Analiza troškova

Sanjin Maržić,
Ris d.o.o
sanjin.marzic@ris.hr

***Sažetak*—Ovaj rad opisuje područje analize troškova (engl. Spend Analysis, Spend Management), postojeću programsku podršku iz područja analize troškova i daje pregled istraživanja iz područja transformacije korisnikovih podataka, odnosno pronalaženja duplikata u bazi podataka što je jedan od početnih koraka u analizi troškova.**

***Ključne riječi*—analiza troškova; upravljanje troškovima; transformacija podataka; pronalaženje duplikata u bazi podatka**

I. Uvod

Uvođenje računala u poslovanje, razvoj informacijskih sustava, programskih jezika i baza podataka učinilo je poslovanje učinkovitijim. Omogućeno je brže, bogatije i točnije izvještavanje, automatizacija rada. S druge strane, na ovaj način prikupljala se i još uvijek se prikuplja sve veća količina različitih podataka (o poslovanju, nabavi, prodaji, kadrovima, primanjima, plaćanjima) koja skriva razne korisne informacije i znanja. Kako bi se otkrilo znanje u podacima potrebno je vršiti razne složene trenutne (*ad-hoc*) analize nad velikom količinom podataka. Ovakve analize nisu učinkovite na transakcijskom sustavu jer su takvi sustavi optimizirani za svakodnevne, jednostavne, preddefinirane operacije i izvještaje. Rješenje ovog problema pronađeno je u skladištu podataka (engl. *Data Warehouse*), odnosno periodičnom izvozu podataka u drugu bazu koja je optimizirana za složene analize. Prilikom izvoza podaci se obogaćuju podacima iz drugih izvora kako bi se mogle vršiti bolje analize.

Tijekom 1980-ih godina u SAD-u pojavila nova potreba —– analiza troškova [1]. Zahtjevi ove analize bili su drugačiji od zahtjeva za skladištenjem podataka jer su na drugačiji način promatrali podatke o dobavljačima i robi. Cilj analize troškova je smanjenje troškova analizom postojećih troškova, dobavljača, robe na koju se sredstva troše. Troškovi se smanjuju centraliziranom nabavom i racionalizacijom broja dobavljača. U ovom radu detaljnije se obrađuje područje analize i upravljanja troškovima.

Upravljanje troškovima (engl. *Spend management*) je način na koji tvrtke kontroliraju i optimiraju svoje troškove, uključuje rezanje operativnih i drugih troškova povezanih s poslovanjem [2]. U poduzećima upravljanje troškovima podrazumijeva upravljanje novcem tako na najučinkovitiji način kako bi se zadovoljile potrebe poslovanja (proizvodnja proizvoda, obavljanje usluga). Profit tvrtke je razlika između ukupnog troška i ukupnog prihoda te se može povećati ili smanjenjem troškova ili povećanjem prihoda. U doba krize tvrtke se često okreću smanjenju troškova zbog činjenice da je omjer prihoda i troškova 3:1, odnosno tri zarađene kune odgovaraju jednoj uštođenoj, odnosno ušteda od jedne kune na troškovnoj strani ima isti učinak kao i povećanje prihodovne strane za tri kune [2].

II. Analiza troškova

Analiza troškova (engl. *Spend Analysis*) je organizacija podataka o troškovima (uglavnom o nabavi) korištenjem hijerarhije dobavljača, sustava artikala i utrošenog iznosa za postizanje slijedećih ciljeva: (1) pronalaženje kategorija troškova, (2) pronalaženje prilika za stratešku nabavu okupljanjem zahtjeva za nabavom i racionalizacijom dobavljača i (3) smanjenje troškova većim popustima (u obliku rabata, ugovora i sl.) [1].

Pronalaženje kategorija troškova podrazumijeva kategorizaciju proizvoda i usluga (odnosno troškova) kako bi se moglo odgovoriti na pitanja oblika koliki je ukupni trošak na računalnu opremu ili na računalne monitore. Transakcijski sustavi najčešće sadrže isključivo podatke o konkretnoj kupljenoj robi npr. konkretan monitor, a za odgovor na pitanje iz primjera je potrebno prikupiti podatke o svim monitorima. Za ovakve analize potrebno je izraditi hijerarhijsku strukturu proizvoda (robe) kako bi se npr. za konkretan model računalnog monitora moglo reći da je to monitor određene veličine (npr. 17' monitor), LCD monitor, hardver i sl. Iz dobivene kategorizacije troškova mogu se vršiti analize između kategorija i unutar kategorije, npr. na koju kategoriju proizvoda otpada najveći trošak ili kolike su razlike u cijeni među artiklima iz iste kategorije. Također, ovakva kategorizacija može se koristiti za detekciju korupcije koja se očituje kao značajna razlika u cijeni unutar kategorije. Jedna moguća shema za kategorizaciju proizvoda i usluga je UNSPSC (Shema proizvoda i usluga Ujedinjenih naroda, engl. ***U**nited **N**ations **S**tandard **P**roducts and **S**ervices **C**ode*) [3]. Ova kategorizacija razvijena je kao jedinstvena shema proizvoda i usluga od strane Ujedinjenih naroda i tvrtke Duns & Brandstreet. Svakom proizvodu i usluzi dodjeljuje se osmoznamenkasti kôd u obliku SS-OO-RR-CC gdje je SS dvoznameknasta oznaka segmenta (engl. *Segment*), OO obitelji (engl. *Family*), RR razreda proizvoda (engl. *Class*) i CC robe (engl. *Commodity*). Važna svojstva UNSPSC sheme su: (1) jednostavnost prodiranja prema gore i dolje (engl. *Drill-down and Roll-up*) pomoću dijelova kôda, (2) konzistentnost koja osigurava da je pojedini element definiran na samo jednom mjestu, (3) potpunost jer sadrži sve proizvode i usluge kojima se može trgovati i (4) praćenje tržišta kako bi shema mogla obuhvatiti nove proizvode i usluge.

