizaciju modela, ne postoji mnogo alata koji pomažu analitičarima u fazi analize zahtjeva. Stoga autori Vidhu Bhala i Abirami u svome radu iz 2013. godine smatraju vrlo vrijednim artefakte koji se odnose na konceptualno modeliranje i vizualizaciju funkcionalnih zahtjeva. Autori smatraju da automatizirana izrada konceptualnih modela omogućuje analitičarima fokusiranje na dionike projekta kao i na analizu i rafiniranje modela, umjesto da se "troši" vrijeme na njihovo kreiranje. Njihova metoda automatizacije počiva na osnovama engleskoga jezika. Rečenice prirodnog jezika koje tvore funkcionalnu specifikaciju su raščlanjene kako bi se razumjela njihova gramatička struktura. Različite komponente dizajna, kao što su to entiteti i veze, klase i atributi se izdvajaju iz raščlanjenih rečenica. Unos korisničkih specifikacija nije ograničen strogo definiranim obrascem. [38]

Jedna od metoda formalizacije tekstualnog znanja iskazanoga u prirodnome jeziku je i Metoda formalizacije znanja iskazanog tekstom (engl. Formalization Method for the Text Expressed Knowledge - FMTEK) [39]. Ova metoda omogućava formalizaciju tekstualnog znanja na način da nad njome računalo može provodi matematičko logičke operacije. Namjera autorice je uključiti navedenu metodu u inteligentni sustav za potporu modeliranju procesa kao jedan od mehanizama kod prevođenja korisničke specifikacije u verbalizirani dijagram toka podataka.

V. ZAKLJUČAK

Brzi razvoj i dostupnost informacijsko komunikacijske tehnologije, te veliki porast potrebe za pravovremenim i valjanim informacijama stavlja pred razvoj informacijskog sustav nove izazove. Uz to je i svaki informacijski sustav vrlo specifičan za svaku vrstu poslovanja.

Pri izgradnji informacijskog sustava nije rijetkost da se rane faze razvoja zanemaruju, te da se umanjuje njihova važnost za ukupnu kvalitetu isporučenog proizvoda. Razlog tome je što se čini da rane faze ne daju "opipljive" rezultate, pa faza analize često predstavlja "kamen spoticanja" kada je u pitanju uspjeh razvoja informacijskog sustava. Stoga je želja autorice dati prijedlog rješenja kojim bi se olakšao razvoj sustava, skratilo vrijeme njegova razvoja i smanjilo pojavu grešaka u ranijim fazama razvoja.

U radu je opisana važnost informacijskog sustava kao osnovnoga mehanizma efikasnoga upravljanja

informacijama nužnih za provođenje poslovnih aktivnosti. Posebno je naglašeno da upravljanje potrebnim informacijama premašuje ljudske mogućnosti te se stoga primjenjuje informacijsko komunikacijska tehnologija u razvoju računalnoga sustava kao potpora informacijskom sustavu.

Inteligentni sustavi računalnim sustavima daju novu razinu kako u kvaliteti potpore raznim ljudskim djelatnostima, tako i u njihovoj primjenjivosti. Jedna od mogućih primjena inteligentnog sustava je i ona kod modeliranja poslovnih procesa u razvoju informacijskog sustava. Namjera autorice je razviti inteligentni sustav koji će zadovoljiti ulogu prevoditelja između različitih dionika u procesu razvoja informacijskog sustava.

U modeliranju poslovnih procesa iz korisničkih zahtjeva, autorica predlaže korištenje metode dijagrama toka podataka jer je ona široko rasprostranjena i razumljiva širem krugu korisnika, te je sastavljena od malog broja koncepata i jednostavnih pravila.

Kako bi se omogućilo korisniku da komunicira s modelom procesa (npr. u obliku postavljanja pitanja i dobivanja odgovora), on treba biti iskazan u nekom verbaliziranom obliku. To znači da je potrebno korisnički zahtjev iskazan u nekom ograničenom prirodnom jeziku prevesti u posebni formalni jezik kojim se može tekstualno opisati model procesa primjenom metode dijagrama toka podataka.

LITERATURA

[1] Graham Curtis, David Cobham, Business Information System analysis, design and pracitce, Prentice Hall, 2005.

[2] M. Pavlić, Informacijski sustavi, Zagreb, Školska knjiga, 2011.

[3] Arthur M. Langer, Analysis and Design of Information Systems, Third Edition, Springer, 2008.

[4] Conceptual modeling of natural language functional requirements, Vidhu Bhala R. Vidya Sager, S. Abirami, The Journal of System and Software, 88 (2014), Elsevier

[5] Philip C. Jackson, Jr., Introduction to artificial intelligence, Dover Publications; Second Edition, Enlarged edition (June 1, 1985)

[6] A. Madureira et al. (eds.), Computational Intelligence for Engineering Systems: Emergent Applications, Intelligent Systems, Control and Automation: Science and Engineering 46, Springer Science + Business Media B.V. 2011.

[7] Klopper, R., Gruner, S., & Kourie, D. (2007), "Assessment of a framework to compare software development methodologies" Proceedings of the 2007 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries, 56-65.

[8] Alan Dennis, Systems Analysis and Design, Wiley, 2012.

