UNIVERZITET U BEOGRADU

FAKULTET ORGANIZACIONIH NAUKA

Bojan B. Tomić

EKSPERTNI SISTEMI I SISTEMI ZA IZVEŠTAVANJE

doktorska disertacija

Beograd, 2012

UNIVERSITY OF BELGRADE

FACULTY OF ORGANIZATIONAL SCIENCES

Bojan B. Tomić

EXPERT SYSTEMS AND REPORTING SYSTEMS

Doctoral Dissertation

Belgrade, 2012

Podaci o mentoru i članovima komisije

Mentor

1. _____

   Redovni profesor dr Vladan Devedžić, Univerzitet u Beogradu, Fakultet organizacionih nauka

Članovi komisije

2. _____

   Redovni profesor dr Dušan Starčević, Univerzitet u Beogradu, Fakultet organizacionih nauka

3. _____

   Vanredni profesor dr Dragan Đurić, Univerzitet u Beogradu, Fakultet organizacionih nauka

4. _____

   Docent dr Jelena Jovanović, Univerzitet u Beogradu, Fakultet organizacionih nauka

5. _____

   Redovni profesor dr Ivan Obradović, Univerzitet u Beogradu, Rudarsko-geološki fakultet

Datum odbrane

16.5.2012.

Izjave zahvalnosti

Prvo zahvaljujem mojoj supruzi Ani i sinu Vuku zato što me vole i što su uvek tu da me saslušaju. Bez njihove ljubavi, podrške i razumevanja ne bih mogao ovoliko da postignem u životu, i sve bi bilo bez ikakvog smisla. Zbog toga, posvećujem ovaj doktorat njima.

Zahvaljujem majci Gordani i ocu Branku što su me naučili da je važno biti pošten, vredan i strpljiv i da će se rezultati rada uvek pokazati. Naučili su me i da optimistički gledam na svet, da uvek težim boljim stvarima, kao i to da nije svaki rad svrsishodan (na čemu im posebno zahvaljujem).

Zahvaljujem babi Koviljki i dedi Branislavu koji su protkali moje detinjstvo dodatnim bojama i učinili ga lepšim i ispunjenijim. Nadam se da ću i ja uspeti da učinim isto za svoju decu i unuke.

Zahvaljujem tetki Seki i babi Nevenki što su uvek bile i ostale izvor mladosti, spontanosti i duhovitosti u porodici i time mi ukazivale na vedriju stranu života.

Zahvaljujem pravim, najboljim, prijateljima Nenadu i Maji koji su svih ovih godina bili uz mene da zajedno podelimo i dobro i loše.

Posebno zahvaljujem sjajnom mentoru, čoveku i prijatelju - profesoru Vladanu Devedžiću. Prvo, zbog toga što je uočio moj akademski potencijal i prihvatio se zadatka da mi bude mentor. Drugo, što bez njegovog iskrenog interesovanja i usmeravanja ni moj rad ni ovaj doktorat ne bi bili to što jesu. I treće, zbog toga što mi je uvek pomogao kad je to bilo neophodno i time me doživotno zadužio.

Bojan Tomić

16.5.2012.

Ekspertni sistemi i sistemi za izveštavanje

Rezime

Problem istraživanja doktorske disertacije se odnosi na veliku količinu podataka i nedostatak informacija u izveštajima sistema za poslovno izveštavanje. Iako uglavnom prepuni grafika i tabela, izveštaji sadrže samo podatke tj. njihove grafičke predstave koje bi trebalo da omoguće jednostavniji uvid u poslovanje. To dovodi do posledice da korisnici ovih izveštaja moraju ručno da tumače ove podatke (pregledanjem više izveštaja), pa mogu da propuste da uoče neke bitne informacije zbog: prevelike količine podataka, premorenosti, nedostatka koncentracije, nedostatka znanja ili iskustva, zbog subjektivnosti ili zlonamernosti.

U radu se predlaže automatizacija procesa tumačenja podataka tj. njihovog pretvaranja u informacije. Ceo pristup se zasniva na relaciji koja definiše da podaci postaju informacije kada im se doda značenje (tumačenje). Da bi se dobilo tumačenje podataka i oni pretvorili u informacije, potrebno je odgovarajuće (domensko) znanje.

Predložena realizacija ovog pristupa obuhvata primenu metoda, tehnika i tehnologija ES u cilju automatizacije procesa tumačenja vrednosti poslovnih pokazatelja. Konkretno, predlaže se da se deklarativni deo znanja (za tumačenje vrednosti pokazatelja) predstavi putem klasa, a proceduralni deo znanja putem pravila i fuzzy teorije. Zaključivanje se može izvršiti ulančavanjem unapred i fuzzy zaključivanjem, a informacije (zaključci) mogu da se pretvore u rečenice koje liče na govorni jezik korišćenjem tehnike učaurenog teksta.

Posle napravljenog pregleda postojećeg stanja iz oblasti sistema za poslovno izveštavanje i ekspertnih sistema detaljno je definisan i obrazložen predloženi pristup sa teorijskog stanovišta. Diskusijom se ukazalo na (potencijalne) pozitivne i negativne aspekte rešenja, a zatim je dat i plan realizacije rešenja.

Demonstracija pristupa putem izrade prototipa - PT aplikacije (PT - enterprise Profit interpretation Tutor) je omogućila proveru i dokazivanje postavljenih hipoteza. Izveštaji koje ova aplikacija pravi sadrže i podatke (u vidu grafika i tabela) ali i informacije u vidu rečenica kontrolisanog jezika.

Ocena pristupa na osnovu evaluacije prototipa je obuhvatila rezultate evaluacione studije u kojoj su studenti bili ispitanici, a koristili su PT aplikaciju kao nastavno sredstvo. Dokazano je da studenti smatraju da je PT aplikacija korisna i relativno jednostavna za korišćenje. Objašnjenja u izveštajima se smatraju za korisna i lako razumljiva, ali bi bilo potrebno da budu obuhvatnija. Najveći broj ispitanih studenata veruje da rad sa PT aplikacijom donekle doprinosi povećanju njihovog znanja ali i da pruža brži i donekle zanimljiviji način za savladavanje gradiva o tumačenju profita.

U radu je prikazana i komparativna analiza mehanizama za objašnjavanje u modernim alatima za razvoj ekspertnih sistema i sistema poslovnih pravila, napravljen je JEFF mehanizam za objašnjavanje u Javi i izvršena je evaluacija ovog mehanizma za objašnjavanje. Evaluacija je bila dvojaka a izvršena je radi procene da li su zadati ciljevi postignuti i gde se JEFF nalazi u odnosu na slične alate. Dokazano je da su skoro svi ciljevi postignuti i da je skoro podjednako dobar kao profesionalni alat u odnosu na koji je urađena komparativna analiza.

Kada se svi rezultati sumiraju, može se reći da su hipoteze dokazane (zaista se predložene metode i tehnike mogu uspešno upotrebiti radi pravljenja informacija), ali da predloženi pristup nije isproban u situacijama za koje je namenjen. Prototip koji je napravljen dokazuje da je pristup moguć i izvodljiv za primenu, ali je isproban i razmotrena je samo njegova uspešnost u smislu nastavnog sredstva za studente, a ne kao sredstva za pomoć menadžerima u cilju tumačenja podataka.

Da bi predloženi pristup zaista bio primenjen u praksi, potrebno je da se napravi aplikacija koja će da bude prilagođena menadžerima i koja će zaista da funkcioniše u nekom poslovnom sistemu.

Ključne reči: ekspertni sistem, poslovna inteligencija, izveštavanje, mehanizam za objašnjavanje, poslovna pravila, ključni indikatori poslovanja

Naučna oblast: Računarske nauke

Uža naučna oblast: Veštačka inteligencija

Expert systems and reporting systems

Abstract

Dissertation research problem relates to the large amount of data and lack of information in the reports for business reporting systems. Although rich with charts and tables, reports actually contain only data (their graphical representations) and little or no information. Report users need to manually interpret this data (reviewing various reports) and, as consequence, may fail to detect some important information because of: too much data, fatigue, lack of concentration, lack of knowledge or experience, because of subjectivity, or malice.

This paper proposes automating the data-to-information transformation process. The proposed approach is based on the relation that states that data becomes information when meaning (interpretation) is added to it. Also, appropriate (domain) knowledge is required in order to obtain meaning from data.

The proposed implementation of this approach involves applying expert system methods, techniques and technologies in order to automate key performance indicator interpretation. Specifically, declarative knowledge is to be presented by classes, and procedural knowledge with rules and fuzzy theory. The information can be derived by forward chaining and fuzzy inference, and it can be then converted into natural-language-like sentences by using canned text.

After reviewing the current situation in the field of business intelligence systems and expert systems, the proposed approach is explained in detail from the theoretical point of view. It is then followed by a discussion focusing on the (potential) positive and negative aspects of the solution, and then by the implementation plan.

By demonstrating the approach with a prototype - PT application (PT - Enterprise Profit Interpretation Tutor), testing and proving of the hypotheses was enabled. Reports that the application provides contain data (in the form of graphics and tables) but also information (in the form of natural-language-like sentences).

The evaluation consisted of an evaluation study in which subjects were students who used PT as a teaching aid. The results show that students believe that PT is useful and

relatively easy to use. Explanations in the reports are considered useful and easy to understand, but would need to be more comprehensive. The largest number of students surveyed believe that working with PT application to some extent contributes to increasing their knowledge and that it provides a faster and somewhat more interesting way to mastering profit interpretation.

A comparative analysis of explanation facilities in modern tools for the development of expert systems and business rules is also presented in this paper. JEFF explanation facility, which was created in order for the prototype to work, is then presented and evaluated. The evaluation was twofold and was performed to assess whether objectives have been achieved and where JEFF stands when compared to similar tools. It was proved that almost all objectives have been achieved and that JEFF is almost as good as the professional tool it was compared with.

When summarizing all results, it can be said that the hypotheses are proved (indeed, the proposed methods and techniques can be successfully used for the purpose of making information), but that the proposed approach was not tested in an enterprise environment. The prototype that was made proves that the approach is feasible to implement, but its effectiveness was evaluated only in terms of teaching aids for students, not as a tool to help managers interpret data.

If the proposed approach was to be applied in practice, it would be necessary to create applications tailored to meet the needs of managers and that would actually function in an enterprise environment.

Keywords: Expert system, business intelligence, reporting, explanation facility, business rules, key performance indicator

Academic field: Computer science

Academic sub-field: Artificial intelligence

Sadržaj

1 Uvod

U današnjem svetu ubrzanog tempa promena, jedan od najvažnijih resursa jesu informacije. Od izveštavanja o svakodnevnim događajima do visoko specijalizovanih analitičkih procena, svako ko je blagovremeno obavešten poseduje moć da odlučuje i reaguje mnogo brže i bolje u odnosu na one koji to nisu. Zbog razvoja informaciono-komunikacionih tehnologija, podaci i informacije su sada postali lako dostupni, pa se mogu preneti, obraditi i umnožiti uz minimalan napor.

Problem koji se u poslednje vreme pojavljuje je, zapravo, veliki obim informacija i nemogućnost da se one pregledaju, prihvate i upotrebe od strane korisnika. Ova doktorska disertacija za temu ima uvođenje novog pristupa za automatizovano tumačenje poslovnih podataka i njihovo pretvaranje u relevantne informacije. Cilj je da se pomogne poslovnim korisnicima tako da u potpunosti mogu da iskoriste potencijal podataka i informacija koji su im dostupni.

1.1 Poslovno izveštavanje

Izveštavanje o rezultatima poslovanja je jako bitan proces za svako preduzeće. Dobijanje pravovremenih izveštaja sa tačnim podacima i informacijama je osnova za donošenje dobrih poslovnih odluka i pravilan izbor ciljeva i strategija poslovanja. Zbog toga je jedan od ključnih delova informacionog sistema svakog preduzeća upravo sistem za (poslovno) izveštavanje.

Razvoj računarskih tehnika i tehnologija je uticao i na *sisteme za poslovno izveštavanje* (u daljem tekstu SPI) i to u toj meri da je sada moguće dobiti bilo koji izveštaj za nekoliko sekundi. Najsavremeniji SPI idu i korak dalje. Naime, primaoci izveštaja mogu da, korišćenjem programa, sami specificiraju kakav izgled izveštaja žele, koja bi tačno trebalo da bude njegova sadržina i da pritiskom na dugme dobiju automatski generisan izveštaj koji je u skladu sa tom specifikacijom. Osim poboljšanja u brzini, dobija se i na drugim aspektima efikasnosti čitavog procesa izveštavanja jer se eliminiše potreba za ljudskom intervencijom u procesu stvaranja izveštaja.

Tehnologije koje se danas koriste za poslovno izveštavanje su, prema tome, specijalizovane za postizanje veoma visokog nivoa efikasnosti i kvaliteta i koriste se u međusobnoj kombinaciji da bi se postigli što bolji rezultati. Neke od najpopularnijih tehnologija su: extract-transform-load (ETL), data warehouse, on-line analytical processing (OLAP), data mining, scorecards, dashboards itd.

1.2 Problem istraživanja

Sada se, međutim, pojavljuju drugi problemi. Proces izveštavanja je doveden do nivoa visoke efikasnosti, ali se postavlja pitanje efektivnosti tj. koliko su izveštaji koje SPI stvaraju zaista korisni donosiocima odluka. Različiti članci i intervjui sa ekspertima iz ove oblasti upućuju na to da efektivnost možda i nije na toliko visokom nivou. Neki od razloga koji se najčešće navode su: prenatrpanost izveštajima, previše podataka, programi su složeni za korišćenje itd. Kada se analiziraju metode koje se koriste u okviru SPI, kao jedini zaključak se nameće to da su SPI orijentisani na pružanje podataka a ne informacija, i da je upravo u tome uzrok problema. Da bi se objasnio ovakav stav, potrebno je poći od definicija podatka i informacije i njihovog međusobnog odnosa. Iako ne postoji jedna, univerzalna definicija informacije, većina naučnika se slaže sa sledećom relacijom:

(1) PODATAK + ZNAČENJE (TUMAČENJE) = INFORMACIJA

Drugim rečima, podaci predstavljaju vrednosti izražene u numeričkoj, simboličkoj ili drugoj formi, a informacije nastaju kada se ti podaci protumače. Može se dodati i da informacije doprinose proširenju znanja primaoca.

Korisnici SPI se dele u dve grupe. Prvu grupu čine poslovni analitičari. Oni koriste SPI na pronađu neke nove, još uvek nepoznate, zakonitosti među podacima. Izveštaji koje oni dobijaju od SPI su puni podataka a ne informacija, ali to je prihvatljivo jer je sam zadatak tumačenja ovih podataka i pronalaženja novih zakonitosti složen i najbolje ga je prepustiti iskustvu, znanju i intuiciji poslovnog analitičara.

Druga grupa korisnika SPI su menadžeri. Ono što je osnovna svrha ovakvih sistema (sa menadžerske tačke gledišta) je da obezbede da se odgovarajući podaci i informacije

potrebni za odlučivanje nađu u pravo vreme i u pravoj formi kod odgovarajućih lica. Izveštaji koje pružaju SPI sadrže vrednosti poslovnih pokazatelja (profit, troškovi, učešće na tržištu itd.) izražene u odgovarajućoj formi. Ove vrednosti se mogu prikazati u numeričkoj formi, ali i korišćenjem tabela, grafikona i drugih sredstava za vizuelizaciju. Ali, kako god da su predstavljeni, ovi pokazatelji i dalje sadrže samo podatke a ne i informacije. Korisnik izveštaja je taj koji ručno analizira sve ove podatke, tumači ih i stvara informacije. U ovom slučaju to nije opravdano, jer su zakonitosti koje se traže već unapred poznate – samo ih treba uočiti ako se pojave.

Posledice ovakvog ručnog pristupa tumačenju podataka od strane menadžera (u daljem tekstu korisnika) mogu da uzrokuju propuštanje da se uoče neke bitne informacije zbog:

- Prevelike količine podataka (teško je ili nemoguće sve detaljno pregledati)
- Premorenosti ili nedostatka koncentracije korisnika
- Nedostatka znanja ili iskustva korisnika
- Zbog subjektivnosti ili zlonamernosti korisnika (nije mu u interesu da se neka informacija sazna)

Čini se da bi rešenje bilo da se automatizuje ovaj proces pretvaranja podataka u informacije. Tada bi SPI kreirao izveštaje koji sadrže podatke ali i informacije. Informacije bi se izvodile automatski i bez ljudskog učešća čime bi postale dostupne bilo kada, bilo gde i bez ikakvog uloženog napora. Takođe, korisnici izveštaja ne bi više morali ručno da analiziraju velike količine podataka. Da bi se ovo moglo postići, potrebno je shvatiti kako se podaci pretvaraju u informacije.

Kada korisnik dobije izveštaj, on/ona pregleda više različitih tabela, grafika i drugih delova izveštaja i postavlja podatke koji su predstavljeni u njima u odgovarajući kontekst, npr.: ove godine profit je 12%, a prošle godine je bio 8%. Sledeći korak je formiranje informacija na osnovu ovih kontekstualno povezanih podataka. U tu svrhu, korisnik koristi svoje znanje: “Kada je ovogodišnji profit veći od prošlogodišnjeg, poslovanje se odvija dobro”. Konačno, informacija koja se dobija izgleda ovako: “Sa obzirom na to da prošlogodišnji profit iznosi 8%, a ovogodišnji 12% poslovanje se odvija dobro”. Ceo proces se može predstaviti jednačinom:

(2) PODATAK + ZNANJE = INFORMACIJA

Sa obzirom na to da važi (1), sledi da:

(3) ZNANJE = ZNAČENJE (TUMAČENJE)

Prema tome, da bi se automatizovao proces transformacije poslovnih podataka u informacije, potrebno je formalizovati i implementirati odgovarajuće znanje za tumačenje ovih pokazatelja i to tako da se može koristiti u okviru SPI.

Ekspertni sistemi (expert system - u daljem tekstu ES) predstavljaju sisteme koji mogu da donose zaključke i rezonuju u okviru neke stručne oblasti na sličan način kao čovek. ES u sebi sadrže ekspertsko znanje koje im omogućava donošenje zaključaka. Na primer, sledeće znanje potrebno za tumačenje profita se može formalizovati korišćenjem tehnike pravila iz oblasti ES:

"Ako je ovogodišnji profit veći od prošlogodišnjeg, poslovanje se odvija dobro"

(4) IF Kompanija.getProfit(TEKUĆA_GODINA) > Kompanija.getProfit(PROŠLA_GODINA)

THEN Kompanija.setStatus(DOBRO_POSLOVANJE)

Pri tumačenju poslovnih pokazatelja, menadžeri često koriste nejasne izraze. Na primer, za neko preduzeće koje je osnovano pre godinu dana, može se reći da je novo, a za neko koje je osnovano pre četiri godine da je "relativno" novo. Takođe, profit se može karakterisati kao "dovoljno" veliki, negativan ili, pak, neutralan, a njegova vrednost, kada se uporedi sa prošlogodišnjom, može biti "malo" veća, "značajno" veća itd. Svaki od ovih izraza ima određene varijacije u granicama koje ga određuju. Tako će za jedno preduzeće profit od 5% biti "veoma" veliki, a za neko drugo neutralan (veći od nule, ali ipak beznačajan). Ove granice se pomeraju čak i vremenom pod uticajem raznih internih i eksternih faktora: stanja na tržištu, konkurencije, politike i ciljeva preduzeća, nabavnih cena materijala itd. Kada na tržištu vlada kriza, čak se i profit oko nule smatra dobrim poslovnim rezultatom.