Okupljanje zahtjeva za nabavom znači okupljanje podataka o potrebama. Ovime se može postići povećanje pojedinačnih narudžbi i smanjenje ukupnog broja narudžbi (povezivanje više manjih narudžbi za istog dobavljača u veće, povezivanje narudžbi za slične proizvode izdanih u kraćem

periodu vremena). Sa većim narudžbama mogu se postići bolji uvjeti, odnosno niže cijene. **Racionalizacija dobavljača** podrazumijeva izradu hijerarhije dobavljača. Pomoću ove hijerarhije mogu se kategorizirati podatci o troškovima i podatci o nabavi te definirati željeni (i nepoželjni) dobavljači. Hijerarhija dobavljača znači definiranje skupova dobavljača koji su dio istog poslovnog sustava (npr. poslovne jedinice istog poslovnog sustava ili višestruki unosi istog dobavljača) ili koji su na neki drugi način povezani. Korištenjem hijerarhije dobavljača moguće je dobiti stvarne ukupne troškove vezane za nekog dobavljača (a ne npr. za poslovne jedinice istog dobavljača). Kombiniranjem ove hijerarhije sa hijerarhijom proizvoda može se doznati koliko različitih dobavljača dobavlja neku kategoriju proizvoda i koliki su ukupni troškovi po dobavljačima. Sa ovim podacima mogu se **smanjiti troškovi većim popustima** tako da se smanji broj dobavljača koji dobavlja neku kategoriju proizvoda. Time se povećava trošak prema istom dobavljaču čime se dobiva bolja pregovaračka pozicija za postizanje nižih cijena. Detektiranjem nepoželjnih dobavljača (po određenim kriterijima iz obogaćenih podataka o dobavljačima) i pregledom ukupnih troškova s tim dobavljačima troškovi prema njima se mogu smanjiti i preusmjeriti na druge dobavljače.

A. Kome je potrebna analiza troškova

Nemaju sve tvrtke potrebu za analizom troškova iz više razloga: programska rješenja za analizu troškova u pravilu su vrlo skupa što ih čini neprimjerenim za manje tvrtke, a takve tvrtke ni nemaju potrebu za analizom troškova jer su njihove nabave uglavnom malene i često centralizirane pa se eventualne analize mogu obavljati i bez složenih softvera. Tvrtke i organizacije kojima analiza troškova može biti zanimljiva su one koje su prostorno dislocirane, bez centralne nabave, sa većim brojem centara iz kojih se vrši nabava. Dakle, činjenica da je neka tvrtka velika i prostorno dislocirana nije dovoljna da bi joj bila potrebna analiza troškova: npr. lanac marketa koji ima velik broj poslovnica neće biti zainteresiran za analizu troškova jer takve tvrtke imaju centraliziranu nabavu, svi podaci o nabavi nalaze se na jednom mjestu. Narudžbe se već grupiraju tako da cijena nabave bude što niža. S druge strane npr. grad Rijeka može biti zainteresiran za analizu troškova jer u njemu djeluje veći broj tvrtki u vlasništvu grada koje imaju autonomiju u nabavi, ali se financiraju iz istog proračuna. Pošto svaka tvrtka samostalno vrši nabavu može se uočiti prostor za uštedu okupljanjem zahtjeva za nabavom. Također analizom kategorija troškova može se doći do uvida kolika je razlika u cijeni između srodnih artikala čime se može detektirati korupcija.

B. Uštede

Pitanje koje se postavlja je: kolike uštede se mogu očekivati uvođenjem analize troškova i u kojim segmentima poslovanja?

Tablica I prikazuje moguće uštede u postocima koje se mogu ostvariti po kategorijama ušteda. Sve navedene uštede postižu se na jedan od tri načina [1]:

- Strateški odabir dobavljača (engl. *strategic sourcing*) — pronalaženje strateških dobavljača i smanjenje broja dobavljača kako bi se ostvarili veći popusti na količinu i povećala kvaliteta (a kvaliteta se podiže odabirom pravog dobavljača)
- Smanjenje neplanirane potrošnje (engl. *maverick spend*) biranjem dobavljača (ugovorima) i boljim procesima (npr. interne odluke, naputci o nabavi)
- Izbjegavanje "curenja" troškova (engl. *spend leakage*) nadzorom pridržavanja ugovorenih odredbi.

Tablica I
MOGUĆE UŠTEDE [4]

Kategorija	Uštede
Sirovine	2–5% (uz bolje upravljanje rizicima)
Pakiranje	10–20%
Indirektni troškovi vezani uz materijale i usluge	10–20%
Informacijska tehnologija	15–30%
Profesionalne usluge	8–15%
Kapitalni projekti	7–15%
Ostali indirektni troškovi	5–15%
Mediji, marketing, promotivni materijali	10–20%
Logistika i transport	7–15%

Ukupni trošak na sirovine	1,000,000,000
Ušteda 5%	50,000,000
Indirektni troškovi, usluge	500,000,000
Ušteda 15%	75,000,000

Slika 1. Primjer mogćih ušteda [4]

Treća točka je osobito važna jer se dio ugovorenih ušteda može izgubiti bez mehanizama nadzora poštivanja odredbi. Sve tri točke potrebno je neprestano ponavljati kako bi se ostvarivale stalne uštede.

Postoci iz tablice I odnose se na udio u troškovima po kategorijama, za konkretne iznose potrebni su ukupni iznosi. Zbog prikaza u postocima prva stavka može na pri pogled izgledati kao neznatna ušteda, ali činjenica je da je upravo ukupni trošak na sirovine najveća stavka u ukupnim troškovima (proizvodnog) poslovnog sustava koja može iznositi i veliku većinu ukupnog troška. Svi ostali troškovi mogu se promatrati kao indirektni, ali ih ne treba zanemariti jer su na njima moguće najveće uštede. Može se pomisliti kako je manja ušteda na sirovinama puno značajnija od svake uštede na svim ostalim stavkama, ali to nije uvijek tako, što je prikazano u primjeru na slici 1.

C. Konkretne prednosti analize troškova

Programska podrška za analizu troškova omogućava pregled podataka o troškovima za cijeli poslovni sustav i smanjuje potrebu za ručnim obradama vezanim uz izvještaje i trendove.

Konkretne prednosti [1]:

- Vidljivost svih troškova poslovnog sustava
- Značajno poboljšanje točnosti i konzistentnosti podataka
- Poboljšanje kvalitete i dubine analiza s vremenom

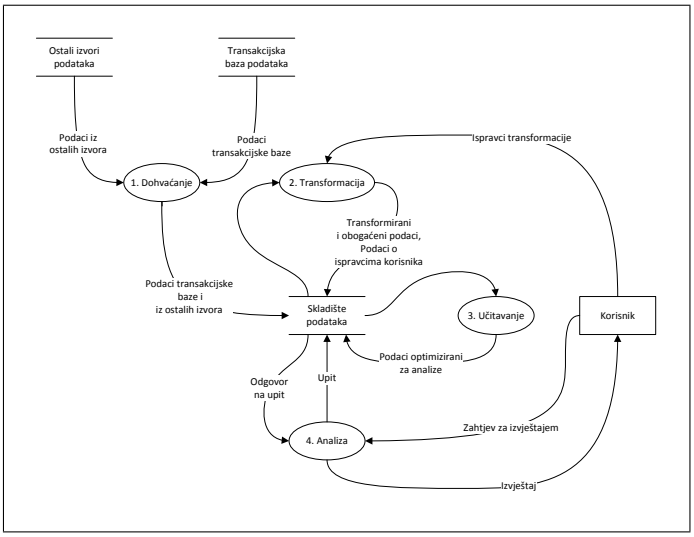
- Smanjenje vremena potrebnog za izradu nestandardnih izvještaja sa do 2 tjedna na gotovo trenutačno
- Smanjenje troškova mimo ugovora kako bi se povećale uštede
- Nestanak potrebe za stalno zaposlenim stručnjacima iz područja dohvaćanja (ekstrakcije) podataka koji bi pročišćavali podatke i izrađivali nestandardizirane izvještaje
- Smanjenje administrativnih troškova obrade i provjere narudžbi
- Brzo pronalaženje i stvaranje liste prioriteta najvećih prilika za uštede
- Povećanje uštede smanjenjem broja dobavljača i velikim narudžbama
- Dodatne uštede identifikacijom prilika za sklapanje ugovora
- Eliminacija preplaćivanja dobavljača i garancija realizacije rabata
- Kontinuirano povećanje poštivanja dogovorenih uvjeta i uštede praćenjem stvarnih i ugovorenih ušteda
- Smanjenje ovisnosti nabave o IT stručnjacima koji dostavljaju podatke o troškovima
- Uklanjanje neslaganja oko kvalitete podataka uvođenjem "jedne istine" za sve troškove.