[9] Maurizio De Tommasi, Angelo Corallo, SBEAVER, A Tool for Modeling Business Vocabularies and Bussiness Rules, Knowledge-Based Intelligent Information and Engineering Systems Lecture Notes in Computer Science Volume 4253, 2006, pp 1083-1091

[10] Roberta M. Roth; Alan Dennis; Barbara Haley Wixom, The System Analyst and Information System Development, John Wiley & Sons, 2012.

[11] Standish Group. (1994) The CHAOS Report. [Online]. http://www.standishgroup.com (29.06.2015.)

[12] Graham Curis, David Cobham; Business Information Systems, Prentice Hall, 2005.

[13] Shari, L. Phleeger, Software Engineering: Theory and Practice, Prentice-Hall, Inc, 1998.

[14] Rosziati Ibrahim, Siow Yen Yen; A Formal Model for Data Flow Diagram Rules, ARPN Journal of Systems and Software, May 2011.

[15] William S Davis; David C. Yen, The information system consultant's handbook, CRC press, 1999.

[16] Dennis, Wixom, Roth; System Analysis & Design, John Wiley & Sons, Inc. 2012.

[17] J.F.M. Burg, R.P. van de Riet., Analyzing Informal Requirements Specifications: A First Step towards Conceptual Modeling, Application of Natural Language to Information Systems, IOS Press, 1996.

[18] Derek Andrews, Formal Methods and Software Development, IEEE, 1996.

[19] Rosziati Ibrahim, Siow Yen Yen, A Fromal Model for Data Flow Diagram Rules, ARPN Journal of System and Software, Volume 1 No.2., May 2011.

[20] Peter Gorm Marsen, Nico Plat, Hans Toetenel, A Formal Semantics of Data Flow Diagrams, Formal Aspects of Computing, 1993.

[21] Maurice H. Halstead, Elements of Software Science, Elsevier North-Holland, Inc., New York, NY, 1977.

[22] Russel J. Abbot, Program Design by Informal English Descriptions, Communicationd of the ACM, 26(11):882-894, Nov 1983.

[23] M. Saeki, H. Horai, H. Enomoto, Software development process from natural language specification, In Proceedings of the 11th International Conference on Software Engineering (ICES-11): IEEE Computer Society Press, 1989.

[24] R. Carasik, S. Johnson, D. Patterson, and G. Von Glahn, Towards a domain description grammar: An application of linguistic semantics. ACM SIGSOFT Software Engineering Notes., 15(5):28-43, Oct 1990.

[25] D. Carver and D. Cordes, An object-oriented framework to support architectural design development, in System Sciences, Proceedings of the Twenty-Third Annual Hawaii International Conference on, vol. 2. Kailua-Kona, HI: IEEE, jan 1990,

[26] Börstler, Jürgen, User-centered requirements engineering in record-an overview, the Nordic Workshop on Programming Environment Research, Proceedings NWPER. Vol. 96. 1996.

[27] Beum-Seuk Lee, Barrett R. Bryant, Automation of Software System Development Using Natural Language Processing and Two-Level Grammar, Volume 2941 of the series Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg 2004.

[28] Peter Gorm Larsen, Nico Plat, Hans Toetenel: A Formal Semantics of Data Flow Diagrams, Formal Aspects of Computing, December 1994, Volume 6, Issue 6

[29] Ahmed Jilani, A. A., Nadeem, A., Kim, T. H. and Cho, E. S., Formal Representations of the Data Flow Diagram: A Survey. Proc. of the 2008 Advanced Software Engineering and Its Applications. Washington, USA: IEEE Computer Society

[30] Lucas, F.J., Molina, F. and Toval, A. (2009). A Systematic Review of UML Model Consistency Management. Information and Software Technology, 51(12)

[31] Tong, L. and Tang, C.S. (1991). Semantic Specification and Verification of Data Flow Diagrams. Journal of Computer Science and Technology, 6(1)

[32] Graham Curis, David Cobham; Business Information Systems, Prentice Hall, 2005.

[33] Boyd Nick, Using Natural Language in Software Development, http://www.educery.com/papers/rhetoric/road/, (14.06.2015.)

[34] Ambriola, V., Gervasi, V., 2006. On the systematic analysis of natural language requirements with CIRCE. Journal of Automated Software Engineering 13(January (1))

[35] Mu, Y., Wang, Y., Guo, J., 2009. Extracting software functional requirements from free text documents. In: International

Conference on Information and MultimediaTechnology, Jeju Island, South Korea.

[36] Yue, T., Briand, L., Labiche, Y., 2011. A systematic review of transformation approaches between user requirements and analysis models, 16. Springer: Requirements Engineering

[37] Elbendak, M., Vickers, P., Rossiter, N., 2011. Parsed use case descriptions as a basisfor object-oriented class model generation. Journal of Systems and Software 84(July (7))

[38] Vidhu Bhala R. Vidya Sager, S. Abirami, Conceptual modeling of natural language functional requirements, The Journal of System and Software, 88 (2014), Elsevier

[39] Alen Jakupović, Mile Pavlić, Zdravko Dovedan Han; Formalisation method for the text expressed knowledge

[40] Bjørner, D., Jones, C.B.: The Vienna Development Method: The Meta-Language. Springer-Verlag 1978.