Zbog svega navedenog, potrebno je koristiti i *fuzzy* skupove i pravila pri predstavljanju znanja za tumačenje poslovnih pokazatelja. Na ovaj način se mogu mnogo preciznije

izraziti i finese u tumačenju pokazatelja, kao npr.:

"Ako je preduzeće relativno novo i ako ostvaruje neutralan profit, poslovanje se odvija dobro"

(5) IF Kompanija.getStarosnaKategorija() == "relativno novo" AND
Kompanija.getProfitStatus(TEKUĆA_GODINA) == "neutralan"
THEN Kompanija.setStatus(DOBRO_POSLOVANJE)

Smatra se da je jedna od najvažnijih karakteristika ES mogućnost da korisniku "objasne" proces rezonovanja tj. kako su došli do rešenja ili zašto postavljaju određeno pitanje. I u ovom slučaju to je veoma važno. Ako ES zaključi da neki pokazatelji ukazuju na to da preduzeće doživljava neku krizu, bitno je da korisnik dobije i objašnjenje kako se došlo do tog zaključka. Na taj način, korisnik može da poveća nivo poverenja u sistem i rešenja koje on vraća. Takođe, ova objašnjenja moraju da budu precizna, tačna i prilagođena korisniku tj. njegovom nivou znanja jer se, u suprotnom, ostvaruje suprotan efekat - korisnik gubi poverenje u sistem.

Mehanizam za objašnjavanje (explanation facility) je deo ES koji je zadužen za pružanje objašnjenja i, u ovom slučaju, možda bi najbolje bilo da pruža objašnjenja u vidu rečenica koje liče na rečenice govornog jezika. To se može ostvariti korišćenjem tehnike *učaurenog teksta* (canned text). Pored toga, da bi objašnjenja bila još preciznija, potrebno je omogućiti i dinamičko umetanje konkretnih vrednosti u ovaj, inače statičan, tekst. Ovakva objašnjenja je dovoljno samo pročitati pa korisnicima nije potrebno nikakvo tehničko predznanje, iskustvo, niti dalje tumačenje da bi ih koristili. Pravilo (5) bi onda, osim zaključka, u THEN delu sadržalo i deo objašnjenja:

(6) IF Kompanija.getStarosnaKategorija() == "relativno novo" AND
Kompanija.getProfitStatus(TEKUĆA_GODINA) == "neutralan"
THEN Kompanija.setStatus(DOBRO_POSLOVANJE)
Kompanija.addObjašnjenje("Preduzeće je relativno novo. Pošto je nedavno počelo sa radom, ne očekuje se da će moći da ostvari pozitivan profit, pa se i ovaj profit od {} smatra za dobar.")

Kada se znanje formalizuje korišćenjem ovih tehnika, neophodno je upotrebiti i neku od tehnika zaključivanja da bi se izvele informacije. U ovom slučaju, u pitanju je *fuzzy zaključivanje* (za fuzzy činjenice i pravila) i *ulančavanje unapred* (za obična pravila). Ulančavanje unapred je *zaključivanje vođeno podacima* (data-driven) tj. omogućava izvođenje što većeg broja zaključaka iz nekog skupa početnih podataka, što se i traži. Naravno, potrebno je iskoristiti i odgovarajuće tehnike za rešavanje konflikta, kako bi se pravila izvršavala u željenom redosledu.

Na kraju, ostaje i pitanje integracije ovakvog ES u SPI. Savremena okruženja za razvoj ES se najčešće prave sa namerom da mogu lako da budu integrisana u neki običan program tako da to, sa tehničke strane, ne predstavlja problem. Trebalo bi da postoje dve dodirne tačke sa SPI: preuzimanje ulaznih podataka i formiranje izveštaja. Drugim rečima, ES bi trebalo da iz skladišta podataka već postojećeg SPI preuzme podatke potrebne za zaključivanje, a da (automatski generisane) informacije unese u izveštaj zajedno sa podacima koji se tamo već nalaze.

Sve navedene karakteristike ES mogu da čine tehnike i tehnologije ES dobrim kandidatima za upotrebu u okviru SPI kao sredstva za podizanje nivoa efektivnosti procesa izveštavanja putem povećanja količine informacija u izveštajima. Efekti bi mogli da budu sledeći:

- Korisnicima bi bilo lakše jer ne bi morali sami da tumače podatke.
- Korisnici ne bi morali da poseduju veliku količinu tehničkog znanja da bi iskoristili potencijal SPI.
- Brže reagovanje zbog toga što bi SPI trebalo manje vremena za uočavanje pojava i njihovih uzroka nego korisniku.
- Informacije u izveštajima bi bile objektivne i imale bi konstantan kvalitet (kvalitet ne bi zavisio od znanja, iskustva, koncentracije ili subjektivnosti osobe koja ih stvara).
- Rešio bi se problem pretrpavanja podacima.
- Stvorila bi se mogućnost praćenja jako velikog broja poslovnih pokazatelja odjednom.

U ovom radu su prikazani rezultati istraživanja koje je izvršeno sa ciljem rešavanja opisanog problema. Ostatak uvoda čini kratak prikaz projekta istraživanja kao i kratak pregled rada po poglavljima. Detaljan prikaz projekta istraživanja se može naći u pristupnom radu [1].

1.3 Predmet i ciljevi istraživanja

Teorijsko određenje predmeta istraživanja je: korišćenje tehnika i tehnologija ekspertnih sistema u okviru sistema za poslovno izveštavanje kao sredstva za automatizovano formiranje informacija u izveštajima.

Konkretan (operacioni) predmet istraživanja je:

1. Primena tehnika i tehnologija ekspertnih sistema u okviru sistema za poslovno izveštavanje kao sredstva za automatizovano formiranje informacija u izveštajima.
    1.1. Korišćenje tehnike pravila, klasa (objekata) i fuzzy logike za predstavljanje znanja potrebnog za tumačenje poslovnih pokazatelja.
    1.2. Korišćenje tehnike ulančavanja unapred i fuzzy zaključivanja za automatsko (bez učešća ljudi) donošenje zaključaka i stvaranje informacija o rezultatima poslovanja na osnovu poslovnih pokazatelja predstavljenih u vidu podataka i unetog znanja potrebnog za tumačenje poslovnih pokazatelja.
    1.3. Korišćenje tehnika za objašnjavanje za pretvaranje donetih zaključaka i stvorenih informacija u rečenice govornog jezika u cilju kreiranja izveštaja.

Opšti cilj istraživanja se sastoji u tome da se ukaže na jedan od mogućih načina za unapređenje i razvoj SPI putem automatizovanja procesa formiranja informacija u izveštajima. Direktna posledica toga je bolje izveštavanje u preduzećima, a samim tim, i poboljšanje poslovanja u celini.

Naučni cilj istraživanja je povećanje naučnog fonda informatike putem ukazivanja na nove moguće primene tehnika i tehnologija iz oblasti ekspertnih sistema. Cilj je da se:

- Pruži pregled postojećeg stanja iz oblasti SPI i ES.

- Formira novi pristup problemu formiranja izveštaja SPI.
- Demonstrira ovaj pristup na delu putem izrade prototipa.
- Pruži pregled i ocena pristupa na osnovu performansi prototipa.

1.4 Hipoteze, metode i plan istraživanja

Generalna hipoteza istraživanja je:

> "Tehnike i tehnologije ekspertnih sistema mogu uspešno da se primene u okviru sistema za izveštavanje kao sredstvo za automatizovano formiranje informacija u izveštajima."

Posebne hipoteze istraživanja su sledeće:

1. "Tehnike za predstavljanje znanja (iz oblasti ES) mogu uspešno da se koriste za predstavljanje znanja potrebnog za tumačenje poslovnih pokazatelja."
2. "Tehnike zaključivanja (iz oblasti ES) mogu uspešno da se koriste za automatsko (bez učešća ljudi) donošenje zaključaka i stvaranje informacija o rezultatima poslovanja na osnovu poslovnih pokazatelja predstavljenih u vidu podataka i unetog znanja potrebnog za tumačenje ovih poslovnih pokazatelja."
3. "Tehnike za objašnjavanje (iz oblasti ES) mogu uspešno da se koriste za pretvaranje donetih zaključaka i stvorenih informacija u rečenice koje liče na govorni jezik u cilju kreiranja izveštaja."

Pojedinačne hipoteze istraživanja su sledeće:

> 1.1."Deklarativni deo znanja potrebnog za tumačenje poslovnih pokazatelja može da se izrazi i formalizuje korišćenjem klasa."
>
> 1.2."Proceduralni deo znanja potrebnog za tumačenje poslovnih pokazatelja može da se izrazi i formalizuje korišćenjem pravila i fuzzy teorije."
>
> 2.1."Tehnike ulančavanja unapred i fuzzy zaključivanja mogu da oponašaju proces ljudskog zaključivanja o poslovnim pokazateljima."
>
> 3.1."Automatski izvedeni zaključci i informacije o poslovnim pokazateljima mogu da se pretvore u rečenice koje liče na govorni jezik korišćenjem tehnike

učaurenog teksta mehanizma za objašnjavanje.”

Opštenaučna metoda koja se koristi u istraživanju je metoda modelovanja. Posebna naučna metoda koja se koristi u istraživanju je informatička metoda.

Metoda modelovanja je izabrana zbog toga što su oblasti informacionih sistema i veštačke inteligencije takve da je, u najvećem broju istraživanja, najpoželjnije napraviti model koji će da simulira željeno ponašanje i onda istraživanje vršiti nad modelom. Takođe, veoma često je moguće realizovati taj model u vidu prototipa. To važi i za ovo istraživanje. Mogućnost primene ES u okviru SPI je najbolje modelovati i predstaviti korišćenjem konkretnog prototipa-programa.

Informatička metoda je posebna naučna metoda u kojoj se pojava koja se istražuje posmatra kao model tj. kroz njene ulaze, izlaze, stanje i povratnu spregu. Sa obzirom na to da se u ovom istraživanju koristiti prototip, jasno je da je informatička metoda potpuno primerena.

Konačni plan istraživanja obuhvata tri osnovna koraka:

1. Formiranje arhitekture i logičkog modela rešenja
2. Kreiranje prototipa
3. Evaluacija prototipa (evaluaciona studija)

1.5 Pregled sadržaja rada

Struktura ovog rada odražava plan istraživanja. Prvo poglavlje predstavlja uvod u istraživanje u kojem je ukratko opisan problem istraživanja, predmet, ciljevi, hipoteze, metode i plan. Osim toga, dat je i sažet prikaz rešenja.

Drugo poglavlje se odnosi na pregled stanja iz naučne oblasti istraživanja. Sa obzirom na to da je u pitanju istraživanje koje spaja dve ne tako srodne oblasti, dat je pregled i SPI i ES. U okviru svake od njih su predstavljene najvažniji pojmovi, klasifikacije, tehnike, metode i tehnologije, kao i neki tekući pravci razvoja i trendovi. Nakon toga, dat je i detaljan opis problema istraživanja.

U trećem poglavlju se nalazi prikaz arhitekture i logičkog modela rešenja. Početak ovog

poglavlja sadrži osnovne odrednice i objašnjenja o tome šta su informacije i podaci, kao osnove za razumevanje rešenja. Nakon toga, prikazana je logička arhitektura rešenja da bi, potom, usledio prikaz tehnika i metoda koje se koriste u rešenju kao i načina na koji se koriste.

Četvrto i peto poglavlje se odnose na implementaciju. Prvo je prikazan mehanizam za objašnjavanje koji je morao da bude razvijen kao preduslov za nastavak implementacije. Razlog zašto je razvijen, ciljevi koje bi trebalo da ispuni, njegova arhitektura i mogućnosti čine najveći deo četvrtog poglavlja. Razvijeni mehanizam za objašnjavanje je zasebno ocenjen poređenjem sa sličnim softverskim alatom a rezultati tog prikaza, kao i primer njegovog korišćenja su dati na kraju ovog poglavlja.

Peto poglavlje je centralno što se tiče implementacije, jer se u njemu opisuje napravljeni prototip - aplikacija za automatizovano tumačenje profita u preduzećima. U pitanju je aplikacija koja automatski tumači podatke o profitu, prihodima, troškovima i još nekim povezanim pokazateljima i daje detaljan, brz i objektivan uvid u sliku o poslovanju preduzeća. Zamišljeno je da se ovaj prototip upotrebi u nekoj realnoj situaciji. Međutim, zbog svoje jednostavnosti, odlučeno je da se, trenutno, iskoristi kao dodatno sredstvo za učenje.

Rezultati evaluacije mehanizma za objašnjavanje i prototipa su dati u šestom poglavlju. Prvi deo evaluacije čine rezultati evaluacione studije sa studentima, dok u slučaju mehanizma za objašnjavanje, postoji i komparativna analiza sa sličnim mehanizmima za objašnjavanje. Nakon toga, dat je prikaz dobrih i loših aspekata rešenja koji su uočeni tokom izrade i puštanja u rad mehanizma za objašnjavanje i prototipa. Na kraju, dati su budući pravci istraživanja.

Poslednje poglavlje ovog rada čini zaključak u kojem se sumira sve što je urađeno u istraživanju, uz opšti osvrt na problem i predmet istraživanja, ciljeve, hipoteze, postignute rezultate i doprinose, kao i rezultate evaluacije. Na samom kraju su data i neka razmišljanja o budućim pravcima istraživanja u smislu razvoja i primene predloženog pristupa.

2 Pregled oblasti istraživanja

Ovo poglavlje sadrži pregled dosadašnjih rezultata istraživanja u oblasti SPI i ES. Sa obzirom na to da su ove dve oblasti međusobno relativno nezavisne, pregledi su dati kao dve odvojene celine. Uz to, dati su i skoriji primeri istraživanja koje spajaju ove dve oblasti.

2.1 Sistemi za poslovno izveštavanje

SPI su uvek bili deo informacionog sistema preduzeća koji je pružao godišnji, mesečni ali i nedeljni uvid u poslovanje. Danas je izveštavanje postalo dnevna, pa čak i učestalija potreba, a to se odrazilo i na način gledanja na SPI. Neformalna definicija izveštavanja je da je to "proces koji podrazumeva preuzimanje i strukturiranje podataka iz različitih izvora uz pomoć različitih logičkih modela u cilju pravljenja izveštaja namenjenim korisnicima." [2]

Iako je ova definicija jasna i jednostavna, definicija SPI kao takvih, ne može da se nađe. Razlog je taj što se umesto pojma SPI koristi popularan pojam *poslovna inteligencija* (business intelligence - u daljem tekstu PI). PI se odnosi na izveštavanje u poslovnom domenu uopšte.

"Poslovna inteligencija predstavlja korišćenje svih potencijala podataka i informacija u preduzeću radi donošenja boljih poslovnih odluka i, u skladu sa tim, identifikaciju novih poslovnih mogućnosti." [3]

"Poslovna inteligencija je svesna i metodična transformacija podataka iz raznih izvora podataka u neki drugi oblik, a sa ciljem pružanja poslovnih informacija koje su orijentisane ka ostvarivanju rezultata" [4]. Isti autor navodi da PI često podrazumeva korišćenje mešavine različitih softverskih alata i baza podataka koji obezbeđuju potrebnu infrastrukturu.

Pojam SPI predstavlja samo deo onoga što PI podrazumeva i odnosi se na konkretne

tehnologije, tehnike i metode koje se koriste u procesima sakupljanja i analize poslovnih podataka. Da bi se izbegle nejasnosti i nedorečenosti, u daljem tekstu će se pod SPI podrazumevati sledeće:

"Sistemi za poslovno izveštavanje (SPI) su sistemi koji podržavaju i omogućavaju izvršavanje procesa formiranja informacija u domenu poslovanja. Ovi procesi mogu biti opisani kao procesi transformacije podataka u informacije, a onda u znanje u poslovnom domenu".

Ovako definisani SPI nisu isto što i *sistemi za podršku odlučivanju* (decision support system). Sistem za podršku odlučivanju je "interaktivna kompjuterska aplikacija koja kombinuje podatke i matematičke modele u cilju pružanja pomoći donosiocima odluka pri rešavanju problema koje vođenje javnih i privatnih organizacija donosi" [5]. Sistemi za podršku odlučivanju su često veoma blisko povezani sa PI [6] ali, za razliku od njih, SPI ne podrazumevaju pomoć pri generisanju alternativa i vršenju izbora. Ove dve vrste sistema se mogu preklopiti u domenu prikupljanja i analiziranja podataka i informacija, pa je pravilno reći da sistemi za podršku odlučivanju mogu da koriste ili sadrže SPI ili neki njegov deo, ali da su njihove svrhe različite: SPI su okrenuti isključivo analizi prošlog, postojećeg i budućeg stanja preduzeća, a ne pružanju pomoći pri procesu donošenja odluke.

Može se reći da postoje dve osnovne vrste korisnika SPI. Prvo, to su oni kadrovi koji vrše neku rukovodeću funkciju u preduzeću - *menadžeri*. Zbog toga se SPI smatraju za sastavni deo *upravljačkog informacionog sistema preduzeća* (management information system, executive information system) kao dela informacionog sistema. Međutim, SPI koriste i tzv. *poslovni analitičari*. Razlike u načinu korišćenja SPI od strane ove dve vrste korisnika se svode na razliku u očekivanim rezultatima. Sa menadžerske tačke gledišta, SPI bi trebalo da obezbede da se odgovarajući podaci i informacije potrebni za odlučivanje nađu u pravo vreme i u pravoj formi kod odgovarajućih lica, a poslovni analitičari koriste SPI da pronađu neke nepoznate zakonitosti u podacima.

Metode i tehnike koje SPI koriste u svom radu se mogu podeliti u nekoliko grupa u skladu sa funkcijom koju obavljaju:

- Metode i tehnike za preuzimanje, integraciju i skladištenje podataka
- Metode i tehnike za analizu podataka i predviđanje
- Metode i tehnike za prikazivanje podataka i pretvaranje podataka u informacije

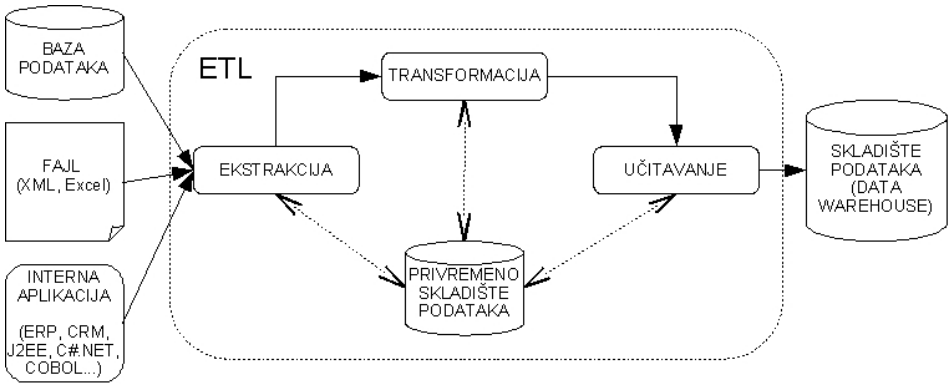
2.1.1 Metode i tehnike za preuzimanje, integraciju i skladištenje podataka

Ova grupa sadrži dve podgrupe metoda i tehnika a njihova svrha je da se svi poslovni podaci preuzmu iz različitih. heterogenih, izvora i objedine i struktuiraju radi lakšeg izveštavanja.