III. Proces analize troškova

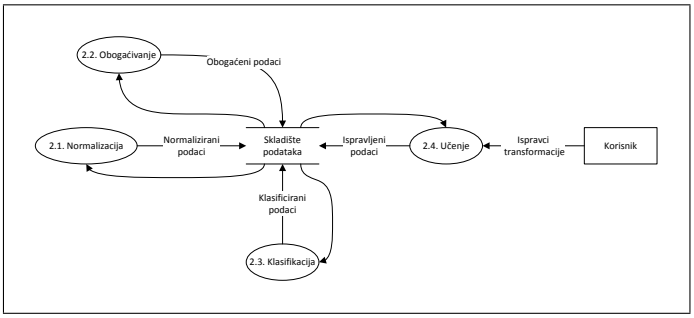
Analiza troškova kao proces slična je procesu skladištenja podataka: oboje se temelje na ETLA procesu. Faze ETLA procesa su dohvaćanje (engl. ***E**xtraction*), transformacija (engl. ***T**ransformation*), učitavanje (engl. ***Loading***) i analiza (engl. ***A**nalyze*) podataka.

Dohvaćanje podataka podrazumijeva povremeno (periodično) dohvaćanje podataka iz transakcijskog sustava, ali i svih ostalih izvora podataka u kojima se nalaze korisni podaci, kako bi analize bile aktualne. Podaci se ne moraju dohvaćati isključivo iz baza podataka, neki korisni podaci mogu dolaziti iz različitih izvora (npr. korisničke datoteke, internet).

Transformacija podataka je najvažniji dio procesa analize troškova (uz analizu). Njime se stvaraju preduvjeti za analizu. Kvaliteta rezultata analize ovisi o kvaliteti transformacije podataka. U ovom koraku podaci iz transakcijskog sustava se normaliziraju tako da su svi zapisani korištenjem istih mjernih jedinica (pretvaranje valuta, mjera veličine) i u istom formatu (npr. cijela adresa u jednom polju ili u više polja (grad, poštanski broj, ulica, kućni broj)), stvara se hijerarhija proizvoda i dobavljača, rješavaju višestruki unosi, čiste podaci o poslovanju (transakcijama). U postupak transformacije podataka spada i obogaćivanje podataka što podrazumijeva dopunjavanje podataka koji nedostaju, korištenje vanjskih izvora (npr. podaci o poslovanju dobavljača, stanju na burzama). Kako ne postoji savršen algoritam za klasifikaciju (dobavljača, proizvoda) koji radi bez grešaka sustav mora omogućiti korisničku intervenciju kod transformacije podataka i iz te intervencije mora učiti kako bi kod slijedeće klasifikacije algoritam povećao točnost i kako bi se s vremenom smanjivala potreba za korisničkom intervencijom, odnosno da korisnik mora brinuti samo za nove iznimke.



Slika 2. Dijagram procesa analize troškova



Slika 3. Dijagram procesa – 2. Transformacija podataka

Korak **učitavanja** podataka podrazumijeva učitavanje transformiranih podataka u ciljnu bazu koja je optimizirana za postupke analize.

Analiza je korak u kojem se na temelju podataka stvaraju izvještaji, bilo predefinirani bilo trenutni (*ad-hoc*). Analiza, kao i kod skladišta podataka, mora omogućavati *slicing&dicing*, pregled podataka po više dimenzija unutar kojih je moguć pogled sa više razina hijerarhije. Kod analize važne su i mogućnosti vizualizacije podataka koje imaju dobre performanse nad velikim skupom podataka.

Slika 2 prikazuje model opisanog procesa analize troškova (nacrtanog po metodi DTP [5]), a slika 3 prikazuje model podprocesa transformacije podatka.

IV. Transformacija korisnikovih podataka

Postupak transformacije korisnikovih podataka je, kako je ranije rečeno, ključni dio procesa analize troškova. Najteži zadatak transformacije, odnosno čišćenja korisnikovih podataka, je klasifikacija. Ovaj postupak uključuje pronalaženje i sparivanje duplikata među podacima iz korisnikove baze. Duplikati u korisnikovoj bazi nastaju iz više razloga: zbog grešaka pri unosu, nedostatka ograničenja u bazi, nedostatka jedinstvenog identifikatora, različitih načina zapisa [6]. Problem duplikata se osobito očituje kod integracije podataka iz više izvora, što je slučaj kod analize troškova. Problemi do kojih dolazi

kod integracije podataka iz više izvora zajednički se nazivaju heterogenost podataka. Heterogenost podataka može biti strukturalna i leksička [6]. Strukturalna heterogenost podataka se odnosi na drugačiju strukturu podataka (npr. jedan izvor pohranjuje adresu u jednom polju, a drugi u više polja–ulica, kućni broj, dodatak broju). Leksička heterogenost se odnosi na različite zapise koji predstavljaju isti entitet (npr. T-Com i Hrvatski Telekom).

Cilj pronalaženja duplikata u bazi podataka je pronaći zapise, iz istog ili različitih izvora, koji se odnose na isti entitet u stvarnosti, naravno zapisi ne moraju biti jednaki [6].