Ekstrakcija, transformacija i učitavanje podataka (Extract Transform Load – ETL)

U svakoj firmi postoji potreba za integrisanjem i centralizovanjem podataka. Osnovni razlog je taj što je, u najvećem broju slučajeva, informacioni sistem firme heterogen tj. sastoji se iz različitih (najčešće nepovezanih) celina koje su implementirane u različitim tehnologijama i različitim vremenskim periodima. ETL metode i tehnike služe tome se podaci iz svih tih delova povežu u jednu celinu i unesu u SPI.

"*Ekstrakcija, transformacija i učitavanje podataka* (Extract Transform Load - ETL) je proces ekstrakcije podataka iz operativnih ili eksternih izvora, njihove transformacije (koja uključuje pročišćavanje, agregaciju, sumiranje, integraciju i osnovne forme obrade npr. "1" se pretvara u "Muško", "2" u "Žensko") i učitavanja u neku vrstu *skladišta podataka* (Data warehouse). Ovaj izraz može da se odnosi i na konkretne softverske alate koji vrše ovaj proces" [7]. ETL proces je prikazan na slici (Slika 1).



Slika 1: ETL proces (slika po uzoru na [8])

Prvi korak u ETL procesu je ekstrakcija podataka. Ekstrakcija podrazumeva preuzimanje podataka iz različitih izvora – baze podataka, nekog fajla ili iz neke interne aplikacije. Kada se podaci ekstrakuju, oni se, u izvornoj formi smeste u neko privremeno skladište podataka. Sledeći korak je transformacija podataka. Ona se radi u cilju postavljanja podataka u formu koja odgovara ciljnom skladištu podataka. Pored toga što se podaci transformišu, oni se istovremeno i pročišćavaju. Poslednji korak je učitavanje podataka tj. prebacivanje transformisanih podataka iz privremenog skladišta u ciljno skladište.

ETL nije proces koji se odvija jednokratno. Podaci moraju stalno da se ažuriraju, pa su sve metode i tehnike smišljene tako da je dovoljno samo jednom postaviti pravila transformacije i lokacije izvora podataka da bi ETL alat, u okviru unapred definisanih vremenskih intervala, sam vršio ceo proces.

Skladište podataka (Data Warehouse)

Kada se podaci preuzmu i integrišu, potrebno ih je skladištiti tako da se kasnije mogu lako preuzeti u cilju vršenja analize i pravljenja izveštaja. Skladišta podataka koja se koriste u okviru operativnih aplikacija (transakcioni deo informacionog sistema) nisu dorasla ovom poslu. Kao posledica, razvijena su skladišta podataka posebno prilagođena vršenju analiza i pravljenju izveštaja. Termin koji se koristi u praksi je *data warehouse* (u daljem tekstu DW). Najjednostavnija definicija je da je DW "kolekcija integrisanih, predmetno orijentisanih baza podataka posebno projektovanih da pruže

informacije potrebne za donošenje odluka” [9]. To znači da DW ima sledeće osnovne karakteristike [10]:

- Integracija - podaci su integrisani iz više izvora.
- Predmetna orijentisanost - podaci su grupisani prema predmetu ili funkciji u organizaciji a ne prema pojedinačnim poslovnim procesima.
- Nepromenljivost - podaci se samo dodaju u DW i skoro nikada se ne menjaju niti brišu.
- Orijentisanost ka donošenju odluka - podaci su, u suštini, denormalizovani da bi se mogle izvršiti brza sumiranja i pravljenja složenih izveštaja.
- Vremenska podeljenost - svaki podatak se odnosi na neki vremenski period.
- Sumiranost - DW sadrži transakcione, ali i sumirane podatke.

Način struktuiranja podataka (a ne tehnologija implementacije) određuje da li je neko skladište podataka DW. Praktično rečeno, DW se može implementirati korišćenjem relacionih baza podataka, hijerarhijskih baza podataka, objektnih baza podataka ili jednog ili više običnih fajlova.

Pojam koji se često koristi uz DW je i *market podataka* (Data Mart) [11]. Svaki DW može da ima jedan ili više marketa podataka od kojih se svaki odnosi na neku konkretnu poslovnu funkciju (prodaja, nabavka, marketing...) ili na neki organizacioni deo (odeljenje za prodaju knjiga, odeljenje za prodaju kancelarijskog materijala...).

2.1.2 Metode i tehnike za analizu podataka i predviđanje

Kada se podaci sakupe, integrišu i sačuvaju u odgovarajućoj formi u DW, moguće je vršenje analize nad tim podacima i, eventualno, predviđanje budućih vrednosti. Razlika između analize i predviđanja je u vremenskoj usmerenosti rezultata. Analiza podataka je proces koji se vrši sa ciljem uočavanja nekih pojava ili zakonitosti u sadašnjem ili prošlom poslovanju na osnovu prošlih i sadašnjih podataka o poslovanju. Predviđanje budućih vrednosti se isto vrši nad podacima o prošlom i sadašnjem poslovanju, ali sa ciljem prognoziranja budućih zakonitosti i pojava. Metode i tehnike koje se koriste u ove svrhe su date u produžetku.

Analiza višedimenzionalnih podataka u realnom vremenu (On-Line Analytical Processing - OLAP)

DW sadrži podatke koji su atomski ili blago sumirani. Izveštaji i analize, sa druge strane, uvek sadrže složene izvedene tj. sumirane podatke. Da bi se izveštaj formirao, potrebno je sumirati atomske podatke iz DW i, eventualno, izvršiti neku selekciju nad njima. Kada bi se celo izračunavanje vršilo svaki put kada je potrebno napraviti izveštaj, vreme odziva za izveštaje bi bilo izuzetno dugo, a ceo DW bi bio preopterećen. Zato se pristupa sumiranju podataka unapred i njihovom indeksiranju radi lakše selekcije i bržeg formiranja izveštaja. Tehnike i metode koje se pritom koriste se jednim zajedničkim imenom zovu *OLAP - analiza višedimenzionalnih podataka u realnom vremenu* (On-Line Analytical Processing).

Sam pojam OLAP-a je prvi upotrebio E. F. Codd, tvorac koncepta relacionih baza podataka. On navodi da "u okviru tradicionalnih transakcionih baza podataka najviše nedostaje mogućnost sjedinjavanja, pregeda i analize podataka prema više dimenzija u svakom trenutku i to na način koji odgovara poslovnim analitičarima. Ovaj zahtev se zove višedimenzionalna analiza. Možda je bolje upotrebiti uopšteniji naziv za ovakvu vrstu funkcionalnosti – višedimenzionalna analiza u realnom vremenu (OLAP), gde je sama višedimenzionalna analiza samo jedna od karakteristika koja će se podrazumevati pod ovim pojmom." [12] Jedna savremenija definicija OLAP-a je da je to "prikupljanje i analiza podataka u realnom vremenu u cilju otkrivanja trendova i statistika vezanih za poslovanje koji nisu uočljivi na osnovu atomskih podataka preuzetih iz DW. OLAP se takođe naziva i višedimenzionalnom analizom" [11].

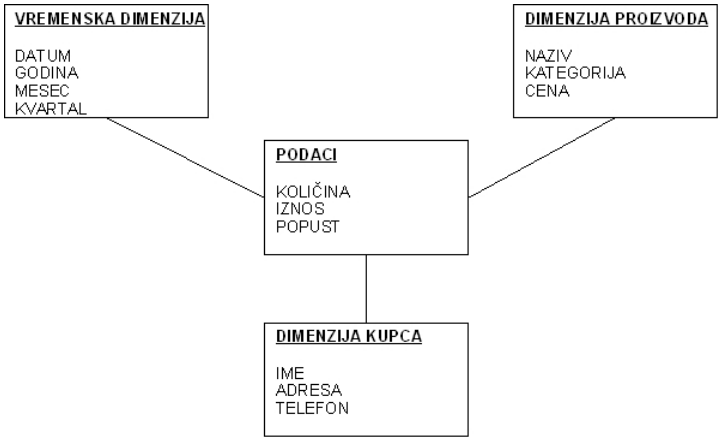
Da bi definisao šta tačno čini jedan OLAP sistem, E. F. Codd je napravio dvanaest pravila za evaluaciju OLAP sistema [12]. Ova pravila su relativno složena, pa se može prihvatiti pojednostavljeni skup pravila za definisanje OLAP sistema, koje je dao Nigel Pendse. On je OLAP sisteme definisao kroz akronim *FASMI* (Fast Analysis of Shared Multidimensional Information - brza analiza zajedničkih višedimenzionalnih informacija) [13]:

- Brza (Fast) – sistem je napravljen tako da korisnicima vrati većinu odgovora za

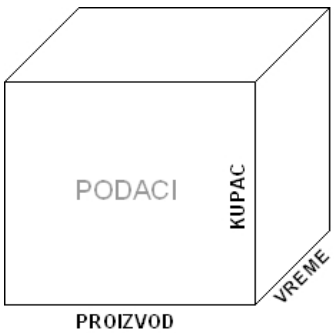
manje od pet sekundi.

- Analiza (Analysis) – sistem je prilagođen vršenju poslovnih i statističkih analiza koje su potrebne krajnjim korisnicima, a njegovo korišćenje je dovoljno jednostavno.
- Zajedničkih (Shared) – sistem implementira sve potrebne sigurnosne zahteve i, ako je potrebno, konkurentno upisivanje podataka.
- Višedimenzionalnih (Multidimensional) – sistem obezbeđuje višedimenzionalni konceptualni pogled na podatke, uključujući potpunu podršku za jednostruke i višestruke hijerarhije.
- Informacija (Information) – sistem pruža sve podatke i izvedene informacije koji su potrebni.

OLAP metode i tehnike podrazumevaju višedimenzionalne podatke. Za razliku od transakcionih skladišta podataka i DW, podaci su u OLAP sistemima podeljeni u dimenzije. *Dimenzija* je kriterijum klasifikacije podataka. Neki primeri dimenzija su: vremenska dimenzija, dimenzija kupca, dimenzija proizvoda itd. Ovaj način modelovanja podataka se zove *zvezdasta šema* (star schema) [14]. Zvezdasta šema može da izgleda kao na slici (Slika 2).



Slika 2: Višedimenzionalni podaci - zvezdasta šema



Slika 3: OLAP kocka

Na slici su prikazane tri dimenzije: vremenska, dimenzija kupca i dimenzija proizvoda. Tabela sa podacima je u sredini. Pošto svaka dimenzija ima polja koja su organizovana u vidu indeksa (npr. kvartal i godina u vremenskoj dimenziji), upiti tipa “vrati podatke o

prodaji proizvoda X za treći kvartal ove godine" se veoma brzo izvršavaju.

Svaka dimenzija se može organizovati u vidu jedne ili više *hijerarhija*. Tako, vremenska dimenzija može da se organizuje u kvartale, a kvartali u godine (vremenska hijerarhija). Kada se konkretni podaci unesu u ovaj model, nastaje tzv. *OLAP kocka* (OLAP cube, data cube) [14]. Ako kocka ima više od tri dimenzije, ona se onda zove *hiper-kocka* (hiper-cube). Primer OLAP kocke je dat na slici (Slika 3). Ivice kocke čine njene dimenzije, dok njeno telo čine podaci. Kada je potrebno napraviti izveštaj, podaci se preuzimaju na osnovu koordinata po dimenzijama direktno iz tela OLAP kocke.

Da bi korisnik mogao da dobije izveštaj koji želi, on mora na odgovarajući način da postavi upit (koji će da se kasnije izvrši na OLAP kocki). Upitni jezik koji se koristi u ove svrhe je *MDX* (Multidimensional Expressions - višedimenzionalni izrazi) [26]. MDX je za OLAP sisteme isto što i SQL za relacione baze podataka – opšteprihvaćeni standardni jezik za upite. Međutim, rezultati SQL upita su uvek tabele, a rezultat MDX naredbe se, u principu, vraća kao višedimenzionalni skup podataka (u formi kocke ili hiper-kocke), a ponekad kao tabela.

OLAP metode i tehnike mogu da se implementiraju korišćenjem različitih tehnologija. Prema tehnologiji implementacije, OLAP sistemi se dele na [16]:

- *ROLAP* (Relational OLAP) – OLAP sisteme zasnovane na relacionim bazama podataka. Podaci se izračunavaju samo delimično, a kompletno formiranje OLAP kocki se vrši po potrebi, a ne unapred.
- *MOLAP* (Multidimensional OLAP) – OLAP sisteme zasnovane na visoko optimizovanim višedimenzionalnim nizovima već unapred izračunatih vrednosti koji se nalaze u radnoj memoriji.
- *HOLAP* (Hybrid OLAP) – OLAP sisteme zasnovane na hibridnim (kombinacija relacionih i višedimenzionalnih) tehnologijama.

OLAP metode i tehnike se koriste za analizu podataka u slučajevima kada se zna koje se zakonitosti mogu očekivati i pojaviti i kada su zakonitosti relativno jednostavne, ili ako poslovni analitičari žele da steknu osnovni uvid u podatke da bi mogli da intuitivno stvore neke pretpostavke.

Statističke metode i tehnike

Kada je potrebno uočiti i dokazati neke složenije i/ili neočekivane zakonitosti u odnosima između vrednosti podataka, poslovni analitičari ponekad koriste klasične statističke metode i tehnike. Ove metode i tehnike se često mogu naći već implementirane u SPI, pa je njihova upotreba pojednostavljena. Proces statističke analize poslovnih podataka uz pomoć SPI se ukratko može opisati u nekoliko koraka:

1. Postavljanje početnih hipoteza
2. Izbor podataka
3. Izbor statističke metode ili testa
4. Sprovođenje analize tj. testa
5. Tumačenje rezultata

Da bi analiza bila uspešna, prvo je potrebno pravilno postaviti početne pretpostavke i formalizovati ih u vidu hipoteza. Sledeći korak je izbor podataka. Ulazni podaci za statističke analize se preuzimaju iz OLAP kocki ili direktno iz DW, pa je i izbor podataka ograničen na skup podataka koji se u njima nalazi. Kada se izaberu podaci, bira se i statistička metoda kojom će se dokazati tj. opovrgnuti zakonitost. Iako moderni SPI imaju u sebi implementirane razne statističke metode, za analizu poslovnih podataka se najčešće koriste metode za analizu stepena korelacije, metode za analizu klastera ili dokazivanje postojanja normalne raspodele [17][18][19].

Statističke metode i tehnike se koriste i za predviđanje budućih vrednosti. Sam proces je veoma sličan procesu statističke analize, ali ne postoje početne pretpostavke. Izvor podataka je DW ili OLAP sistem, samo vršenje predviđanja je veoma automatizovano a tehnike su: ekstrapolacija trenda, regresija itd. [17][18][19].

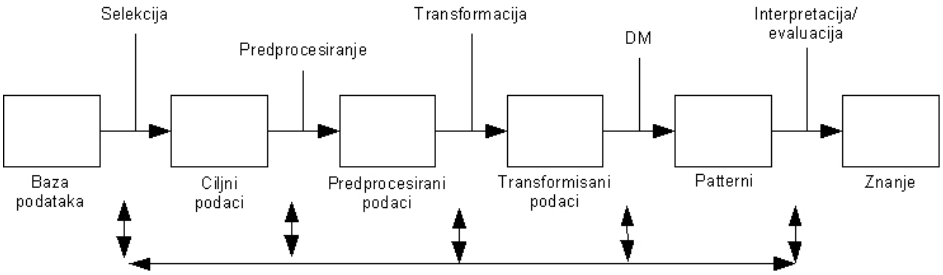
Automatsko traganje kroz podatke (Data mining)

OLAP i statističke metode i tehnike su uvek vođene od strane osobe koja vrši analizu. To znači da poslovni analitičar određuje šta će se istraživati, kako i koji se rezultati mogu očekivati. Ovaj pristup ima i jednu manu: poslovni analitičari

proveravaju samo one zakonitosti koje su im se učinile verovatnim, tj. one zakonitosti koje su im pale na pamet. Na taj način, sve implicitno prisutne ali neočekivane zakonitosti ostaju neotkrivene.

U cilju otkrivanja i ovih zakonitosti, SPI koriste metode i tehnike *automatskog traganja kroz podatke* (Data Mining - u daljem tekstu DM). "DM je netrivijalna ekstrakcija implicitnih, prethodno nepoznatih, a potencijalno korisnih informacija iz podataka" [20].

DM se smatra za oblast veštačke inteligencije, a koristi metode i tehnike iz nekoliko osnovnih disciplina: statistike, mašinskog učenja (npr. neuronske mreže), baza podataka i matematike [21]. Ovakva kombinacija je omogućila relativno autonomno pronalaženje informacija u velikim kolekcijama podataka. Pojam koji se često koristi umesto DM je *otkrivanje znanja u bazama podataka* (Knowledge Discovery in databases - u daljem tekstu KDD). KDD je širi pojam jer pored DM obuhvata i još neke faze (Slika 4).



Slika 4: KDD proces - preuzeto iz [22]

Na početku KDD procesa se nalazi baza podataka sa transakcionim podacima. Posle selekcije, dobijaju se ciljni podaci tj. oni koji su bitni za proces otkrivanja znanja. Ciljni podaci moraju da prođu fazu predprocesiranja u kojoj se uklanjaju duplikati, dopunjavaju ili ispravljaju vrednosti koje su van opsega ili nedostaju itd. Transformacija predprocesiranih podataka podrazumeva uklanjanje onih atributa i dimenzija koji nisu potrebni za otkrivanje znanja. Tek kada se dobiju transformisani podaci, može da se pristupi procesu DM. Ovaj proces kao izlaz daje određene zakonitosti koje se pojavljuju među podacima. Kada poslovni analitičar interpretira (protumači šta znače) i evaluira (proceni važnost) ove pattern-e, nastaje znanje. Prema tome, DM je samo jedna od faza

KDD, jer KDD pored DM obuhvata i faze selekcije, predprocesiranja, transformacije i interpretacije. Ako se malo bolje pogleda Slika 4, može se uočiti da su selekcija, predprocesiranje i transformacija podataka aktivnosti koje moderni ETL alati i DW već rade u okviru SPI. Zbog toga se, u kontekstu SPI, retko kada koristi pojam KDD već samo DM.

Postoji više mišljenja u odnosu na to gde tačno počinje proces DM u okviru KDD procesa i šta tačno sadrži. Prema prethodnom primeru, to se dešava posle transformacije podataka. Neki autori malo drugačije gledaju na KDD proces i na ulogu DM [21]. Prema ovom pogledu na KDD proces, DM počinje transformacijom podataka (redukovanje i projekcija) a obuhvata i tumačenje rezultata.

Bez obzira na to koja verzija KDD procesa se posmatra, proces DM uvek obuhvata dva ključna koraka:

1. Izbor DM zadatka
2. Izbor DM algoritma

Izborom DM zadatka se definiše šta se želi postići korišćenjem DM-a. Postoji nekoliko osnovnih vrsta DM zadataka [23]:

- Klasifikacija
- Klasterizacija
- Sumiranje
- Modeliranje zavisnosti
- Detekcija odstupanja
- Određivanje asocijativnih pravila
- Određivanje karakterističnih putanja
- Analiza sekvenci

Izborom DM algoritma se definiše algoritam koji će se koristiti u DM procesu. Izabrani algoritam mora da bude u skladu sa prethodno izabranim ciljem tj. sa DM zadatkom, a mnogi algoritmi mogu da se koriste uspešno i za više zadataka. Ovi algoritmi implementiraju i koriste u radu metode i tehnike iz statistike, matematike i mašinskog

učenja, a neki od najpoznatijih su ([22] i [23]): ID3, C4.5, SLIQ, nearest neighbor, Naive-Bayes, OODG, Lazy Decision trees, Decision table, CDP, Algoritam Apriori, DHP, Algoritam AprioriAll, Algoritam AprioriSome, Generički DM algoritam i MPTP.

Izlaz iz DM procesa su *paterni* (pattern) koji predstavljaju automatski uočene zakonitosti među podacima. Naravno, nije svaki patern bitan – oni mogu biti i trivijalni, već poznati, neupotrebljivi, nepouzdani, previše složeni itd. Zbog toga se uvek vrši njihova interpretacija i evaluacija. Da bi mogli da se interpretiraju i evaluiraju, paterne je potrebno predstaviti na neki način koji je razumljiv osobi koja vrši interpretaciju. Tri najčešće korišćene tehnike za predstavljanje paterna su: pravila, stabla odlučivanja i različite tehnike vizuelizacije [22][23].

Kao što je rečeno, DM kao izvore podataka najčešće koristi podatke iz DW ili iz OLAP sistema. Da bi korisnik mogao da pusti željenu DM analizu ili predviđanje da se izvrši, on mora na odgovarajući način da postavlja upite i parametre analize. Upitni jezik koji se koristi u ove svrhe je *DMX* (Data Mining Extensions) koji je razvijen u Microsoftu [24].

2.1.3 Metode i tehnike za prikazivanje podataka i pretvaranje podataka u informacije

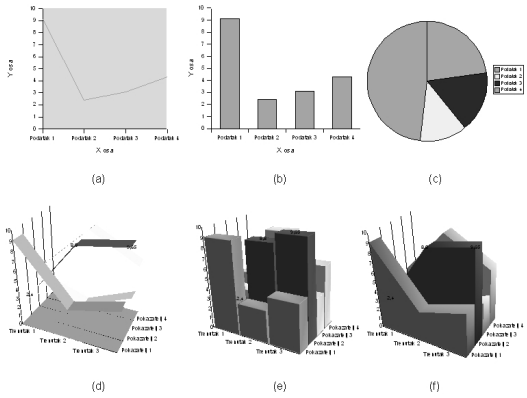
Kada se vrši analiza podataka, ključna karika u celom procesu je čovek. Da bi on mogao da stvori sliku o poslovanju i izvuče informacije na osnovu podataka, potrebno je da podaci budu predstavljeni na pogodan način. U tom slučaju, koriste se uobičajene metode i tehnike za predstavljanje podataka: tabele i grafici. Međutim, postoje i druge metode i tehnike u kojima se donekle automatizovano pretvaraju podaci u informacije. U pitanju su *balansirane kartice rezultata* (Balanced Scorecards) i *komandne table* (Dashboards).

Tabele i grafici

Tabele (spreadsheet) predstavljaju jednu od najstarijih tehnika za predstavljanje podataka. Iako se, u principu, koriste i za manipulaciju nad podacima, njihova izvorna upotreba je vezana za izveštavanje. Zbog njihove popularnosti i jednostavnosti, danas

postoje mnogi softverski paketi koji isključivo podržavaju rad sa tabelama. U kontekstu modernih SPI, tabele se često koriste da prikažu podatke koji su rezultat OLAP upita, statističke analize, transformacije i predprocesiranja podataka za DM i sl.

Još jedna od klasičnih tehnika za prikazivanje podataka su *grafici* (chart). Grafik je vizuelna predstava podataka najčešće u dve ili tri dimenzije. Cilj je da se korisniku prikažu podaci tako da on lako uoči relacije između njih i formira informacije. Postoji više vrsta grafika, ali se oni grubo mogu podeliti u dve grupe: dvodimenzionalne i trodimenzionalne grafike (Slika 5 - grafici (a), (b) i (c) su dvodimenzionalni, a (d), (e) i (f) trodimenzionalni). Potrebno je napomenuti da se i na dvodimenzionalnim i na trodimenzionalnim graficima mogu prikazivati samo jednodimenzionalni (niz) ili dvodimenzionalni (tabela) nizovi podataka.



Slika 5: Različite vrste grafika

Balansirane kartice rezultata (Balanced Scorecards)

Balansirane kartice rezultata potiču iz menadžmenta i to kao naslednik *sistema totalnog upravljanja kvalitetom* (Total Quality Management – TQM). Robert S. Kaplan i David P. Norton su ih uveli kao sredstvo koje bi trebalo da omogući da se poslovni podaci lakše pretvore u informacije i povežu sa vizijom i strategijama [25]. "Balansirane kartice rezultata – integrisani sistem za opisivanje strategije poslovanja korišćenjem međusobno povezanih pokazatelja uspešnosti podeljenih u četiri perspektive koje bi trebalo da budu izbalansirane – finansijske, klijentske, perspektive internih procesa i perspektive učenja i rasta. Balansirane kartice rezultata imaju

karakteristike sistema merenja, menadžment sistema i sredstva za komunikaciju.” [25]

Ceo proces BKR je cikličan i može se podeliti u četiri koraka [26]:

1. PLANIRANJE – prevođenje vizije i strategija u operativne ciljeve i povezivanje sa individualnim pokazateljima uspešnosti;
2. IZVRŠAVANJE – izvršavanje konkretnih poslovnih procesa i aktivnosti;
3. PROVERA – prikupljanje vrednosti za izabrane pokazatelje uspešnosti, stvaranje informacija i provera ostvarenja vizije i strategije;
4. AKCIJA – prilagođavanje strategije, vizije ili preduzimanje konkretnih akcija da se strategija i vizija ostvare. Samo povezivanje vizije i strategije u okviru neke perspektive sa konkretnim podacima se vrši u četiri koraka: definisanje operativnih ciljeva, definisanje pokazatelja - *ključnih pokazatelja uspešnosti* (Key Performance Indicator – KPI u daljem tekstu), određivanje ciljnih vrednosti i formiranje inicijativa.

BKR metoda je, u nešto izmenjenom obliku, našla svoju primenu u modernim SPI kao sredstvo za pretvaranje podataka u informacije. Princip funkcionisanja je isti, ali su neki koraci automatizovani. Kada se izvrši faza planiranja (definisanje ciljeva, pokazatelja, ciljnih vrednosti i inicijativa) podaci o svim elementima koji su rezultat ove faze se unesu u SPI (ciljevi, pokazatelji, ciljne vrednosti i inicijative). Po završetku faze izvršavanja, prelazi se na fazu provere. Suština je u automatizaciji faze provere: vrednosti izabranih pokazatelja se izračunavaju uz pomoć OLAP kocki, proveravaju se njihove vrednosti u odnosu na ciljno zadate i formira se izveštaj koji opisuje po kojim perspektivama i konkretnim ciljevima je preduzeće bilo uspešno, a po kojim ne. Sam izveštaj je najčešće u grafičkoj formi gde je svaki cilj i pokazatelj predstavljen nekom raznobojnom skalom i pokazateljem nivoa. Na ovaj način, menadžeri mogu da, već na prvi pogled, uoče šta nije u redu.

Komandne table (Dashboards)

Klasični grafici i tabele su korisni za prikazivanje podataka, međutim problem nastaje ako ih ima mnogo. *Komandne table* (dashboards) su nastale kao potencijalno

rešenje. U analogiji sa komandnom tablom automobila, one bi trebalo da pruže tačan uvid u trenutno stanje (procesa, poslovanja...) na osnovu samo jednog pogleda. "Komandna tabla je grafički prikaz najvažnijih informacija potrebnih za postizanje jednog ili više ciljeva. Ove informacije su združene i prikazane na jednom ekranu tako da se mogu proveriti i videti jednim pogledom" [27]. Komandne table su čitavi logički modeli za pretvaranje podataka u informacije i prikaz samo bitnih informacija. Njihove glavne karakteristike su [28]:

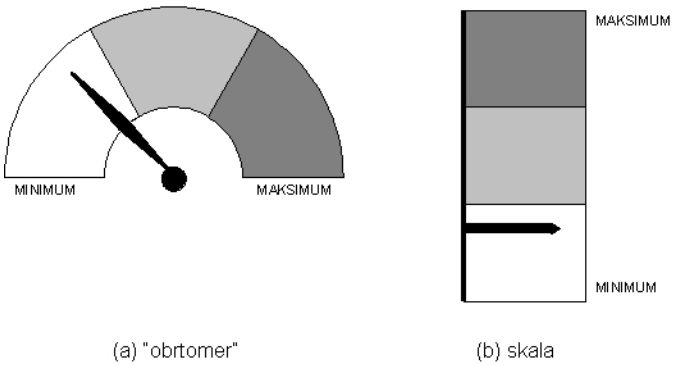
- Intuitivni grafički prikaz aranžiran u nekom unapred smišljenom redosledu koji je istovremeno interaktivan i lak za navigaciju kao web browser.
- Logička struktura koja omogućava lak pristup informacijama.
- Obuka za korišćenje je minimalna ili nije potrebna.
- Prilagodljivost prikaza potrebama svakog korisnika.
- Često i redovno osvežavanje informacija radi održavanja tačnosti.
- Mogućnost istovremenog prikaza informacija iz više izvora, sektora odeljenja ili tržišta.

Neki autori navode samo tri osnovne karakteristike komandnih tabli [27]:

- Sažeti prikazi.
- Jasan, jednostavan i intuitivan mehanizam prikazivanja.
- Prilagodljivost.

Pretvaranje podataka u informacije u okviru komandnih tabli je automatizovano korišćenjem graničnih vrednosti, alarma i pravila. Ideja je da se bilo kakvo odstupanje vrednosti KPI označi nekim pravilom ili alarmom koji će da se oglasi ako se odstupanje zaista desi.

Grafički prikaz komandne table bi trebalo da bude takav da sve potrebne informacije i podaci stanu na jedan ekran, a da ekran i dalje bude pregledan tj. da se uvid u trenutno stanje može ostvariti jednim pogledom. Za prikaz se koriste tabele, grafici, skale, instrumenti nalik na obrtomere itd. Mnogi SPI koriste skale i obrtomere jer se već na prvi pogled vidi gde je vrednost u odnosu na dozvoljene tj. granične vrednosti (Slika 6).



Slika 6: Grafički elementi za predstavljanje KPI u okviru komandnih tabli

2.1.4 Arhitektura

Metode i tehnike opisane u prethodnom poglavlju se implementiraju u okviru SPI u formi više međusobno povezanih modula. Logička arhitektura savremenih SPI je prikazana na slici (Slika 7). SPI se najčešće implementiraju kao distibuirane aplikacije.

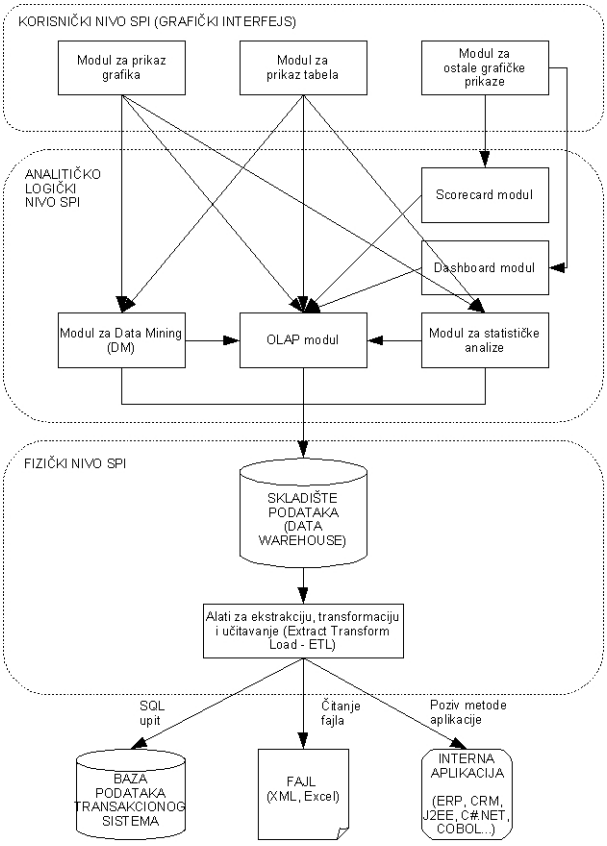
Fizički nivo čine ETL alati koji prikupljaju sirove podatke iz transakcionih baza podataka, fajlova ili internih aplikacija putem odgovarajućih poziva i DW u koji se potom smeštaju podaci.

Analitičko-logički nivo SPI čine: OLAP, DM, statistički modul, Scorecard modul i Dashboard modul. OLAP modul preuzima podatke direkno iz DW i sumira ih na taj način da budu pogodni za različite vrste upita – formira predefinisane dimenzije i odgovarajuće OLAP kocke i popunjava ih izračunatim pokazateljima. Statistički modul sadrži implementacije statističkih metoda i tehnika, a DM modul preuzima podatke iz OLAP modula ili direktno iz DW i vrši za automatsko traganje kroz podatke. Scorecard modul koristi dve grupe podataka. Prvu grupu čine oni podaci koje korisnik unese a tiču se poslovnih perspektiva, definicija ključnih pokazatelja i njihovih ciljnih vrednosti. Druga grupa podataka su konkretne vrednosti KPI. Ove vrednosti se preuzimaju iz OLAP kocki. Potrebno je napomenuti da ovaj modul sadrži samo logičku (a ne i grafičku) reprezentaciju BKR. Grafički prikaz izlaza iz scorecard modula se formira u okviru korisničkog nivoa. Dashboard modul je unekoliko sličan scorecard modulu jer koristi dve grupe podataka. Jednu grupu čine definicije ključnih pokazatelja sa njihovim

graničnim vrednostima, a drugu konkretne vrednosti za te pokazatelje. Konkretne vrednosti se preuzimaju iz OLAP kocki. Dashboard modul takođe sadrži samo logičku (a ne i grafičku) reprezentaciju komandnih tabli.

Korisnički nivo SPI čini odgovarajući grafički interfejs. Konkretna implementacija modula na ovom nivou je najčešće u vidu web aplikacija. Moduli koje grafički interfejs diskretnih SPI mora da ima su: modul za prikaz tabela, modul za prikaz grafika i modul za ostale grafičke prikaze.

SPI najčešće koriste *pull* model komunikacije između komponenti. To znači da moduli sa višeg nivoa pozivaju module sa nižeg nivoa na zahtev korisnika (videti smer strelica – Slika 7) tj. korisnik je inicijator akcije sistema. Međutim, postoje i oni SPI koji implementiraju *push* model komunikacije, ali o tome će biti više reči u poglavlju o trendovima u oblasti SPI.



Slika 7: Arhitektura SPI

2.1.5 Trendovi

U oblasti SPI se, u poslednjih nekoliko godina, mogu uočiti sledeća tri trenda (tendencije):

- Tendencija prelaska na izveštavanje u realnom vremenu
- Tendencija spajanja sa sistemima za podršku odlučivanju
- Tendencija povećanja aktivnosti i samostalnosti putem povećanja količine ugrađenog poslovnog znanja

Sve tendencije se najbolje mogu opisati idejom koja stoji iza njih. Ta ideja je da se smanji vremenski period od formiranja informacija do preduzimanja akcija. Koncept kojim se opisuje ovaj vremenski period je akciona distanca. "Akciona distanca je kašnjenje (vremenski razmak) između generisanja informacija od strane SPI i preduzimanja skupa akcija koje odgovaraju toj specifičnoj poslovnoj situaciji. To je mera napora koji je potrebno uložiti da bi se informacije protumačile i pretvorile u konkretne akcije koje odgovaraju datim informacijama" [29]. Prema istom autoru, tri faktora utiču na akcionu distancu:

- Kašnjenje u podacima (data latency) - vreme koje prođe između obavljanja poslovne transakcije i trenutka kada su podaci iz te transakciji dostupni za analizu.
- Kašnjenje u analizi (analysis latency) - vreme od početka analize, preko sređivanja rezultata do trenutka dostavljanja rezultata analize onima kojima su potrebni.
- Kašnjenje u odlučivanju (decision latency) - vreme potrebno za razumevanje informacija i reagovanje na odgovarajući način.

Prelazak na izveštavanje u realnom vremenu se zasniva na ideji da se, što je moguće više, smanji kašnjenje u podacima, a donekle i u analizi. Osnovni pojam koji se koristi da opiše ovakve SPI je *poslovna inteligencija u realnom vremenu* (Real-Time Business Intelligence) [30][33][34][35]. U nekim tekstovima se koristi i pojam *poslovna inteligencija 2.0* (Business Intelligence 2.0) [31]. Ideja ovih SPI je da se spoji praćenje

poslovnih procesa i klasična PI u cilju smanjenja akcione distance. Karakteristike ovih sistema su [31]:

- Glavni pokretači akcija sistema su događaji.
- Rad u realnom vremenu.
- Automatizovana analiza.
- Orijentisanost ka budućnosti.
- Orijentisani ka procesima.
- Skalabilni.

U ovim sistemima se čak i model komunikacije komponenti SPI menja u *push* model, pa su i moduli koji se koriste drugačiji. Drugim rečima, sakupljanje, transformacija i učitavanje podataka se ne vrši u predefinisanim vremenskim intervalima ili na poziv, već svaki put kada se ti podaci promene. ETL se alat aktivira svaki put kada se desi odgovarajući događaj u bazi podataka ili u nekoj internoj aplikaciji i prenosi se putem *trigera (database trigger)* ili poziva od aplikacije ka ETL alatu. *IM-DW* modul (In Memory DW – skladište podataka u radnoj memoriji) funkcioniše slično kao i običan DW modul sa tom razlikom što, da bi promene u podacima mogle da se odigraju dovoljno brzo, celo skladište podataka mora da se nalazi u radnoj memoriji. *RT-OLAP* modul (Real Time OLAP – OLAP u relanom vremenu) je modul koji implementira OLAP funkcionalnost, ali je prilagođen radu u realnom vremenu. Dashboard modul se aktivira pozivom iz RT-OLAP modula. Ovaj modul onda preuzima podatke iz OLAP kocki i popunjava svoj logički model. Korisnički nivo SPI u realnom vremenu čini odgovarajući grafički interfejs, a razlika u odnosu na isti nivo običnih SPI je samo u načinu poziva modula. U ovom slučaju, module ne poziva korisnik već RT-OLAP ili Dashboard modul. Posledica toga je da grafički moduli uvek prikazuju trenutno tj. ažurno stanje i ažurne podatke.