Prije primjena metoda pronalaženja duplikata potrebno je provesti postupak normalizacije podataka čime se osigurava da su svi podaci zapisani na jednak način čime se djelomično rješava problem strukturalne heterogenosti. Ova faza uključuje parsiranje, transformaciju i standardizaciju podataka čime se svi podaci dovode u oblik pogodniji za kasnije usporedbe [6]. Zadatak **parsiranja** je pronaći pojedine elemente u izvornim podacima. Ovime se olakšavaju daljnje obrade jer je moguće uspoređivati odgovarajuće dijelove podataka, umjesto većih znakovnih nizova. Primjer parsiranja je razdvajanje naziva na ime i prezime ili adrese na ulicu i kućni broj. **Transformacija** podataka se odnosi na promjene pojedinih atributa, neovisno o vrijednosti drugih atributa. Ove promjene uključuju promjenu tipa podatka za atribut, preimenovanje atributa, promjenu kodne stranice, provjeru raspona vrijednosti (engl. *range checking*, npr. datum narudžbe ne smije biti u budućnosti). U transformaciju spada i provjera ovisnosti (engl. *dependency checking*) koja uspoređuje vrijednosti dva (ili više) atributa kako bi se osigurala osnovna konzistentnost podataka (npr. datum računa ne smije biti manji od datuma narudžbe ili veći od datuma uplate). **Standardizacija** podataka dovodi podatke pojedinih atributa u odabrani zajednički format. Npr. pošto ne postoji globalni standard zapisa adrese ona može biti zapisana na mnogo različitih načina (npr. riječ „Ulica“ se može izostaviti ili zapisati kraticom „Ul.“, naziv ulice se može skratiti, u adresi se može pojaviti i podatak o katu, broju stana ili poštanskom pretincu, naziv grada se može zapisati sa ili bez poštanskog broja) pa se elementi pronađeni parsiranjem zapisuju na standardiziran način (standard određuje od kojih elemenata će se adresa sastojati, u kojem obliku i kojim redoslijedom) kako bi usporedbe mogle biti uspješnije.

A. Usporedba znakovnih nizova

Nakon normalizacije podataka oni su zapisani u obliku pogodnijem za otkrivanje duplikata, no iako je problem pojednostavljen ovaj zadatak i dalje nije jednostavan. Kako većina duplikata u bazi podataka nastaje kao posljedica tipkarskih grešaka u znakovnim poljima tako se i većina metoda za pronalaženje duplikata temelji na tehnikama usporedbe znakovnih nizova (engl. *string*). Do grešaka može doći i u poljima s drugim vrstama podataka (numeričkim), no istraživanja ovog problema su još u početcima. Zbog toga se kod određivanja sličnosti numeričkih vrijednosti najčešće primjenjuju metode korištene za usporedbu znakovnih nizova, odnosno brojevima se pristupa kao nizu znamenki.

Pristup problemu usporedbe, odnosno određivanja sličnosti, znakovnih nizova može se podijeliti na pristup temeljen na znakovima (slovima), pristup temeljen na *tokenima* (engl. *token*, jedinice koje se sastoje od više znakova), pristup temeljen na fonetskoj sličnosti.

Sličnost na temelju znakova (slova) je osmišljen za rješavanje problema tipkarskih grešaka. Najpoznatiji algoritam iz ove skupine je **Levenshteinov algoritam** [6] kojim se određuje udaljenost između znakovnih nizova (engl. *Levenshtein distance*, *Edit distance*). Udaljenost između dva izraza se po Levenshteinovom algoritmu definira kao najmanji broj operacija (sa jednim znakom) potrebnih za pretvaranje jednog niza u drugi. Dozvoljene operacije su umetanje, brisanje i zamjena, svaka operacija ima težinu 1. Ovaj algoritam se pokazao dobrim kod tipkarskih grešaka i kod usporedbi izraza koji se sastoje od jedne riječi, ali za druge vrste duplikata nije učinkovit.

Jedan od glavnih problema koje Levenshteinov algoritam ne rješava je korištenje kratica u jednom od izraza jer udaljenost raste za svaki znak razlike. Ovaj problem rješava **algoritam susjednih praznina** [6] (engl. *affine gap distance*) tako da Levenshteinovom algoritmu dodaje dvije operacije: početna praznina i dodatna praznina. Dodatna praznina obično ima manju težinu od početne čime se postižu manje udaljenosti za dodavanje veći broj praznina kod transformacije znakovnog niza što je slučaj kod korištenja kratica ili dodane riječi.

Smith-Watermanov algoritam nadograđuje Levenshteinov algoritam i algoritam susjednih praznina tako da se daje veća težina razlikama u sredini znakovnog niza, a manja razlikama na početku [6]. Ovaj algoritam je pogodan za usporedbu naziva osoba u slučajevima kada imenu i prezimenu osobe može biti dodana i titula (bilo na početku ili kraju).

Za usporedbu imena i prezimena osmišljeni su Jarov i **Jaro-Winklerov algoritam**. Jarova sličnost zapisa s_1 i s_2 se računa na slijedeći način [7]:

1) Određuje se duljina oba znakovna niza, $|s_1|$ i $|s_2|$
2) Pronalaze odgovarajući znakovi (engl. *common characters*) u oba niza i broj odgovarajućih znakova c. Odgovarajući znakovi su definirani kao:

$$s_1[i] = s_2[j] \quad |i - j| \leq \frac{1}{2} \min\{|s_1|, |s_2|\}$$

   Odnosno, to su jednaki znakovi čija je razlika pozicija u nizu manja ili jednaka polovici duljine kraćeg niza
3) Određuje se broj transpozicija t tako da se i-ti odgovarajući znak iz prvog niza uspoređuje s i-tim znakom iz drugog niza, svaki par nejednakih znakova je jedna transpozicija

Jarova sličnost se računa kao:

$$J(s_1, s_2) = \begin{cases} 0 & c = 0 \\ \frac{1}{3}\left(\frac{c}{|s_1|} + \frac{c}{|s_2|} + \frac{c - \frac{t}{2}}{c}\right) & c \neq 0 \end{cases}$$

Svojstva Jarove sličnosti su da joj je vrijednost u intervalu $[0, 1]$, 0 odgovara potpuno različitim nizovima, a 1 potpuno jednakim.

Winkler [7] je Jarov algoritam učinio osjetljivijim na razlike u prefiksu znakovnog niza tako da se uz izračun Jarove udaljenosti računa dodatno uspoređuje prefiks (najčešće prva 4 znaka) i određuje se broj zajedničkih znakova l. Određuje se

i koeficijent p kojim se određuje koliko će zajednički prefiks utjecati na sličnost. Jaro-Winklerova udaljenost se računa kao:

$$JW(s_1, s_2) = J(s_1, s_2) + (l \cdot p\,(1 - J(s_1, s_2)))$$

Winkler je za koeficijent p uzimao vrijednost 0.1, ovaj koeficijent mora imati vrijednost manju ili jednaku $\frac{1}{duljina\ prefiksa}$ kako bi udaljenost sačuvala svojstvo da poprima vrijednosti iz intervala $[0, 1]$ (ako se promatra prefiks duljine 4, mora vrijediti $p \leq 0.25$).