Spajanje sa sistemima za podršku odlučivanju je još jedan trend [6] ali, je praćen i trendom **težnje ka većoj aktivnosti putem povećanja količine ugrađenog poslovnog znanja**. Sve to se radi u cilju pružanja podrške za celokupne procese izveštavanja i odlučivanja tj. smanjivanja kašnjenja u analizi i kašnjenja u odlučivanju. Sa obzirom na

to da su ova dva trenda međusobno povezana, tako će biti i razmatrana i objašnjena.

U suštini, posledica ugradnje sve veće količine znanja u SPI je i približavanje oblasti veštačke inteligencije i PI tj. korišćenje metoda i tehnika veštačke inteligencije u konkretnim SPI kao npr: DM, neuronskih mreža, ekspertnih sistema itd. [6].

Već je napomenuto da se u SPI koristi DM. Međutim, sada se tradicionalne DM metode nadograđuju i kombinuju sa domenskim znanjem [37], a sve to u cilju automatizovanja celog KDD procesa, njegovog prilagođavanja za neki konkretan domen i jednostavnijeg izveštavanja. Takođe, *zaključivanje po slučajevima* (case-based reasoning) i *ontologije* (ontologies) se koriste u cilju stvaranja inteligentnih DM asistenata čija je uloga da olakšavaju izbor DM metoda, zadataka ili algoritama poslovnim analitičarima sa manje iskustva [38]. Konačno, postoje pokušaji da se naprave i formalne strukture za otkrivanje znanja i komunikaciju tog znanja kroz različite delove organizacije [39].

I fuzzy ekspertni sistemi se koriste u kombinaciji sa SPI, ali najčešće za neke specifične poslove. Na primer, jedan takav sistem se koristi za davanje preporuke koje akcije kupiti na berzi [40]. On analizira razne finansijske pokazatelje i stvara portfolio akcija koje su pogodne za investiranje. Takođe, postoji i fuzzy ekspertni sistem koji procenjuje poverenje klijenata u internet marketingu i svrstava ih u nekoliko kategorija prema ocenjenom nivou poverenja [41]. Na kraju, fuzzy ekspertni sistemi mogu da se koriste i u kombinaciji sa BKR u cilju pružanja informacija o tome da li je preduzeće postiglo zadate ciljeve ili ne [42].

Još jedan popularan izraz je nastao sa ciljem da opiše konkretan spoj SPI i veštačke inteligencije. U pitanju je *prilagođavajuća poslovna inteligencija* (Adaptive Business Intelligence). U suštini to je “disciplina u kojoj se kombinuju predviđanje, optimizacija i prilagođavanje (autonomno) i ugrađuju u jedan sistem koji onda može da odgovori na dva osnovna pitanja: šta će se (najverovatnije) desiti u budićnosti i koja je najbolja odluka u ovom trenutku” [36]. Ideja je da se upotrebe različite metode i tehnike SPI u kombinaciji sa neuronskim mrežama, DM, genetičkim algoritmima i inteligentnim agentima i to u cilju pružanja odgovora na ova pitanja.

2.2 Ekspertni sistemi

Ekspertni sistemi su nastali u okviru veštačke inteligencije i dobro su istraženi kao oblast, što se vidi i iz definicija – relativno su konzistentne i konvergiraju ka istom pojmu. Neke od definicija ES su:

"Vrsta programa koja donosi odluke ili rešava probleme iz određene oblasti korišćenjem znanja i analitičkih pravila koja su definisali eksperti za datu oblast." [43]

"ES je kompjuterski program koji je projektovan tako da oponaša sposobnost rešavanja problema eksperta (čoveka) za neku oblast." [44]

"Ekspertni sistem je kompjuterski program koji koristi ugrađeno znanje iz neke specifične oblasti i donosi zaključke na osnovu njega, a u cilju rešavanja problema ili pružanja saveta." [45]

Da bi neki sistem mogao da se nazove ES on mora da zadovolji tri uslova [45]: mora da sadrži znanje, to znanje mora biti iz neke konkretne oblasti (*domena*) i mora da poseduje sposobnost samostalnog rešavanja problema (zaključivanja i donošenja odluka).

Najpoznatija klasifikacija ES je prema načinu predstavljanja znanja. Prema ovom kriterijumu ES se dele na [44]:

- ES zasnovane na pravilima (Rule-based ES)
- ES zasnovane na okvirima (Frame-based ES)
- Hibridne ES (Hybrid ES)

ES se mogu podeliti i prema osnovnom cilju (u odnosu na eksperta). Osnovni cilj ES može da bude da pomogne ekspertu u radu ili da zameni eksperta [44]. Prema tome, postoje dve vrste ES: ES koji pomažu ekspertima i ES koji zamenjuju eksperte.

Metode i tehnike koje ES koriste u svom radu se mogu podeliti u nekoliko grupa u skladu sa funkcijom koju obavljaju:

- Metode i tehnike za predstavljanje znanja
- Metode i tehnike za predstavljanje neizvesnog znanja

- Metode i tehnike za zaključivanje
- Metode i tehnike za formiranje objašnjenja

2.2.1 Metode i tehnike za predstavljanje znanja

Da bi ES mogao da formira zaključke, on mora da poseduje domensko znanje i ono mora biti strogo formalizovano. Domensko znanje je znanje koje ekspert (čovek) poseduje a odnosi se na konkretnu, usko specifičnu oblast (domen). ES i drugi inteligentni sistemi sadrže eksplicitno predstavljeno znanje koje je lako izmenljivo, dopunjivo i potpuno je nezavisno od mehanizama kontrole [46]. Postoje sledeće vrste znanja [44]:

- Proceduralno znanje – znanje o tome kako se neki problem rešava.
- Deklarativno znanje – može se posmatrati kao skup kratkih iskaza koji su tačni ili netačni i odnose se samo na taj problem.
- Meta znanje – znanje o znanju.
- Heurističko znanje – iskustvena pravila ili situacije koje eksperti znaju i koriste.
- Strukturno znanje – opisuje domenske koncepte, objekte i njihove atribute i veze.

Prema sličnoj podeli [47], pored navedenih, uvode se i sledeće vrste znanja:

- Nasledivo znanje (relacijom nasleđivanja se dobija znanje)
- Izvedeno znanje
- Relaciono znanje
- Zdravorazumsko znanje (iskustveno, heurističko znanje koje je generalizovano i nezavisno od domena)
- Eksplicitno znanje
- Implicitno znanje
- Neizvesno znanje (kada se činjenice predstavljaju sa određenom dozom izvesnosti)

Pravila (Rules)

Pravila su jedna od najčešće korišćenih metoda za predstavljanje znanja u okviru ES. "Pravilo je struktura za predstavljanje znanja koja uspostavlja vezu između neke poznate informacije i informacije za koju se, na osnovu tačnosti prve, može zaključiti da takođe važi." [44]

Teorijska zasnovanost ove metode počinje na *iskaznoj logici* i na njenom proširenju - *predikatskom računu*. Pravila su, u stvari, pojednostavljena računarska implementacija predikatskog računa, a jedna od osnovnih tehnika zaključivanja je *modus ponens*. Primer pravila je:

IF Auto neće da upali **AND** Farovi ne rade
THEN Akumulator je prazan

Pravila se sastoje iz dva dela: *IF* (ako) i *THEN* (onda). U IF delu pravila se nalazi jedan ili više iskaza na osnovu kojih se izvode zaključci. Ovi iskazi se nazivaju *premise*. Ako pravilo sadrži više premisa onda su one povezane nekom logičkom operacijom. U prethodnom primeru, to je bila logička operacija *I* (AND), ali se često može naći i *ILI* (OR) i *NE* (NOT) logička operacija. THEN deo pravila takođe sadrži iskaze, ali se oni zovu *zaključci*. Princip funkcionisanja je identičan modus ponensu – ako su premise iz IF dela pravila tačne (ako ih ima više potrebno je da ukupan logički izraz bude tačan) onda su i zaključci iz THEN dela pravila tačni.

Svaki ES sadrži više pravila. Zaključci jednog pravila najčešće predstavljaju premise drugih pravila. Ovaj princip po kome se pravila međusobno logički nadovezuju (zaključak jednog pravila je premisa drugog) se zove *ulančavanje pravila* (rule chaining) [44]. Na primer:

Pravilo 1	**Pravilo 2**
IF Auto neće da upali **AND** Farovi ne rade	**IF** Akumulator je prazan
THEN Akumulator je prazan	**THEN** Napuni akumulator

Pravila se koriste za opisivanje proceduralnog znanja jer ukazuju na način rešavanja

nekog problema i izvođenja zaključaka o njemu. ES koji za predstavljanje znanja koriste samo pravila se nazivaju *ES zasnovani na pravilima* (Rule-based ES).

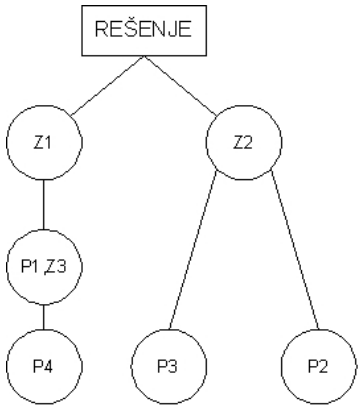
Poseban problem koji se pojavljuje kod ES zasnovanim na pravilima je održavanje *konzistentnosti baze znanja* (consistency maintenance). Nekonzistentnost baze znanja može da nastane zbog sledećih anomalija [48]:

- Nezadovoljivo pravilo – Pravilo čija premisa nikad ne može biti zadovoljena.
- Beskorisno pravilo – Pravilo čijim izvršavanjem se ne postiže ništa.
- Podrazumevano pravilo – Postoji neko opštije pravilo koje obuhvata njegove premise i zaključke.
- Redundantno pravilo – Pravilo kojene doprinosi zaključivanju nijedne ciljne činjenice u toj bazi znanja.
- Nekonzistentan par pravila – Dva pravila čije premise mogu biti zadovoljene u istoj situaciji ali vode ka suprotstavljenim zaključcima.
- Nekonzistentan skup pravila – Ako izvršavanje skupa pravila dovodi do suprotstavljenih zaključaka.
- Cikličan skup pravila – Nijedno pravilo ne može da se izvrši dok se ne izvrši neko dr. pravilo iz skupa.
- Nekorišćene ulazne vrednosti.
- Nekompletan skup pravila – Postoje određena ciljna stanja ili zaključci do kojih se ne nikada može stići.

Mreže zaključivanja (Inference networks)

Sistemi pravila mogu da budu i jako komplikovani, a ljudi uglavnom vole da imaju jasnu sliku o problemu koji rešavaju. Kada se čvorovi grafa koriste da predstave premise i zaključke pravila, a grane logičke veze između njih nastaju *mreže zaključivanja* (inference networks) [44]. Primer jedne ovakve mreže je dat na slici (Slika 8). Sistem pravila koji je predstavljen putem mreže je:

IF P1	**IF** P2 **OR** P3	**IF** P4
THEN Z1	**THEN** Z2	**THEN** Z3 (pri čemu je Z3 = P1)



Slika 8: Primer mreže zaključivanja

Semantičke mreže (Semantic networks)

Pravila i mreže zaključivanja služe za predstavljanje proceduralnog znanja. Da bi se domensko znanje na kvalitetan način formalizovalo i unelo u ES, potrebno je na formalan način opisati i osnovne pojmove i koncepte. Metoda koja se često koristi u ove svrhe je metoda *semantičkih mreža* (semantic networks).

"Semantičke mreže su metoda za predstavljanje znanja uz pomoć grafova, pri čemu tačke na grafu predstavljaju objekte tj. koncepte a lukovi veze između njih." [44]

"Sem. mreža je grafička notacija za predstavljanje znanja uz pomoć mreže povezanih tačaka i lukova." [62]

Prema [62], i mreže zaključivanja su, u stvari, semantičke mreže i to tzv. *implikacione mreže* (implicational networks), a mreže koje opisuju strukturno znanje se zovu *definicijske mreže* (definitional networks).

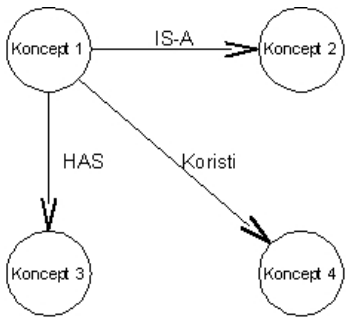
U semantičkoj mreži, tačke predstavljaju *koncepte* tj. objekte. Koncept može biti bilo šta, na primer: mačka, auto, kuća itd. Lukovi predstavljaju veze između koncepata. Ove veze su usmerene i dele se na tri osnovne vrste:

- IS–A veza – to je slično vezi nasleđivanja. Kada važi "O2 IS-A O1" to znači da koncept O2 ima sve karakteristike koncepta O1 i neke dodatne.
- HAS veza – slično vezi agregacije. Kada važi "B HAS A", sledi da koncept B

sadrži u sebi i koncept A.

- Ostale veze – odnose se na bilo kakve druge asocijacije.

Semantička mreža sa slike (Slika 9) upravo ilustruje sve opisane elemente. Koncept 1 nasleđuje Koncept 2 tj. ima sve njegove karakteristike i još neke. Koncept 1 takođe sadrži u sebi i Koncept 3, a koristi Koncept 4.

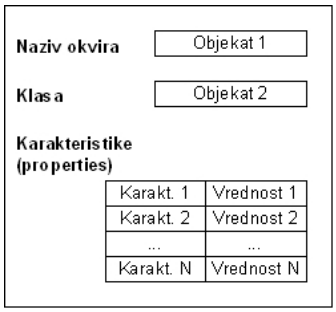


Slika 9: Semantička mreža

Okviri (Frames)

Okviri (frames) su preteča savremenih klasa iz objektno orijentisanog programiranja. Okvire je izmislio čuveni Marvin Minsky [50] sa osnovnom idejom da omoguće predstavljanje strukturalnog, proceduralnog i deklarativnog znanja. "Okviri su strukture podataka koje služe za predstavljanje stereotipnog znanja o nekom konceptu ili objektu" [44]. Na sledećoj slici je predstavljen jedan okvir (Slika 10).

Svaki okvir sadrži naziv. U ovom slučaju naziv je "Objekat 1". Ovaj naziv se odnosi na konkretnu instancu i razlikuje je od svih ostalih instanci iste klase. Klasa predstavlja skup zajedničkih karakteristika niza sličnih objekata (potpuno isto kao u objektno orijentisanom pristupu). Primer okvira sa slike pripada klasi "Objekat 2". *Karakteristike* (properties) su atributi okvira i imaju konkretne vrednosti. Još jedan naziv koji se koristi da opiše karakteristike je *slot* (slot). Svaki slot ima svoju vrednost. Slotovi mogu biti *statički* i *dinamički*. Statički slotovi imaju vrednost koja se ne menja u toku vremena. Dinamički slotovi su upravo suprotni od toga.



Slika 10: Primer okvira

Veze između okvira su takođe moguće. Okviri podržavaju vezu nasleđivanja, agregacije i asocijacije. Agregacija i asocijacije se zapravo implementiraju kao slotovi pri čemu je vrednost slota pokazivač prema nekom drugom okviru.

Okviri imaju i elemente koji definišu neku vrstu ponašanja. Ovi elementi se nazivaju *facet* (koristiće se originalni naziv u nedostatku adekvatnog prevoda). Tako, za svaki slot postoje dva facet-a: *IF-NEEDED* i *IF-CHANGED*. IF-NEEDED facet se aktivira ako je potrebno preuzeti vrednost za neki slot. On može da sadrži komande ili pravila koja definišu kako se dolazi do vrednosti. IF-CHANGED facet se aktivira ako se vrednot slota promeni. Ovaj facet najčešće sadrži komande za proveru da li je nova vrednost u okviru granica.

U jednom trenutku, u okvire su uvedene i metode. Metode predstavljaju elemente koji opisuju ponašanje karakteristično za okvir i mogu se po potrebi pozvati i izvršiti. Komunikacija između okvira se može ostvariti pozivanjem metoda jednog okvira iz metode drugog okvira.

Okviri se, kao metoda za predstavljanje znanja, mogu koristiti sami ili u kombinaciji sa pravilima. Kada se koriste sami, ES napravljen na taj način pripada vrsti *ES zasnovanim na okvirima* (Frame-based ES). Ako se koriste u kombinaciji sa pravilima, u pitanju je *hibridni ES* (Hybrid ES). Danas su okviri skoro potpuno napušteni jer su ih zamenili objekti tj. klase.

Trojke objekat-atribut-vrednost (OAV triplets – Object Attribute Value)

Trojke objekat-atribut-vrednost (object attribute value triplets) su metoda za predstavljanje deklarativnog znanja. ES zasnovani na pravilima koriste obične promenljive za formiranje premisa i zaključaka. Problem nastaje kada promenljivih ima mnogo, jer nemaju nikakvu strukturu niti su grupisane.

Da bi se problem rešio uvedene su OAV trojke [44]. Promenljive se, na ovaj način, grupišu kao atributi nekih objekata. Primer pravila u kojem se činjenice izražavaju korišćenjem OAV trojki je:

IF Auto.verglanje = TRUE **AND** Auto.radi_jedan_far = TRUE
THEN Rešenje.pregorela_je_sijalica = TRUE

OAV trojke se mogu koristiti jedino u kombinaciji sa pravilima. Može se reći da predstavljaju dosta pojednostavljenu verziju okvira jer ne podržavaju asocijacije, veze nasleđivanja, agregaciju niti predstavljanje proceduralnog znanja. U modernim ES su i OAV trojke takođe napuštene, jer su ih zamenili objekti tj. klase.

Objekti (Klase)

Osnovna ideja objekata i klasa je da se napravi odgovarajuća struktura koja bi u sebi opisivala neko stanje (karakteristike), ponašanje ali i odnose (relacije). “Klasa je je opšti predstavnik nekog skupa objekata koji imaju istu strukturu i ponašanje. Ona predstavlja pojednostavljenu sliku ovih objekata, a obuhvata njihove karakteristike, ponašanje i relacije sa drugim klasama. Objekat je konkretan primerak neke klase.” [51]

Međutim, ideja klase nije nastala odjednom. Preteča klasa su bili okviri koji su imali veoma slične karakteristike ali nisu mogli da u potpunosti opišu ponašanje. Kada su klase i OO pristup postali popularni, odmah je postalo jasno da se mogu koristiti i kao metoda za predstavljanje znanja u okviru ES jer mogu da opišu strukturno, proceduralno i deklarativno znanje.

Naziv klase je ono što razlikuje jednu klasu od druge i mora da bude jedinstven. Atributi

klase predstavljaju njene karakteristike, a njihove konkretne vrednosti predstavljaju trenutno stanje objekta klase. Metode klase opisuju ponašanje koje klasa može da ispolji. Kao i kod okvira i klase mogu imati veze nasleđivanja, agregacije i asocijacije.

U modernim ES, klase se uvek koriste u kombinaciji sa pravilima. Atributi i metode klasa služe za formiranje premisa i zaključaka pravila. Glavna prednost korišćenja klasa je ta što se ovakav ES može lako integrisati u neki već postojeći program (napravljen u nekom objektno-orijentisanom programskom jeziku).

2.2.2 Metode i tehnike za predstavljanje neizvesnog znanja

Kada je potrebno predstaviti neizvesno znanje u okviru ES, koriste se drugačije metode od prethodnih. U pitanju su sve one situacije u kojima ne postoji dovoljno činjenica ili znanja da bi se moglo tvrditi da je neko rešenje ili zaključak sigurno tačno. Problem definisanja neizvesnosti u okviru ES se generalno deli na dve situacije: situaciju kada su poznate verovatnoće važenja zaključaka na osnovu premisa i situaciju kada nisu poznate verovatnoće važenja zaključaka na osnovu premisa.

U prvom slučaju se koristi Bayes-ova formula za izračunavanje uslovnih verovatnoća, dok se u drugom slučaju koriste ili *teorija izvesnosti* (certainty theory) ili *fuzzy* teorija.

Bayes-ova teorema i teorija verovatnoće

Osnovni koncept koji teorija verovatnoće uvodi je broj P(E) koji predstavlja verovatnoću da se neki događaj E zaista i desi u toku izvođenja nekog slučajnog eksperimenta. Ovaj broj ima vrednost između nula i jedan pri čemu se smatra da događaj ne može da se desi ako je verovatnoća nula, a da je sigurno da će da se desi ako je verovatnoća jedan. Uslovna verovatnoća podrazumeva verovatnoću da se neki događaj A desi ako se već desio događaj B. Formula za uslovnu verovatnoću je [17] [18]:

$$P(A/B)=\frac{(P(A\cap B))}{(P(B))}$$

Ova verovatnoća se zove i *a priori verovatnoća* jer se odnosi na budućnost: koja je

verovatnoća da se desi događaj A u budućnosti ako se već desio događaj B.

Međutim, za ES je bitno moći odgovoriti i na obrnuto postavljen problem: koja je verovatnoća da se već desio neki događaj B pre događaja A ako se događaj A desio. U 18. veku je britanski matematičar Thomas Bayes definisao teoremu koja omogućava izačunavanje i ove tzv. *a posteriori verovatnoće* (odnosi se na prošlost)[17]:

$$P(B/A)=\frac{(P(B)*P(A/B))}{(P(A))}$$

Pemisa pravila A i zaključak pravila B se posmatraju kao događaji. Na osnovu ovakve postavke i Bayes-ove formule može se izračunati verovatnoća da je zaključak B tačan ako je premisa A tačna, pa pravilo izgleda:

IF A **THEN** B (P(B/A))

Teorija izvesnosti (Certainty theory)

Teorija verovatnoće funkcioniše u onim situacijama kada su poznate a priori verovatnoće odigravanja svih događaja. Međutim ova teorija, upravo zbog toga, često ne može da se primeni. Pored ovog problema teorije verovatnoće, postojale su i dodatne poteškoće koje su se odnosile na to da teorija verovatnoće ne modeluje na odgovarajući način čovekovo razmišljanje u situacijama neizvesnosti i nepotpunih informacija [44].

Da bi rešili ove probleme Shortliffe i Buchanan su 1975. godine stvorili teoriju izvesnosti i primenili je u okviru čuvenog MYCIN ES [46]. Ova teorija je stvorena tako da što približnije modeluje način čovekovog razmišljanja u slučajevima nepotpunih i neizvesnih informacija. Osnovni element ove teorije je *faktor izvesnosti* (certainty factor). Faktor izvesnosti (u daljem tekstu CF) predstavlja broj koji može da ima vrednost između -1 i 1 i označava subjektivan nivo izvesnosti (ekspert je taj koji definiše CF) da je neki iskaz tačan. Pri tome, ako je CF jednako -1 iskaz je potpuno netačan, ako je CF jednako 1 iskaz je potpuno tačan, a sve međuvrednosti predstavljaju različite stepene slaganja ili neslaganja. CF koji ima vrednost nula označava da nemogućnost da se utvrdi tačnost iskaza. Pravilo koje koristi CF za označavanje neizvesnog znanja može da izgleda ovako:

IF A **THEN** B (CF = 0.7)

Kada se pravilo protumači, dolazi se do zaključka da ako važi iskaz A, važiće i iskaz B ali samo sa faktorom izvesnosti CF = 0.7 tj. B je vrlo verovatno tačan. Ovaj faktor izvesnosti je faktor izvesnosti pravila (jer je u vezi sa konkretnim pravilom).

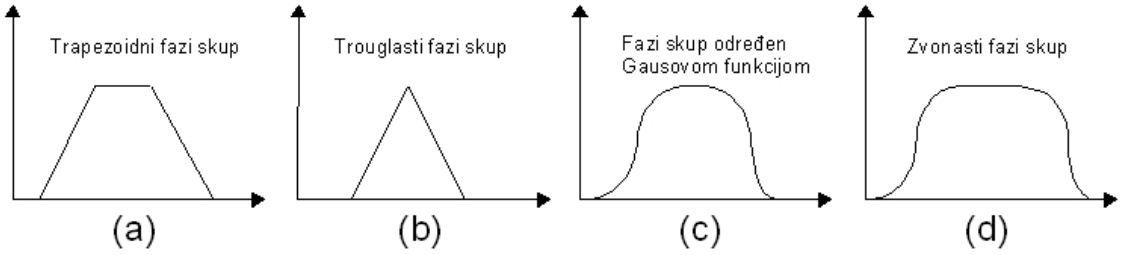
Prethodno opisana situacija podrazumeva da iskaz A važi sa CF = 1 tj. da je potpuno tačan iskaz. Međutim, moguća je i situacija da i premise imaju svoj CF (*faktor izvesnosti premisa*) tj. da nisu potpuno tačne. Još je složenija situacija kod pravila koje imaju više premisa, a svaka ima CF manji od jedan. U svim ovim slučajevima postoje posebne formule za računanje CF zaključaka, ali to zavisi od tehnike za neizvesno zaključivanje.

Fuzzy teorija (Fuzzy theory)

Fuzzy teorija je konačan oblik dobila u radovima matematičara Zadeha [52][53]. Glavni koncept kojim fuzzy teorija upravlja su nejasni i neprecizni lingvistički pojmovi kao npr.: visok, nizak, mršav, debeo itd. Zbog toga, uvodi se pojam *fuzzy skupa* (fuzzy set) u kojem svaka vrednost ima svoj *stepen pripadnosti skupu*. Ovaj stepen pripadnosti može da ima vrednost između nula i jedan pri čemu nula označava da ta vrednost sigurno ne pripada skupu, dok jedinica označava suprotno.

$$\mu_A(X) - stepen\ pripadnosti\ elementa\ X\ skupu\ A\,(uzima\ neku\ vrednost\ iz\ intervala\ [0,1])$$

Stepen pripadnosti i tip fuzzy skupa su određeni funkcijom pripadnosti. Funkcija pripadnosti može biti bilo koja matematička funkcija koja svaku ulaznu vrednost preslikava na interval [0,1]. Izbor odgovarajuće funkcije pripadnosti se vrši uvek u kontekstu sa neodređenim pojmom za koji se kreira skup. Najčešći tipovi fuzzy skupova se mogu videti na slici (Slika 11). Na ordinati je prikazan stepen pripadnosti čija je maksimalna vrednost jedan.



Slika 11: Tipovi fuzzy skupova

U kontekstu ES, fuzzy teorija se svodi na formiranje i upotrebu *fuzzy pravila* (fuzzy rule). Fuzzy pravila nastaju kada se u premise običnih pravila unesu fuzzy pojmovi. Primer fuzzy pravila je:

IF Čovek je visok
THEN Čovek je težak

Potrebno je napomenuti da se teorija verovatnoće i teorija izvesnosti primenjuju u onim situacijama kada su iskazi precizni ali nisu obavezno potpuno tačni, dok se fuzzy teorija primenjuje onda kada su iskazi tačni, ali se ne mogu precizno iskazati.

2.2.3 Metode i tehnike za zaključivanje

Kada čovek razmišlja o nečemu i donese neke zaključke, to se zove *rasuđivanje* (reasoning). ES sa druge strane, mogu samo da simuliraju čovekovo rasuđivanje. Ovaj proces kojim ES dolaze do zaključaka se zove *zaključivanje* (inference). "Rasuđivanje je proces u kojem čovek na osnovu znanja, činjenica i strategija za rešavanje problema izvodi neke zaključke." [44]

Definicija zaključivanja je veoma slična i može se izvesti iz prethodne definicije. "Zaključivanje je proces u kojem ES na osnovu ugrađenog znanja, unetih činjenica i strategija za rešavanje problema korišćenjem odgovarajućeg algoritma izvodi nove zaključke."

Ako se posmatraju ulazi i izlazi procesa rasuđivanja, često se pominju tri vrste rasuđivanja [44][45][46]:

Deduktivno rasuđivanje: uzrok + implikacije => efekat

Abduktivno rasuđivanje: efekat + implikacije => uzrok

Induktivno rasuđivanje: uzrok + efekat => implikacije

U kontekstu ES, najčešća situacija je da ES vrši deduktivno ili abduktivno zaključivanje, dok oblast induktivnog zaključivanja spada u domen mašinskog učenja i ređe se pojavljuje u okviru ES. Sve ove vrste zaključivanja se zasnivaju na matematičkom pravilu zaključivanja modus ponens.

Pored toga, rasuđivanje se može klasifikovati i prema kriterijumu postojanosti činjenica o problemu [44][54]. Ako se činjenice o problemu ne menjaju u toku rasuđivanja u pitanju je *monotono rasuđivanje* (monotonic reasoning). Ako se, sa druge strane, može desiti da se činjenice promene i u toku rasuđivanja, u pitanju je *nemonotono rasuđivanje* (non-monotonic reasoning) [44][54]. U ovom slučaju, algoritam za zaključivanje mora da sadrži i *sistem za održavanje istinitosti* (truth maintenance system). Ovaj sistem automatski ažurira listu činjenica i briše sve zaključake koji su nastali na osnovu činjenica koje više ne važe.

Izbor tehnike za zaključivanje prvenstveno zavisi od korišćene metode za predstavljanje znanja. Tehnike zaključivanja koji se najčešće koriste u okviru ES su date u nastavku.

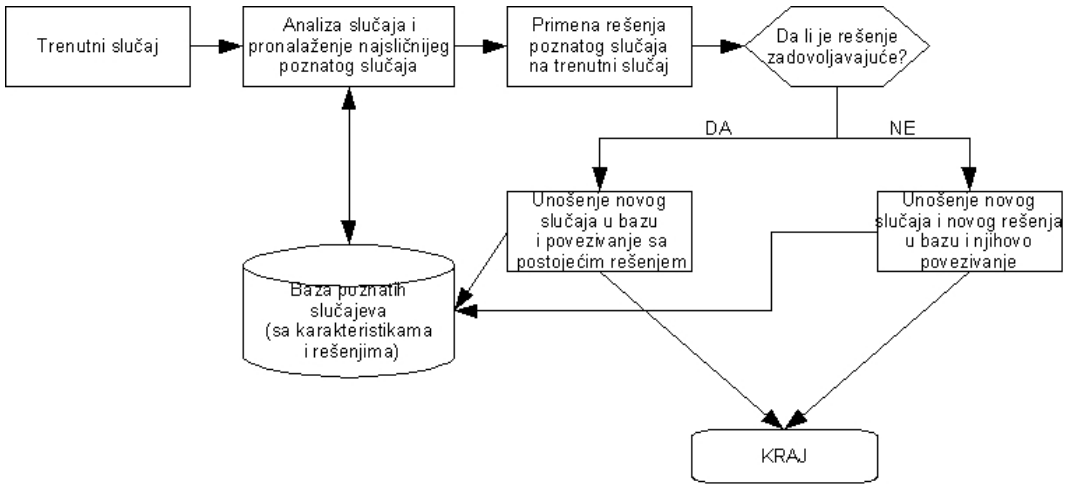
Zaključivanje na osnovu već rešavanih slučajeva (Case-based inferencing)

Zaključivanje na osnovu već rešavanih slučajeva (Case-based inferencing) se zasniva na jednom od principa ljudskog razmišljanja: pronalaženje rešenja nepoznate situacije na osnovu prethodnih iskustava iz sličnih situacija. U kontekstu ove vrste zaključivanja, situacije se nazivaju *slučajevima*. Svaki slučaj se može definisati kao "određeni događaj ili situacija koji ima neke jedinstvene karakteristike" ([46], [55]).

Ideja je da se u ES na neki formalan način unesu svi već rešavani slučajevi sa njihovim karakteristikama i rešenjima, i da se onda, na osnovu procene karakteristika trenutnog slučaja, primeni rešenje najsličnijeg slučaja. Uvodi se i povratna sprega koja proverava da li je dobijeno rešenje zadovoljavajuće. Ako jeste, novi slučaj se unosi u bazu znanja i povezuje se sa postojećim rešenjem. Ako nije, slučaj se opet unosi u bazu, ali se za njega formuliše neko novo rešenje. Ceo proces je prikazan na sledećoj slici (Slika 12).

Metode i tehnike koje se koriste za predstavljanje samih slučajeva i njihovih karakteristika i rešenja su veoma trivijalne. Najčešće su to relacione baze podataka ili odgovarajuće strukture objekata u radnoj memoriji.

Sama metoda za zaključivanje, kao svoj ključni deo, sadrži algoritam za pronalaženje najsličnijeg slučaja iz baze. Poređenje se vrši prema kriterijumima koji mogu biti kvantitativni (temperatura, pritisak itd.), ali i kvalitativni (boja, status nekog objekta itd.). Ako se ne nađe slučaj sa potpuno istim karakteristikama, traže se slučajevi sa najvećim brojem istih karakteristika, i u zavisnosti od važnosti pojedinačnih kriterijuma, rangiraju se prema sličnosti sa trenutnim slučajem.



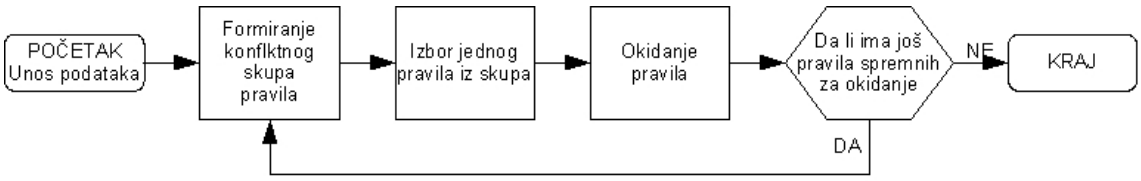
Slika 12: Algoritam zaključivanja na osnovu rešavanih slučajeva

Ulančavanje unapred (Forward chaining)

Ulančavanje unapred je jedna od najčešće korišćenih tehnika za zaključivanjekoja se koristi isključivo u kombinaciji sa pravilima. Ovo je "strategija za zaključivanje koji polazi od skupa poznatih činjenica o problemu, izvodi nove činjenice korišćenjem pravila čije premise odgovaraju postojećim činjenicama i vrši ovaj proces sve dok se ne stigne do neke ciljne činjenice ili dok ne ponestane pravila čije premise odgovaraju početnim ili izvedenim činjenicama" [44].

Prirodna posledica ovog pristupa je ta da je moguće izvesti dosta zaključaka na osnovu malo ulaznih podataka. Zbog toga je drugi naziv za ulančavanje unapred i *podacima*

vođena strategija (data-driven strategy). Prikaz algoritma za ulančavanje unapred je dat na slici (Slika 13).



Slika 13: Algoritam za ulančavanje unapred

Prvi korak je formiranje *konfliktnog skupa pravila* (conflict rule set). To su sva ona pravila čije premise odgovaraju činjenicama, pa se zaključak pravila može izvesti. Znači, u ovom koraku se prolazi kroz skup svih pravila, uočavaju se ona koja mogu da se izvrše i smeštaju se u konfliktni skup pravila. Konflikt nastaje zato što se često dešava da se istovremeno može izvršiti više pravila, pa je pitanje koje bi pravilo trebalo prvo izvršiti. Radi optimizacije, u ovom koraku se često koristi *Rete algoritam* (Rete – mreža) [56].

Drugi korak je *rešavanje konflikta* (conflict resolution) i izbor samo jednog pravila za izvršavanje. Postoji nekoliko mogućih strategija za rešavanje konflikta ([44], [46], [55]):

- Izbor prvog pravila
- Izbor pravila sa najvišim prioritetom
- Izbor najspecifičnijeg pravila
- Izbor pravila koje se odnosi na najskorije dodate činjenice
- Izbor pravila koja nisu izvršena
- Izvršavanje svih pravila iz konfliktnog skupa ali u odvojenim linijama zaključivanja

Treći korak je izvršavanje tj. *okidanje pravila* (rule firing). Kada se pravilo okine, njegovi zaključci se dodaju postojećim činjenicama u radnoj memoriji i koriste se u narednom ciklusu zaključivanja.

Kada se izvrši i treći korak, završava se jedan ciklus zaključivanja. Zaključivanje se najčešće vrši u više ciklusa, a prekida se ako se stigne do neke ciljne činjenice ili više

nema pravila koja se mogu izvršiti.

Ulančavanje unazad (Backward chaining)

Slično ulančavanju unapred i *ulančavanje unazad* (backward chaining) je metoda za zaključivanje koja se može koristiti isključivo u kombinaciji sa pravilima. Međutim, ulančavanje unazad je *ciljno vođena strategija* (goal-driven strategy). Dakle, ne ide se od podataka ka zaključcima, već obrnuto: traže se podaci da bi se dokazali ili opovrgli zaključci. Prema definiciji, ulančavanje unazad je "strategija zaključivanja u kojoj se pokušava dokazati neka teorema sakupljanjem podataka potrebnih za njeno dokazivanje" [44]

Proces ulančavanja unazad počinje zadatim ciljem tj. činjenicom koju je potrebno dokazati. Prvi korak je provera da li se ta činjenica slučajno već ne nalazi u memoriji. Ako se ne nalazi, sistem traži pravila (jedno ili više njih) koja u svom THEN delu sadrže tu činjenicu. Kada ih nađe, sistem proverava da li se premise ovih pravila nalaze u memoriji u vidu činjenica. Ako se nalaze, početni cilj je ostvaren i činjenica je dokazana. Ako se ne nalaze, traži se nov skup pravila koji u svom THEN delu imaju premise pravila iz prethodnog kruga. Ceo proces se vrši dok se ne naiđe na primitive. Primitive su činjenice koje su premise pravila ali se ne nalaze u THEN delu nijednog pravila. Tada korisnik daje podatke o tačnosti primitiva i time dokazuje ili opovrgava početnu činjenicu.

Ciljevi se u okviru ovog algoritma dokazuju hijerarhijski. To znači da se prvo dokazuju ciljevi nižeg nivoa, pa tek onda ciljevi višeg nivoa. Hijerarhijska struktura u kojoj se čuvaju ciljevi se zove agenda ciljeva.

Ulančavanje unazad se koristi u trenucima kada postoji samo par rešenja problema ali treba dokazati koje od njih je tačno. To su situacije u kojima bi pretraživanje celog problemskog prostora ulančavanjem unapred dovelo do eksplozije činjenica i među-zaključaka i prikupljanja nepotrebnih podataka.