Još jedna metoda za određivanje sličnosti znakovnih nizova je usporedba q-grama [6]. Q-gram je podniz (engl. *substring*) duljine q, q-grami se određuju pomicanjem "prozora" (engl. *sliding window*) duljine q po izvornom nizu za po jedan znak. Npr. riječ "analiza" se može podijeliti na 5 q-grama duljine 3 (ana-nal-ali-liz-iza). Pretpostavka ove metode je da slični nizovi sadrže velik broj zajedničkih q-grama. Često se na početak niza dodaje $(q - 1)$ znak koji nije dio abecede od koje se sastoji niz kako bi se dala veća važnost znakovima na početku niza jer bi inače znakovi na početku niza bili sadržani u manjem broju q-grama (u primjeru se prvo slovo "a" pojavljuje u jednom 3-gramu, a drugo slovo "a" u 3). Najčešće se koriste trigrami $(q = 3)$, bigrami $(q = 2)$ i unigrami $(q = 1)$.

Slična metoda usporedbi q-grama je usporedba n-grama opisana u [8] koja generalizira Levenshteinov algoritam tako da se umjesto pojedinih znakova (slova) uspoređuju nizovi duljine n. Glavna razlika n-gramske sličnosti u odnosu na usporedbu q-grama je što se u obzir uzima poredak n-grama unutar izraza koji se uspoređuju.

Do sada spomenuti algoritmi su dobri za usporedbu znakovnih nizova koji se razlikuju zbog tipkarske greške, korištenja kratice ili umetnute dodatne riječi. Problem kojeg ne rješavaju je različit poredak riječi. Ovaj problem rješavaju algoritmi temeljeni na tokenima. **Usporedba atomičkih nizova** [6] dijeli niz (izraz) na atomične nizove (riječi) koji su razdijeljeni (engl. *delimited*) interpunkcijskim znakovima. Dva atomična niza su odgovarajuća ako su jednaki ili ako je jedan niz prefiks drugome. Sličnost se određuje kao omjer odgovarajućih atomičnih nizova i prosjeka broja atoma u nizovima koji se uspoređuju.

Agoritam WHIRL [9] koristi mjere iz područja otkrivanja informacija iz teksta (engl. *information retrieval*), konkretno TF-IDF. TF-IDF (engl. *term frequency-inverted document frequency*) je vrijednost koja određuje težinu (značaj) pojedine riječi, definira se kao: $\log(tf_w + 1) \cdot \log(idf_w)$ gdje je tf_w broj pojavljivanja riječi w u konkretnom zapisu, a idf_w je omjer ukupnog broja zapisa n i broja zapisa koji sadrže promatranu riječ n_w. Vrijednost TF-IDF je veća što je riječ rjeđa i što se više puta pojavljuje u promatranom izrazu. Sličnost dvaju izraza se određuje kao

$$\frac{\sum_{j=1}^{n} v_{s_1}(j) \cdot v_{s_2}(j)}{\sum_{j=1}^{n} \left(v_{s_1}(j)\right)^2 \cdot \sum_{j=1}^{n} \left(v_{s_2}(j)\right)^2}$$

Ovaj pristup rješava problem različitog poretka riječi, ali nije otporan na greške u zapisima pa i najmanja razlika između riječi rezultira sa sličnošću 0. Ova mana se može riješiti *SoftTF-IDF* pristupom [6] u kojem se u obzir uzimaju i slične riječi (sličnost se računa po jednom od ranije opisanih algoritama) tako da se umnožak težina sličnih riječi množi sa sličnošću (koja je vrijednost u intervalu $[0, 1]$). Drugo moguće rješenje je korištenje q-grama umjesto riječi: TF-IDF vrijednosti se računaju za q-grame, a riječi unatoč tipkarskim greškama dijele veći broj zajedničkih q-grama.

Treća skupina metoda za određivanje sličnosti znakovnih nizova temelji se na fonetskoj sličnosti, odnosno zvučenju izraza i riječi. Ovaj pristup rješava probleme riječi koje nisu slične na razini znakova ili tokena od kojih se sastoje, ali zvuče slično (npr. *Kageonne* i *Cajun*). Najpoznatiji algoritam iz ove skupine je Soundex [6] koji svaki izraz kodira po zvučnosti i uspoređuje dobivene kodove. Algoritam je:

1) Prvo slovo izraza se sadržava, uklanjaju se sva pojavljivanja slova H i W
2) Suglasnici se zamjenjuju znamenkama po pravilu:
   - B, F, P, V $\rightarrow$ 1
   - C, G, J, K, Q, S, X, Z $\rightarrow$ 2
   - D, T $\rightarrow$ 3
   - L $\rightarrow$ 4
   - M, N $\rightarrow$ 5
   - R $\rightarrow$ 6
3) Sva uzastopna ponavljanja iste znamenke se zamjenjuju jednom znamenkom, samoglasnici i Y služe kao separatori, znamenke između koji je jedan od ovih slova se ne smatraju susjednim
4) Uklanjaju se samoglasnici i Y
5) Zadržava se prvo slovo izraza i prve tri znamenke, ostale znamenke se odbacuju. U slučaju kada nema dovoljno znamenki dodaju se nule na kraj

Opisani algoritam se još zove Američki Soundex (engl. *American Soundex*) jer je prilagođen engleskom jeziku. Za germanske i slavenske jezike definiran je Daitch-Mokotoff Soundex [10]. Algoritam za kodiranje je sličan s drugačijim pravilima za zamjenu slova brojkama, a rezultirajući kôd izraza se sastoji od 6 umjesto 3 znamenke, te se i prvo slovo pretvara u broj. Soundex se pokazao najkorisnijim kod usporedbe imena i prezimena, odnosno pretraživanju baze po imenu ili prezimenu osobe.

Algoritam NYSIIS [11] (engl. New York State Identification and Intelligence System) je još jedan algoritam za usporedbu izraza po zvučnosti, ali za razliku od Soundexa pamti i položaje samoglasnika i ne zamjenjuje slova brojkama već drugim slovima. NYSIIS se, kao i Soundex, najviše primjenjuje na usporedbu imena i prezimena. Postupak dobivanja kôda izraza za NYSIIS je sličan Soundexovom, može se naći u [11]. NYSIIS je postigao točnost od 98.72% kod pronalaženja prezimena u bazi Američke savezne države New York, dok je Soundex bio 2,73% lošiji [6].

B. Pronalaženje duplikata u bazi podataka

Kod pronalaženja duplikata u bazi podataka algoritmi iz prethodnog odjeljka se koriste za usporedbu pojedinih atributa zapisa, no nakon dobivenih rezultata usporedbe potrebno je par zapisa označiti kao duplikate, odnosno neduplikate. Postoje dva osnovna pristupa ovom problemu: (1) korištenje skupa za treniranje (učenje) u kojem se nalaze zapisi već označeni

kao duplikati i neduplikati i (2) korištenje znanja iz domene problema ili općih mjera udaljenosti za označavanje zapisa.