Pretraživanje po dubini, po širini i heurističko pretraživanje

Već je navedeno da se skup pravila može predstaviti u vidu grafa – mreže zaključivanja. Zaključivanje na osnovu ovako predstavljenog znanja zahteva posebne tehnike i algoritme. U tu svrhu se može upotrebiti opšta tehnika pretraživanja i njeni algoritmi: *pretraživanje po dubini* (Depth first), *pretraživanje po širini* (Breadth first) i *heurističko pretraživanje* (Best first). Ova tehnika se koristi i van oblasti ES i nije ograničena na mreže zaključivanja, već je primenljiva na bilo kojoj vrsti grafa.

"Pretraživanje po dubini je algoritam pretraživanja koji prvo traži rešenje duž cele vertikale trenutne grane problemskog prostora pre nego što pređe na sledeću granu." [44]

U praksi to znači da se prvo pretraži trenutna grana do svih njenih dubina u cilju pronalaženja rešenja, a ako to nije moguće, prelazi se na sledeću granu itd. Prednost pretraživanja po dubini je ta što se sigurno pronalazi rešenje ako postoji. Mana je ta što se pretražuje ceo prostor po dubini pa se, u početku, često ne mogu uočiti neka jednostavna (tzv. plitka) rešenja. Još jedna mana je to što se pretraživanje vrši nediskriminantno tj. pretražuje se uvek ceo problemski prostor i to istim redom.

Pretraživanje po širini je algoritam koji se takođe može koristiti u kombinaciji sa mrežama zaključivanja. Za razliku od pretraživanja po dubini, pretraživanje po širini prvo proverava najjednostavnija tj. najplića rešenja, pa ako ih ne nađe prelazi na sledeći nivo dubine problemskog prostora.

"Pretraživanje po širini je algoritam pretraživanja u kome se proveravaju svi čvorovi na istom nivou mreže zaključivanja pre nego što se pređe na niži nivo." [44]

Prednosti pretraživanja po širini su ta da se odmah nalaze plitka rešenja ako postoje, i da će rešenje sigurno biti nađeno. Nedostatak je sličan kao i kod pretraživanja po dubini, a to je da se pretražuje uvek ceo problemski prostor i to redom.

Pretraživanje po dubini i pretraživanje po širini uvek prolaze kroz ceo problemski prostor. U većini slučajeva to je u redu, međutim problemi nastaju kada je problemski prostor velik. Tada je potrebno suziti polje pretrage na najverovatnija rešenja. Algoritam koji to omogućava je heurističko pretraživanje.

"Heurističko pretraživanje je algoritam pretraživanja koja koristi znanje o problemu da

bi usmerio pretragu problemskog prostora prema onim rešenjima koja se čine najverovatnijim.” [44]

Heurističko pretraživanje podrazumeva korišćenje jedne ili više tehnika za usmeravanje pretraživanja: uređivanje redosleda grana problemskog prostora, korišćenje meta-pravila, uvođenje prioriteta za grane ili podelu problemskog prostora na manje celine.

Prednosti heurističkog pretraživanja su te da se veoma brzo nalazi rešenje ako je očekivano. U suprotnom, može se desiti da algoritam mnogo duže traži rešenje od prethodna dva. To je ujedno i najveći nedostatak algoritma za heurističko pretraživanje.

Zaključivanje na osnovu okvira (Frame-based inferencing)

Zaključivanje na osnovu okvira (Frame-based inferencinf) tj. konkretna tehnika zaključivanja zavisi od načina ostvarivanja komunikacije između okvira [44]:

- Komunikacija ostvarena korišćenjem pravila
- Komunikacija ostvarena korišćenjem facet-a
- Komunikacija ostvarena korišćenjem metoda

Komunikacija ostvarena korišćenjem pravila se odnosi na hibridne ES. Okviri se koriste kao sredstvo za struktuirano predstavljanje podataka (slotovi) i složenog ponašanja (metode) a pravila su ta koja ih povezuju. Zaključivanje se onda svodi na neku proširenu verziju zaključivanja na osnovu pravila. Pravila se izvršavaju korišćenjem neke od tehnika ulančavanja, ali dozvoljavaju i pozivanje i izvršavanje metoda i facet-a.

Komunikacija ostvarena korišćenjem facet-a je karakteristična isključivo za ES zasnovane na okvirima. Proces zaključivanja se aktivira svaki put kada se desi neki događaj i to po sledećem redosledu:

1. Desi se događaj. U pitanju je promena vrednosti slota ili potreba za pribavljanjem te vrednosti.
2. Aktivira se facet i izvrši zaključivanje (faceti sadrže proceduralno znanje).
3. Ako facet izaziva još neki događaj (promenu vrednosti slota), ide se na korak 1.

Komunikacija ostvarena korišćenjem metoda je takođe karakteristika ES zasnovanih na okvirima. Tehnika zaključivanja se sastoji iz nekoliko koraka i veoma je slična kao kod komunikacije zasnovane na facetima, samo što su događaji koji aktiviraju proces zaključivanja drugačiji. Kod faceta, zaključivanje se aktivira promenom vrednosti slota ili potrebom za pribavljanjem te vrednosti. Kod metoda, događaj je poziv metode.

Veoma je važno obratiti pažnju na to da su oblici komunikacije karakteristični za okvire (poslednja dva oblika) *orijentisani ka događajima* (event driven) i predstavljaju pravi izbor ako ES treba da radi u realnom vremenu.

Zaključivanje na osnovu faktora izvesnosti

Tehnika *zaključivanja na osnovu faktora izvesnosti* se svodi na sledeća tri koraka:

1. Utvrđivanje koje činjenice važe i koji je njihov CF.
2. Uparivanje činjenica sa premisama, izvođenje zaključaka i izračunavanje CF svakog novog zaključka na osnovu CF pravila i CF premisa.
3. Ako ima još pravila koja nisu izvršena a mogu biti, ponavljanje celog procesa.

Kada bi, na primer, iskaz koji predstavlja premisu pravila (neka je to A) važio sa faktorom izvesnosti CF = 0.8, a CF pravila bio jednak 0.7, faktor izvesnosti zaključka (npr. B) bi se računao prema formuli [44]:

$$CF(B)=CF(A)*CF(pravila)=0.8*0.7=0.56$$

Kada pravilo ima više premisa faktor izvesnosti se računa u zavisnosti od toga kojim logičkim operacijama su premise povezane. Najčešće su u pitanju I (AND) i ILI (OR) logičke operacije. Formula za izračunavanje CF u slučaju povezanosti premisa logičkim operatorom I tj ILIglasi [44]:

Logičko I: $CF(B)=min[CF(A1),CF(A2),...,CF(An)]*CF(pravila)$

Logičko ILI $CF(B)=max[CF(A1),CF(A2),...,CF(An)]*CF(pravila)$

Veoma je česta situacija da, u okviru jednog ES, dva pravila sa različitim premisama vode ka istom zaključku. Ako samo važi A ili C, to nije problem jer se aktivira samo jedno pravilo a CF se izračunava prema poznatoj formuli. Problem nastaje kada važe i A

i C jer je potrebno iskombinovati dve vrednosti da bi se dobio CF (B). Neka je faktor izvesnosti iskaza B koji se dobije iz pravila jedan CF1(B) a faktor izvesnosti koji se dobije iz formule dva CF2(B). Formula za izračunavanje ukupnog CF je [44]:

$$CF(B) = CF1(B) + CF2(B) * (1 - CF1(B)), \text{ ako su i } CF1 \text{ i } CF2 \text{ veći od nule}$$

$$CF(B) = \frac{(CF1(B) + CF2(B))}{(1 - min[|CF1(B)|, |CF2(B)|])}, \text{ ako je jedan od } CF \text{ manji od nule}$$

$$CF(B) = CF1(B) + CF2(B) * (1 + CF1(B)), \text{ ako su oba } CF \text{ manja od nule}$$

Zaključivanje na osnovu fuzzy pravila

Fuzzy teorija se koristi u kombinaciji sa pravilima da bi se predstavilo neizvesno ili neprecizno znanje u okviru ES. Tehnika zaključivanja na osnovu fuzzy pravila se svodi na nekoliko koraka [53].

Prvi korak je formiranje *fuzzy asocijativne matrice* (Fuzzy associative matrix – FAM). Svako pravilo predstavlja odnos između dva ili više fuzzy skupova tj. definiše preslikavanje između ova dva skupa. Ovo preslikavanje se formalizuje formiranjem FAM. FAM se izračunava na osnovu vektorskih specifikacija oba skupa po principu MAX-MIN ili MAX proizvoda množenja. FAM se izračunava samo jednom - na početku procesa zaključivanja.

Drugi korak je relativno trivijalan i podrazumeva prikupljanje činjenica. Činjenice mogu biti u fuzzy obliku, ali i ne moraju.

Treći korak je nešto složeniji. Kada se unesu činjenice u formi fuzzy iskaza, ovi iskazi se transformišu u vektorsku formu i množe sa FAM time dobijajući zaključke u vektorskoj formi. Poslednji korak je *defazifikacija* (defuzzyfication) u kojoj se rezultirajući vektor svodi na jednu vrednost. Ta vrednost predstavlja zaključak koji se unosi u skup poznatih činjenica.

Četvrti korak podrazumeva preispitivanje uslova za prekid. Ako uslovi nisu ispunjeni, ceo proces se ponavlja ispočetka, sa tim što se zaključci izvedeni u poslednjem ciklusu unose u listu činjenica.

2.2.4 Metode i tehnike za formiranje objašnjenja

Smatra se da je jedna od najvažnijih karakteristika ES njihova mogućnost da objasne svoj proces zaključivanja [57]. Dokazano je da dobro formirano objašnjenje podiže nivo poverenja korisnika u ES [58] i da, ako se korisniku pruži objašnjenje koje nije prilagođeno za njega može da se postigne gori efekat nego da objašnjenje uopšte nije pruženo [59].

Postavlja se pitanje kakve sve vrste objašnjenja ES može da pruži. Veliki broj autora smatra da su to objašnjenja na pitanja *KAKO* i *ZAŠTO* ([44], [46]). Objašnjenje KAKO bi trebalo da pojasni kako je ES došao do rešenja, korak po korak. Ova vrsta objašnjenja se formira u toku zaključivanja. Da bi se otklonila bilo kakva sumnja od strane korisnika, ES može da pruži objašnjenje o tome ZAŠTO postavlja neko konkretno pitanje.

Neki autori smatraju da ES može da pruži i treći tip objašnjenja. To je odgovor na pitanje *STRATEGIJA* [60]. Kada ES rešava neki problem, često se dešava da je problemski prostor relativno veliki, i da potraga za rešenjem mora da bude usmerena na odgovarajući način. Praktično, to znači da se proveravaju prvo ona rešenja za koje se pretpostavlja da su najverovatnija za dati slučaj. Ovo usmeravanje se često vrši uvođenjem neke strategije u vidu meta-pravila.

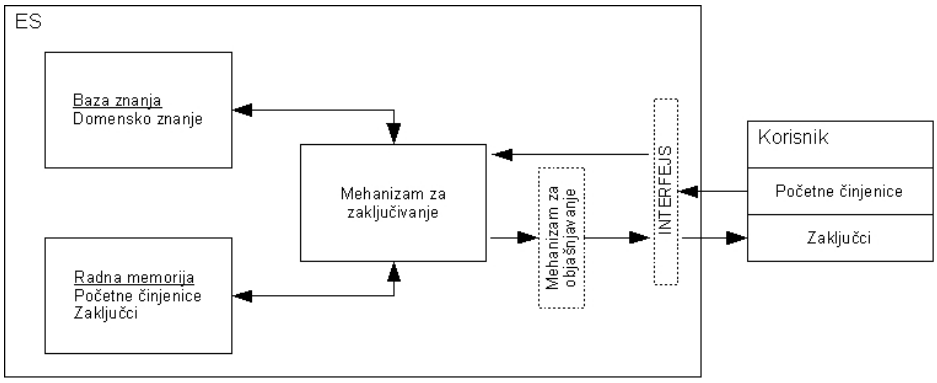
Što se tiče tehnika koje se koriste za formiranje objašnjenja, najčešće su to [61]:

- Trag izvršavanja pravila
- Učaureni tekst
- Domensko znanje

Trag izvršavanja pravila treba da pruži uvid u to koja su se pravila izvršila u toku zaključivanja i kojim redosledom. *Učaureni tekst* (canned text) predstavlja neki statički, već unapred definisan tekst koji objašnjava neki zaključak. Često je moguće i dinamički uneti neki podatak u taj tekst i na taj način dobiti objašnjenje koje liči na govorni jezik, a prilagođeno je unetim činjenicama. Konačno, *domensko znanje* se unosi uz svako pravilo u vidu objašnjenja zašto je baš to pravilo važno za ES [61].

2.2.5 Arhitektura

ES oponašaju proces rasuđivanja, i to se direktno odražava na njihovu arhitekturu. Razlika je ta što ES ne rasuđuju (to mogu samo ljudi) već koriste mehanizme za zaključivanje. Arhitektura ES je data na slici (Slika 14).



Slika 14: Arhitektura ES - slika prema [44]

Baza znanja (knowledge base) ES služi da imitira dugoročnu memoriju eksperta. U njoj se nalazi domensko znanje izraženo korišćenjem neke metode za predstavljanje znanja.

Radna memorija (working memory) ES simulira kratkoročnu memoriju eksperta. Tu se skladište početne činjenice o problemu, i svi zaključci do kojih ES dođe u radu. Izvor početnih činjenica ne mora da bude korisnik. ES može da pristupi nekoj bazi podataka, fajlu ili nekom drugom sistemu i preuzme potrebne podatke.

Mehanizam za zaključivanje (inference engine) uparuje činjenice iz radne memorije sa znanjem iz baze znanja korišćenjem neke metode za zaključivanje i izvodi zaključke. Izvedeni zaključci se skladište u radnoj memoriji.

Svaki ES može da ima i dva dodatna elementa: interfejs i mehanizam za objašnjavanje. *Interfejs* (korisnički) je najčešće grafički i služi za direktnu komunikaciju između ES i korisnika. Preko njega korisnik unosi početne činjenice o problemu i dobija prikaz zaključaka do kojih je ES došao. ES ne mora da ima korisnički interfejs u slučaju da je deo nekog većeg programa. *Mehanizam za objašnjavanje* (explanation facility) služi za pružanje objašnjenja o tome kako je ES došao do zaključaka. Ideja je da to objašnjenje bude što sličnije onome koje pruža živi ekspert, što podrazumeva formiranje rečenica govornog jezika u zavisnosti od zaključaka.

Ekspertni sistemi mogu da budu i distribuirani. Prvu grupu čine ES koji funkcionišu kao i obični ES ali koriste web tehnologije radi omogućavanja istovremenog pristupa većeg broja korisnika. Arhitektura ovih ES je ista kao i za nedistribuirane ES, uz tu razliku što se interfejs kreira korišćenjem web tehnologija.

Drugu grupu distribuiranih ES čine oni čiji je proces zaključivanja distribuiran na više računara. Ideja je da se ceo proces zaključivanja podeli na više manjih, jednostavnijih podprocesa. Najčešći vid implementacije distribuiranih ES ovog tipa je tzv. *blackboard* arhitektura [44][22][23]. U ovom slučaju više ES koristi zajedničku radnu memoriju. Pri tome, svaki ES ima svoju bazu znanja i svoj mehanizam zaključivanja, i može biti smešten na poseban računar.

2.2.6 Trendovi

Kao što je napomenuto, ES su jedna dobro istražena oblast. Posledica toga je da sadašnja istraživanja nisu usmerena ka otkrivanju novih metoda, tehnika i tehnologija ES već ka novim primenama postojećih [62]. Primenu modernih ES i njihovih metoda, tehnika i tehnologija karakterišu tri glavne tendencije:

- Primena u raznim oblastima.
- Primena pod drugim imenom.
- Primena u okviru ili u kombinaciji sa drugim sistemima.

Sve tri navedene tendencije su međusobno povezane, pa će u daljem tekstu biti tako i objašnjene.

ES su se oduvek koristili u raznim oblastima od avio i auto industrije do medicine [44], i to je tendencija koje se nastavlja i danas. Međutim, naziv ekspertni sistem se ne pojavljuje često, već se ovi ES i njihove metode, tehnike i tehnologije koriste pod drugim nazivima, npr.: dijagnostički alati, automatski nadzor, *sistemi za upravljanje poslovnim pravilima* (Business Rule Management System), *sistemi za preporučivanje* (Recommender System) itd. Takođe, ES uglavnom više nisu zasebne aplikacije već se ugrađuju u neku drugu aplikaciju i predstavljaju komponentu koja nije “vidljiva” korisniku. U narednih par pasusa su ukratko prikazani najprisutniji načini primene

metoda, tehnika i tehnologija ES u svetu danas u kojima su ilustrovane ove tri tendencije.

U auto industriji se koriste ES za automatsku dijagnozu kvarova na vozilima. Zahvaljujući već prihvaćenim *OBD* (On-Board Diagnostics), *OBD-II* i *EOBD* standardima [63], svaki noviji automobil poseduje ugrađenu senzorsku mrežu čija svrha je prikupljanje različitih parametara o radu automobila. Kada se desi neki kvar, preko standardnog konektora (DLC - Data Link Connector) se prenose svi kodovi grešaka kao i očitavanja sa senzora. Tada alat za dijagnostiku (koji sadrži ES sa domenskim znanjem) tumači ove kodove i pretvara ih u konkretne informacije o tome u čemu je tačno problem i kako ga je moguće rešiti. Inače, dijagnostički softver može i da prikuplja podatke o radu motora u različitim režimima rada i automatski uočava abnormalnosti [64].

Iako su retki, ES za dijagnozu kvarova na vozilu koji ne rade sa OBD standardima mogu da se kupe za relativno mali iznos. Obično je u pitanju softver (najčešće desktop aplikacija) koji vrši dijagnozu na osnovu simptoma koje unosi korisnik [66]. Čitav proces utvrđivanja problema počinje kada softver postavi korisniku nekoliko jednostavnih pitanja. Kada korisnik odgovori, ES pruža objašnjenje uzroka i načina za rešavanje problema.

OBD-II i EOBD standardi i tehnologije ES su našli primenu i u nadzoru nad korišćenjem vozila [65]. Glavni cilj nadzora je da se identifikuju situacije zloupotrebe automobila (brza vožnja, nedozvoljeno korišćenje itd.). Ceo sistem funcioniše tako što se uređaj za nadzor priključi na već pomenuti standardni konektor za prenos podataka. Ovaj uređaj prikuplja i pamti sve vrste podataka sa senzora automobila koji su u vezi sa lokacijom automobila (uređaj najčešće sadrži GPS uređaj), brzinom, pređenom razdaljinom, vremenom korišćenja itd. Kada auto nije u upotrebi, uređaj može da se skine i priključi na kompjuter (koji poseduje odgovarajući softver) da bi se preneli svi podaci. Softver onda vrši inteligentno filtritanje učitanih podataka na osnovu pravila koje je korisnik uneo i ukazuje na nepravilnosti (npr. neko je vozio auto u 2h ujutru brzinom od 150 km/h).