Ako je kod označavanja duplikata dostupan skup za treniranje moguće je koristiti metode nadziranog učenja i statističke metode za definiranje modela.

Najjednostavniji statistički pristup je korištenje Bayesovog zaključivanja, odnosno korištenje Bayesovog teorema [6]. Kao ulaz funkciji za određivanje jesu li zapisi A i B duplikati koristi se vektor $\mathbf{x}$ čiji su elementi sličnosti parova odgovarajućih polja iz ovih zapisa, a izlaz je oznaka klase, duplikat (D) ili neduplikat (N).

Uz pretpostavku da je vektor sličnosti slučajni vektor čija je funkcija gustoće različita za svaku klasu i uz poznate funkcije gustoće za klase moguće je izravno primijeniti Bayesovu formulu. Naravno, cilj je da klasifikator ima što manju grešku pa se zapisu dodjeljuje klasa s najvećom vjerojatnosti, odnosno:

$$\langle A, B\rangle \in \begin{cases} D & p(D|\mathbf{x}) \geq p(N|\mathbf{x}) \\ N & p(D|\mathbf{x}) < p(N|\mathbf{x}) \end{cases}$$

Primjenom Bayesove formule dobiva se:

$$\langle A, B\rangle \in \begin{cases} D & \frac{p(\mathbf{x}|D)}{p(\mathbf{x}|N)} \geq \frac{p(N)}{p(D)} \\ N & \frac{p(\mathbf{x}|D)}{p(\mathbf{x}|N)} < \frac{p(N)}{p(D)} \end{cases}$$

Problem ovog klasifikatora je što su distribucije vjerojatnosti iz funkcije odlučivanja najčešće nepoznate. Čest pristup izračunu koeficijenata $p(\mathbf{x}|D)$ i $p(\mathbf{x}|N)$ je Naivni Bayes (engl. *Naive Bayes*). Ovaj pristup se temelji na pretpostavci neovisnosti vjerojatnosti $p(x_i|D)$ i $p(x_j|D)$, gdje su x_i i x_j elementi vektora $\mathbf{x}$. Tada vrijedi:

$$p(\mathbf{x}|D) = \prod_{i=1}^{n} p(x_i|D) \qquad p(\mathbf{x}|N) = \prod_{i=1}^{n} p(x_i|N)$$

Ove vrijednosti se mogu izračunati iz skupa za treniranje. No, čak i ako skup za treniranje nije dostupan vrijednosti je moguće procijeniti korištenjem binarnih vrijednosti za x_i iz parova zapisa koji s visokom vjerojatnošću pripadaju jednoj, odnosno drugoj, klasi.

Kod klasificiranja para zapisa kao duplikat ili neduplikat minimizacija broja grešaka nije uvijek najbolji izbor jer u određenim slučajevima različite greške imaju različite posljedice (težine). Zbog toga se u funkciju odlučivanja dodaje i cijena greške c_{ij} koja označava težinu greške dodjeljivanja klase i kada je stvarna klasa j. Određuju se očekivane cijene grešaka klasifikacije para kao duplikata r_D, odnosno neduplikata r_N:

$$r_D(\mathbf{x}) = c_{DD} \cdot p(D|\mathbf{x}) + c_{DN} \cdot p(N|\mathbf{x})$$

$$r_N(\mathbf{x}) = c_{ND} \cdot p(D|\mathbf{x}) + c_{NN} \cdot p(N|\mathbf{x})$$

Funkcija odlučivanja je tada:

$$\langle A, B\rangle = \begin{cases} D, & r_D(\mathbf{x}) \leq r_N(\mathbf{x}) \\ N, & r_D(\mathbf{x}) > r_N(\mathbf{x}) \end{cases}$$

Još jedna dorada funkcije odlučivanja je dodavanje treće mogućnosti — nedonošenje odluke [6] (engl. *reject*). Ova mogućnost se koristi u slučajevima kada je cijena obje odluke visoka tako da se odredi prag odbijanja (engl. *reject region*) u kojem će se kao vrijednost klase odrediti vrijednost "nema odluke" te će se par proslijediti na odlučivanje ekspertu (čovjeku).

U slučajevima kada je dostupan skup za treniranje najkorištenije su nadzirane klasifikacijske metode strojnog učenja. Za pronalaženje duplikata korištena su stabla odlučivanja, linearno diskriminacijski algoritmi i kvantizacija vektora (generalizacija algoritma KNN) [6]. Pokazalo se da stabla odlučivanja imaju najmanji postotak pogreške za problm klasifikacije duplikata u odnosnu na druge primijenjene metode [6]. U ovim pristupima svakom paru duplikata se pristupa neovisno o ostalim, to znači da će algoritam za par zapisa (A, B) donijeti odluku jesu li ili ne duplikati neovisno o tome da li su drugi parovi već proglašeni duplikatima, odnosno neduplikatima, iz čega se može zaključiti klasa para (A, B).

Nakon primjene klasifikatora, bilo da se radi o Bayesovom klasifikatoru ili klasifikatoru dobivenom nekom od metoda strojnog učenja, dobivene podatke je potrebno dodatno obraditi (engl. *post-processing*) kako bi se u obzir uzela tranzitivnost duplikata. Moguća situacija je da se primjenom algoritma parovi zapisa (A, B) i (B, C) proglase duplikatima, a par (A, C) neduplikatom. Skup koji sadrži ovakve proturječnosti potrebno je dovesti u stanje da je količina ovakvih problema što manja: ako se označeni skup promatra kao graf gdje su čvorovi zapisi, a veze su oznake (duplikat ili neduplikat) tada je rješenje ovog problema dijeljenje grafa na podgrafove koji imaju svojstvo da su im svi čvorovi povezani oznakom duplikat.

Jedan od uzroka nastanka proturječnog skupa duplikata je drugačija reprezentacija iste logičke vrijednosti, pa je prijedlog rješenja problema korištenje parova vrijednosti atributa iz zapisa koji su proglašeni duplikatima kao sinonima čime se mogu poboljšati rezultati usporedbe. Pretpostavka je da, pošto se duplikati odnose na isti entitet u realnom sustavu da vrijednosti atributa također predstavljaju istu logičku vrijednost [6].

Tehnike koje se temelje na aktivnom učenju (engl. *active learning based techniques*) mogu napraviti početni model na relativno malenom početnom skupu za učenje. Problem skupa za učenje je činjenica u njemu najčešće nedostaju rubni slučajevi, a uglavnom se pojavljuju parovi zapisa koji su izraziti duplikati i izraziti neduplikati. Pomoću početnog modela se u skupu pronalaze ovakvi zapisi te im se dodjeljuje oznaka klase, a među ostalim, neklasificiranim, parovima se traže oni koji će, kada budu klasificirani, donijeti najveći doprinos modelu, odnosno budućim klasifikacijama [6].

U slučajevima kada skup za učenje nije dostupan do sada navedene metode i tehnike se ne mogu koristiti. Jedno od rješenja u ovakvim slučajevima su tehnike temeljene na udaljenosti. U ove tehnike spada spajanje cijelog sloga u jedan veliki znakovni niz (engl. *string*) te usporedba s drugim takvim nizovima korištenjem jednog od algoritama za usporedbu znakovnih nizova. Naravno, udaljenost se može odrediti i usporedbom parova polja, no onda je potrebno odrediti težinu svake dobivene udaljenosti kako bi se odredila ukupna udaljenost te odrediti prag vrijednosti za koje će se par zapisa proglasiti duplikatima.

Drugi mogući pristup je korištenje pravila za određivanje da li je par duplikat ili nije. Pravila se definiraju na teme-

lju ekspertnog znanja iz domene problema. Pravila najčešće odražavaju (moguće) scenarije nastanka duplikata ili rješavaju greške koje mogu dovesti do nastanka duplikata. Primjer takvog rješenja je opisan u [12] gdje je korišten klasifikator koji se sastojao tri vrste pravila – pozitivnih (koje potvrđuju duplikat), negativnih (koja eliminiraju par kao siguran neduplikat) i pravila temeljenih na sličnosti.

U slučaju kada skup za učenje nije dostupan mogu se koristiti tehnike nenadziranog učenja (grupiranje). Grupiranje se koristi kako bi se smanjio opseg ručnog označavanja duplikata tako da grupira parove iz baze na temelju vrijednosti atributa (vektor značajki). Pretpostavka ove metode je da slični vektori pripadaju istoj klasi [6].

C. Učinkovitost pronalaženja duplikata

Učinkovitost pronalaženja duplikata u bazi podataka ovisi o dva faktora: (1) broju potrebnih usporedbi i (2) složenosti jedne usporedbe.

Za skup sa n zapisa broj mogućih usporedbi unutar ovog skupa je $\frac{n\cdot(n-1)}{2}$, a pošto ova vrijednost brzo raste s porastom broja zapisa usporedba svih zapisa međusobno nije izvediva (npr. za $n = 5.000$ broj potrebnih usporedbi je $12.500.000$). No, iako je broj svih mogući usporedbi vrlo velik, većina od tog broja usporedbi nije potrebna jer većina tih parova nisu duplikati.

Broj usporedbi može se smanjiti korištenjem blokova [6] (engl. *blocking*): za svaki zapis u bazi određuje se vrijednost *hash* funkcije. Ove vrijednosti se koriste za određivanje grupa zapisa, jednaki zapisi će se nalaziti u istoj grupi, no to ne mora vrijediti za slične zapise. Primjenom funkcije kao što je Soundex ili Metaphone na polje čija vrijednost ima visok utjecaj na odluku radi li se ili ne o duplikatu te uspoređivanjem zapisa koji imaju slične vrijednosti funkcije mogu se pronaći i duplikati koji nisu jednaki. Glavna prednost korištenja blokova je činjenica da se pronalaženje duplikata može obaviti sa značajno manjim brojem usporedbi, ali to je i njegova glavna mana: s manjim brojem usporedbi mnogi rezultati se mogu i izgubiti (npr. zbog greške u polju na koje se primjenjuje funkcija). Rješenje ovog problema je višestruka primjena algoritma s drugačijom funkcijom za definiranje blokova čime vrijeme izvođenja ne raste značajno, ali se točnost može značajno poboljšati.

Slična metoda je metoda uređenog susjedstva (engl. *Sorted Neighbourhood Method*, SNM). U ovoj metodi za svaki zapis se određuje ključ, odnosno vrijednost izračunata na temelju značajnih polja zapisa te se zapisi sortiraju po ključu. Svaki zapis se uspoređuje samo s n susjednih zapisa (u sortiranoj listi). Ova metoda ima istu manu kao i prethodna, a to je da se u jednom prolazu može propustiti usporediti velik broj duplikata što se rješava višestrukim prolazima s različito definiranim ključevima.

Metoda korištenja grupa (engl. *Clustering and Canopies*) se temelji na ideji da je relacija "je duplikat" tranzitivna, pa nije potrebno vršiti usporedbe sa svim elementima grupe duplikata, nego samo s jednim predstavnikom. Još jedan način smanjivanja broja usporedbi je određivanje brze usporedbe kojom se parovi zapisa grupiraju u grupe koje ne moraju biti disjunktne (engl. *canopy*). Nakon toga se unutar grupa zapisi uspoređuju složenijim usporedbama za određivanje radi li se ili ne o duplikatima.

Navedene metode koriste se za smanjenje ukupnog broja usporedbi, drugi način ubrzavanja pronalaženja duplikata je ubrzanje jedne usporedbe. Kako se zapisi uspoređuju tako da se računa sličnost parova polja, obrada se može ubrzati tako da se brzo odredi da se ne radi o duplikatu npr. izračunavanjem i ispitivanjem vrijednosti manjeg skupa atributa čime se izbjegava računanje ostalih sličnosti.

V. Programska podrška za analizu troškova

Prvi koraci u analizi troškova dogodili su se 1980-ih godina, a prvi računalni programi specijalizirani za analizu troškova pojavili su se početkom 2000-ih.

A. Ariba

Ariba [13] je njemački proizvođač softvera osnovan 1996. godine specijaliziran za izradu aplikacija za analizu troškova, upravljanje ugovorima, dobavljačima i sl. Aplikacija za analizu troškova zove se *Ariba Spend Analysis*. Ova aplikacija izrađena po modelu softvera kao usluge (engl. *Software as a service*, *SaaS*). Tvrtku Ariba u 2012. godini kupila je tvrtka SAP.

Dohvaćeni podaci obogaćuju se iz baze *Dun & Bradstreet* (D&B). To je američka kompanija koja prikuplja podatke o tvrtkama za potrebe odlučivanja o kreditiranju, upravljanja lancem dobavljača. Podaci se obogaćuju i podacima vezanim uz ekologiju poslovanja, odnosima među tvrtkama, rizikom.

Svojstva aplikacije koja proizvođač ističe su da omogućuje usporedbu sa srodnim tvrtkama iz iste branše. Aplikacija omogućuje predviđanja i analize mogućnosti, klasifikacije (artikala i usluga, dobavljača) po preddefiniranim ili vlastitim shemama. Ariba svakom klijentu dodjeljuje upravitelja projekta.

B. Bravo Solution

BravoSolution [14] je talijanski proizvođač softvera iz domene upravljanja troškovima. Aplikacija BravoSolution Spend Management je njihova aplikacija za upravljanje troškovima. Aplikacija je, kao i Aribino rješenje, realizirana po SaaS modelu.

Svojstva koja proizvođač ističe su da aplikacija za klasifikaciju podataka o troškovima koriste više klasifikacijskih shema, omogućena je i izrada korisničke klasifikacijske sheme. Troškovi se klasificiraju na temelju na temelju pravila i klasifikatorima dobivenim nadziranim učenjem na temelju prošlih odluka. Kod klasifikacije dobavljača koriste se podaci iz D&B baze i usporedba naziva te ostalih dostupnih atributa dobavljača.

C. CVM Solutions

CVM Solutions [15] je proizvođač softvera usmjerenog na analizu dobavljača. Bave se prikupljanjem podataka o dobavljačima na način da dobavljači sami dostave određene dokumente (npr. izvješće o poslovanju), a CVM Solutions

ih provjeravaju s javno dostupnim i zatvorenim izvorima (ne navode konkretno koje) te kontinuirano prate promjene vezane uz dobavljače. Na taj način su prikupili veliku količinu podataka o dobavljačima te kao uslugu nude čišćenja podataka o dobavljačima (uklanjanje višestrukih unosa, povezivanje dobavljača koji su međusobno povezani), obogaćivanja podataka o dobavljačima, analizu i praćenje dobavljača tvrtke i upozoravanje kod rizičnih situacija.

D. Primjenjivost na hrvatske (regionalne) prilike

U Hrvatskoj i susjednim državama ne postoji velik broj velikih tvrtki koje imaju potrebu za analizom troškova pa ni analiza troškova kao područje nije zastupljena. Opisana rješenja su djelomično primjenjiva na lokalne tvrtke iz nekoliko razloga: (1) baze kao što su D&B ili baza CVM Solutionsa ne sadrže ili sadrže vrlo malo podataka o tvrtkama iz regije pa se kod čišćenja podataka o dobavljačima mogu osloniti samo na postupke koji međusobno uspoređuju pojedine zapise, (2) navedena rješenja u svojim podacima ne navode imaju li mogućnost prijevoda sučelja te podržavaju li hrvatski jezik (odnosno jezike regije), također kod klasifikacije proizvoda i usluga neka rješenja navode da koriste stop-riječi (engl. *stop-words*, riječi koje ne doprinose značenju pa se zanemaruju kod analize teksta, npr. veznici, uzvici, neki prilozi i sl.), a ta metoda nije primjenjiva na hrvatski jezik bez istraživanja kojim će se dobiti baza riječi za hrvatski jezik. Još jedan aspekt kojeg proizvođači ne navode je cijena, a koja za velike strane korporacije (u referencama su AT&T, Dell, Cisco, McDonalds, Mitsubishi) može biti visoka pošto će im se investicija isplatiti ostvarenim uštedama. U regiji nema toliko multinacionalnih kompanija pa ni njihove moguće uštede neće biti tolike pa cijena može biti značajan faktor.

Postojeća rješenja jesu primjenjiva za regionalne kompanije, ali se od njih ne mogu očekivati rezultati koje postižu u tvrtkama iz engleskog govornog područja. U Hrvatskoj je pokrenut projekt MaxyTransparency [16] čiji je cilj razvoj rješenja prilagođenog jeziku i potrebama regionalnog tržišta.

VI. Zaključak

U radu je opisano područje analize troškova. Prikazani su očekivani rezultati primjene analize troškova u poslovnom sustavu kao i primjeri gubitaka do kojih dolazi neprimjenom analize troškova, a ti troškovi se događaju i prolaze nezapaženo. Opisan je proces analize troškova s posebnim naglaskom na transformaciju korisnikovih podataka.

Uvođenje sustava za upravljanje troškovima nužno je svakoj organizaciji, bilo državnoj ustanovi ili privatnoj korporaciji, posebno u uvjetima velike konkurencije, smanjenja prihoda i krize na tržištu. Sustav za upravljanje troškovima daje bolje rezultate ako se implementira zajedno sa sustavom e-nabave i sustavom za upravljanje dobavljačima. Sustavom e-nabave pokrivaju se radnje i postupci službe nabave i u njemu se, na temelju rezultata analiza iz sustava za upravljanje troškovima, može koncentrirati na robe i dobavljače koji najviše doprinose uštedama. Sustav za upravljanje dobavljačima čuva ažurnu bazu podataka o dobavljačima i njihovoj robi na koju se oslanjaju i sustav za upravljanje troškovima i sustav za e-nabavu. Ovakva baza se može postići udruživanjem kupaca i dobavljača u jedinstvenu mrežu, a može se koristiti kupcima kod e-nabava kao i dobavljačima za plasman proizvoda i usluga.

Jedno od pogodnih mjesta za implementaciju ova tri sustava je javni sektor (npr. Republika Hrvatska) i nadamo se da će takav sustav u bliskoj budućnosti biti izgrađen.

Literatura

[1] K. Pandit i H. Marmanis, *Spend Analysis: The Window Into Strategic Sourcing*. J. Ross Publishing, 2008.

[2] Wikipedia, "Spend Management." [S Interneta]. Dostupno na: http://en.wikipedia.org/wiki/Spend_management

[3] Granada Research, *Using the UNSPSC - United Nations Standard Products and Services Code*, 2001.

[4] R. A. Rudzki, D. A. Smock, M. Katzorke, i S. J. Stewart, *Straight to the bottom line*. J. Ross Publishing, 2006.

[5] M. Pavlić, *Informacijski sustavi*. Odjel za informatiku Sveučilišta u Rijeci, 2009.

[6] A. K. Elmagarmid, P. G. Ipeirotis, i V. S. Verykios, "Duplicate record detection: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 1, pp. 1–40, 2007.

[7] W. Winkler, "String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage." pp. 1184–1187, 1990.

[8] G. Kondrak, "N-gram similarity and distance," *String Processing and Information Retrieval*, pp. 115–126, 2005.

[9] W. Cohen, "Integration of heterogeneous databases without common domains using queries based on textual similarity," *ACM SIGMOD Record*, vol. 3546, pp. 201–212, 1998.

[10] Wikipedia, "Daitch-Mokotoff Soundex." [S Interneta]. Dostupno na: http://en.wikipedia.org/wiki/Daitch%E2%80%93Mokotoff_Soundex

[11] ——, "New York State Identification and Intelligence System." [S Interneta]. Dostupno na: http://en.wikipedia.org/wiki/New_York_State_Identification_and_Intelligence_System

[12] M. Weis, F. Naumann, i U. Jehle, "Industry-scale duplicate detection," *Proc VLDB Endow*, vol. 1, pp. 1253–1264, 2008.

[13] "Ariba." [S Interneta]. Dostupno na: http://www.ariba.com/

[14] "Bravo Solution." [S Interneta]. Dostupno na: https://www.bravosolution.com/

[15] "CVM Solutions." [S Interneta]. Dostupno na: http://www.cvmsolutions.com/

[16] Ris, *Projekt Maxy Transparency*, 2012.