ES su danas u primeni i u administraciji i bezbednosti kompjuterskih mreža. Ideja je da

se mreža učini što je moguće fleksibilnijom, a da se ne dovede u pitanje bezbednost ili jednostavnost rekonfiguracije. To znači da mreža mora da postane inteligentna da bi mogla da automatski, umesto administratora, završi ove poslove. To je mreža koja bi pružala automatsku i proaktivnu zaštitu od napada i automatsku rekonfiguraciju po potrebi. Cisco je proizveo mrežu koja to radi - *Self Defending Network* [67]. SDN sadrži komponente u koje je ugrađeno ekspertsko znanje: automatski sistem za detekciju napada, agente za bezbednost, modul za automatsku konfiguraciju i puštanje u rad, modul za automatsko upravljanje resursima i automatsku dijagnozu grešaka.

Baza podataka sa automatskim upravljanjem (Self Managing Database [68]) je Oracle rešenje koje koristi ES. Ideja je da se sistem za upravljanje bazom podataka učini dovoljno inteligentnim da može da neke od administratorovih poslova vrši samostalno. Srž sistema je *automatski dijagnostički monitor baze podataka* (Automatic Database Diagnostic Monitor - ADDM). U njega je, u formi stabla odlučivanja, uneto svo ekspertsko znanje iz oblasti administracije baza podataka. ADDM pomaže administratorima u: upravljanju aplikacijama, optimizaciji SQL upita, upravljanju sistemskim resursima, upravljanju rasporedom u memoriji, upravljanju pravljenjem rezervnih kopija i oporavkom baze podataka, upravljanju hardverom.

Sistemi za preporučivanje (recommender systems) koriste tehnologiju ES da bi pružili preporuku o nekom proizvodu, usluzi isl. i objašnjenje o tome zašto je baš to preporučeno [69]. Sama ekspertiza može biti predstavljena nekom od tehnika za kolaborativno preporučivanje (najčešće je to neki DM algoritam npr. nearest neighbor), filtritanje teksta (sadržaja) ali i metodom već rešavanih slučajeva (CBR), dok tehnika objašnjenja zavisi od izabrane tehnike za predstavljanje znanja. Objašnjenje je veoma važna komponenta rezultata ovakvog sistema jer može da podigne nivo poverenja korisnika u sistem a, u slučaju da je preporuka pogrešna, da pomogne korisniku da utvrdi zašto je to tako [70].

ES se koriste i u kombinaciji sa SPI, ali su to najčešće pojedinačni slučajevi i specifični zadaci [40][41][42]. Ovi ES su ukratko opisani u poglavlju o trendovima u oblasti SPI i neće biti ponovo razmatrani ovde.

Međutim, najpopularnija i najrasprostanjenija upotreba tehnologija ES je zapravo pod

drugim imenom - sistemi *poslovnih pravila* (Business Rules) [62]. Poslovna pravila su ograničenja pod kojima poslovni proces mora da funkcioniše. Primer ovakvog ograničenja za rent-a-car agenciju bi bio: "Svaki auto se mora izvesti na test vožnju i proveriti pre iznajmljivanja" ili "Auto se može iznajmiti samo osobama sa važećom vozačkom dozvolom i makar pet godina vozačkog iskustva". "Poslovno pravilo je iskaz kojim se definiše ili ograničava neki aspekt poslovanja. Njegova svrha je uspostavljanje određene strukture ili vršenje uticaja na tok poslovanja" [77].

Poslovna pravila se relativno često menjaju. Promene se dešavaju usled sprovođenja politike preduzeća, izmena zakona, smenjivanja menadžmenta preduzeća i stvaranja drugačijeg pogleda na poslovanje. U svakom slučaju, jasno je da se ova pravila moraju nekako uneti u informacioni sistem preduzeća da bi on mogao da u potpunosti podržava poslovne procese. Poslovna pravila se najčešće iskazuju u formi ako-onda (IF-THEN) iskaza programskih jezika i unose se direktno u izvorni kod aplikacije. U nekim slučajevima ovakav pristup je opravdan, međutim, u većini slučajeva nije jer je krut i ne dozvoljava laku izmenu poslovnih pravila bez menjanja ostatka izvornog koda programa. Zbog toga se tehnike i tehnologije ES koriste za kreiranje, izvršavanje i održavanje skupa pravila.

Posledica toga je da se većina aktuelnih alata za razvoj ES koji su napravljeni u programskom jeziku Java više ne nazivaju ES shell, već *sistemi za upravljanje poslovnim pravilima* (Business Rule Management System - BRMS) ili *mehanizmi za izvršavanje poslovnih pravila* (Business Rule Engine - BRE)[71][72][73][74][75][76]. Razlika između klasičnih ES shell-ova i BRMS tj. BRE je samo u tome što ova druga dva, osim standardnih alata za razvoj, testiranje i pokretanje baze znanja sadrže i druge specijalizovane alate [78][79]:

- Za puštanje velikih baza znanja u rad kao distribuiranih sistema
- Za autorsko potpisivanje pravila
- Za kontrolu verzije pravila
- Za izmenu pokrenute baze znanja u toku rada

Prema jednom izvoru [80] BRMS se mogu podeliti na one koji prvenstveno rade sa

bazom podataka i one koji služe za donošenje zaključaka. Što se tiče integracije poslovnih pravila u poslovne procese, postoje i pokušaji da se to uradi na nivou modelovanja proširenjem BPMN jezika za modelovanje procesa tako da obuhvati i definisanje poslovnih pravila - rBPMN [81].

3 Problem i idejno rešenje

U ovom poglavlju je dat detaljan opis problema savremenih SPI koji se pokušava rešiti, kao i prikaz idejnog rešenja koje se predlaže. Određivanje problema počinje definisanjem pojmova podatka i informacije jer je razumevanje razlike između ova dva pojma ključno za razumevanje problema. Nakon toga, dat je precizan prikaz problema koji se odnosi na mali udeo informacija (i veliki udeo podataka) u izveštajima koji SPI pružaju. Ostatak poglavlja čini opis ideje rešenja, primera izmenjene arhitekture SPI, kao i metoda koje su upotrebljene. Nešto manje detaljan opis idejnog rešenja, njegove arhitekture i potencijalnih pozitivnih i negativnih aspekata se može naći u [86].

3.1 Problem

U cilju definisanja problema istraživanja, potrebno je jasno odrediti pojmove podatka i informacije, pa se tek onda može preći na uzroke koji utiču na pojavljivanje problema i njihovu analizu.

3.1.1 Podaci i informacije

Neformalana definicija podatka je da je to jedna ili više vrednosti predstavljenih u numeričkoj, simboličkoj ili nekoj drugoj formi. U zavisnosti od toga kako je podatak predstavljen i koji se odnosi mogu uspostaviti između dve vrednosti iz iste grupe podataka, u statistici se definiše nekoliko *skala podataka* [17][18]:

- *Nominalna skala* - podaci su predstavljeni simbolički i ne mogu se uspostaviti odnosi između vrednosti osim da li su iste ili različite. Na primer, pol se predstavlja ovom skalom a vrednosti su: muški i ženski.
- *Ordinalna skala* - podaci su numerički i jedino se mogu uspostaviti odnosi veće od ili manje od tj. podaci se mogu sortirati. Primer je rang takmičara - neko je na cilj stigao pre a neko kasnije.
- *Intervalna skala* - podaci su numerički a između njih se mogu postaviti relacije manje od ili veće od ali i razlike između dve vrednosti. Na primer, temperatura

od 20C je za 15 stepeni veća od temperature 5C.

- *Racio skala* - podaci su numerički a između njih se mogu postaviti relacije manje ili veće od, razlike između dve vrednosti, ali i relativnog odnosa između dve vrednosti. Na primer, plata osobe A je 200$ i dva puta je veća od plate osobe B koja iznosi 100$.

Informacija nije isto što i podatak. Neke popularne definicije informacije su:

- "Informacije su podaci koji su pretvoreni u oblik koji ima smisao za primaoca tih informacija" [82]
- "Podaci su onaj sirovi materijal koji se obrađuje i prerađuje u cilju stvaranja informacija." [83]
- "Informacija je podatak kojem je pridruženo značenje." [84]

Iako ne postoji jedinstvena definicija informacije, u naučnoj i stručnoj javnosti je generalno prihvaćen sledeći odnos između podatka i informacije [85]:

PODATAK + ZNAČENJE = INFORMACIJA

Primeri podataka su: plava, 32$, 17C itd. Podaci, ako se gledaju izolovano, imaju vrlo malo značenja. Da bi dobili značenje, moraju prvo da se grupišu po kontekstu, a onda i analiziraju. Na primer, ako neko želi da stekne uvid u vremenske prilike, saznanje da je trenutno 17C ne znači mnogo. Međutim, ako se uz to predstave i (kontekstualno povezani) podaci da je trenutno decembar, i da se temperatura odnosi na Moskvu, postoji osnova da se zaključi da je vreme neuobičajeno toplo za tu lokaciju i doba godine. Ovaj zaključak predstavlja informaciju i, između ostalog, doprinosi znanju primaoca. Ova informacija omogućava momentalno delovanje i donošenje odluka jer nije potrebna nikakva dodatna analiza - na osnovu nje primalac može da reaguje i da se obuče na odgovarajući način. Konačno, može se zaključiti da su informacije mnogo korisnije od podataka.

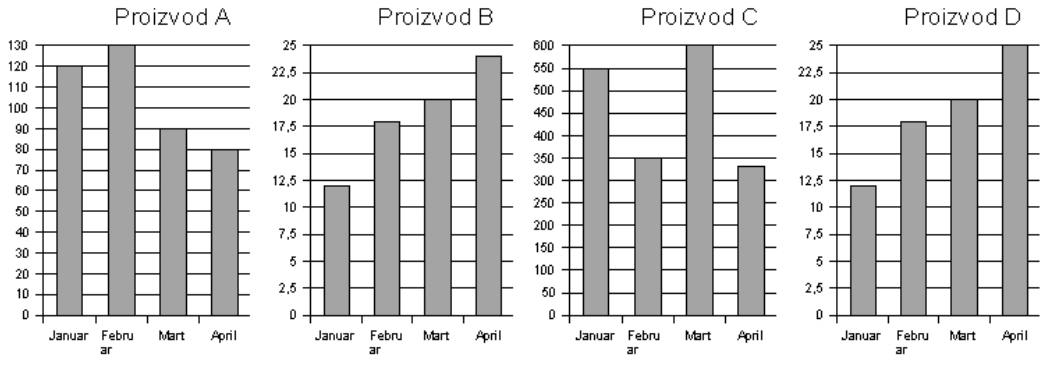
3.1.2 Nedostaci procesa tumačenja izveštaja SPI

Kakve to ima veze sa SPI i izveštajima uopšte? Već je navedeno da postoje dve vrste

korisnika SPI: *poslovni analitičari* i *menadžeri*. Poslovni analitičari imaju za zadatak da uoče neke nove zakonitosti među podacima. Ovaj posao zahteva posedovanje i korišćenje velike količine poslovnog ali i tehničkog znanja jer su metode koje se koriste složene. To je unekoliko opravdano zbog toga što su zakonitosti koje se žele otkriti često dobro sakrivene, pa analitičar mora da poseduje i veliko iskustvo i intuiciju da bi uopšte znao gde i šta da traži. Drugim rečima, normalno je i očekivano da poslovni analitičari rade sa velikim količinama sirovih i prerađenih podataka jer traže neke zakonitosti (informacije) koje nisu do tada postojale.

Menadžeri, sa druge strane, imaju zadatak da prate poslovne pokazatelje i da, na osnovu njihovih vrednosti, uočavaju neke već poznate zakonitosti. Jednostavnije rečeno, njihov cilj je da, na osnovu pregledanja vrednosti više unapred poznatih pokazatelja, utvrde da li firma posluje dobro ili loše i koji je uzrok tome. Da bi uspešno izvršili ovaj zadatak, menadžeri moraju da poseduju veću količinu poslovnog znanja. Međutim, savremeni SPI i njihove metode uslovljavaju i veoma dobro poznavanje rada sa konkretnim SPI da bi se izvukli optimalni rezultati. Pored toga, iako su zakonitosti koje se traže i način na koji se do njih dolazi već unapred poznati (profit je opao, prodaja je veća u prvom kvartalu itd.), SPI metode ne mogu automatski da ih uoče, već to radi menadžer tako što ručno pregleda podatke. Da bi se objasnile ove dve tvrdnje koristiće se sledeći primer.

	Obim prodaje	
Proizvod	April 2007.(kom)	April 2006.(kom)
Proizvod A	80	150
Proizvod B	24	10
Proizvod C	330	500
Proizvod D	25	26



Slika 15: Primer izveštaja SPI

Neka je potrebno da menadžer prodaje analizira podatke prodaje četiri proizvoda za

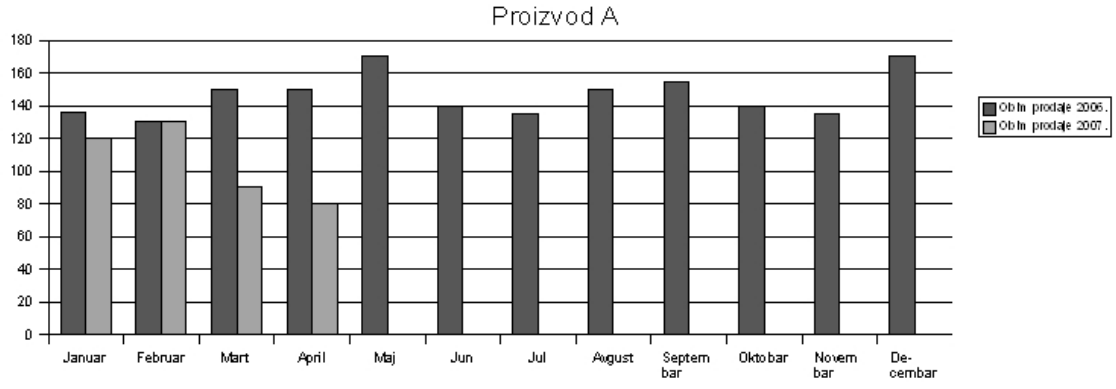
koje je zadužen. Menadžer dobija na ekranu neki već predefinisani izveštaj u formi jedne tabele i četiri grafika (Slika 15).

Kao što se može videti, ovaj izveštaj prikazuje podatke o obimu prodaje četiri proizvoda. Tabela sadrži numeričke vrednosti mesečnog obima prodaje svakog proizvoda i vrednosti za isti period prošle godine. Grafici prikazuju obim prodaje za sve mesece tekuće godine.

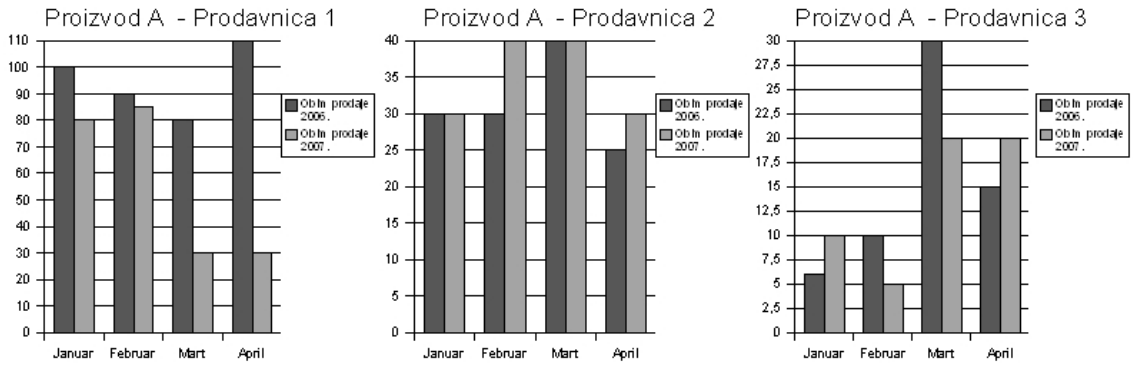
Menadžer sada treba da uoči da li prodaja proizvoda teče kako treba. Jednim pogledom na tabelu, menadžer vidi da proizvod A ima mnogo nižu prodajnu stopu u odnosu na isti period prošle godine. Kada vidi i grafik prodaje proizvoda A, menadžer shvata da prodaja već konstantno opada u martu i aprilu. Na osnovu svega, on zaključuje da prodaja proizvoda A ne ide baš najbolje.

Da bi potvrdio svoje sumnje, menadžer postavlja OLAP upit da mu na jednom grafiku predstavi obim prodaje proizvoda A po mesecima prošle i tekuće godine. Izveštaj koji dobija je dat u formi vremenske serije (Slika 16).

Sumnje su sada potvrđene. Menadžer na osnovu vremenske serije zaključuje da je prodaja proizvoda A u ovoj godini značajno opala. Sledeći korak je utvrđivanje uzroka. On tada postavlja OLAP upit koji bi trebalo da mu vrati podatke o prodaji proizvoda A po prodavnicama i to u vidu vremenskih serija za januar, februar, mart i april tekuće i prošle godine. Rezultat upita je dat u formi četiri vremenske serije (Slika 17).



Slika 16: Primer izveštaja SPI



Slika 17: Primer izveštaja SPI

Poslednji izveštaj ukazuje na uzrok problema. Menadžer uočava da je ukupan pad obima prodaje proizvoda A uzrokovan padom obima prodaje u prodavnici 1 u martu i aprilu. Ostale dve prodavnice su funkcionisale podjednako dobro kao i prošle godine.

Da bi proverio i ostale proizvode, menadžer prodaje mora da ponovi ovu proceduru još tri puta. Iako je u pitanju jednostavan primer vidi se da je u pitanju relativno složen proces jer menadžer mora ručno da pregleda veliki broj pokazatelja bilo u grafičkoj, bilo u tabelarnoj formi.

Kada bi koristio balansirane kartice rezultata i dashboard metode, menadžeru bi bilo unekoliko lakše jer bi mogao brže da uoči neke od problema i abnormalnosti. U slučaju kartica rezultata, izašlo bi automatski generisano upozorenje da obim prodaje proizvoda A nije u okviru ciljno zadatih vrednosti. Dashboard bi prikazao obim prodaje proizvoda A u vidu skale sa kazaljkom u crvenoj zoni – pa bi i tada bilo lako uočljivo da je problem u obimu prodaje proizvoda A. Međutim, proces potvrđivanja problema i otkrivanja uzroka bi ostao identičan i oslanjao bi se na korišćenje OLAP upita i ručno pregledanje podataka.

Nedostaci postojećih SPI sa aspekta menadžera se mogu formulisati ovako:

- ***Menadžer mora ručno da pregleda i analizira veliki broj izveštaja da bi uočio i potvrdio makar i osnovne zakonitosti.***
- ***Menadžeru je potrebno dosta tehničkog znanja da bi iskoristio SPI na pravi način.***

Posledice su očigledne. Oni menadžeri koji ne vole da rade sa računarima ili jednostavno ne znaju rad sa SPI nikad neće moći da u potpunosti iskoriste njegove potencijale. Ali, čak i tehnički kvalifikovani menadžeri mogu da imaju probleme. Ako je menadžer zadužen za veliki broj proizvoda, radnih jedinica isl. broj izveštaja koji on mora da pregleda da bi utvrdio kako ide poslovanje je ogroman. Pored velikog utroška vremena za pregledanje izveštaja, pojavljuje se i još jedna nepogodnost. Dovoljno je da menadžeru, iz bilo kog razloga, opadne koncentracija u nekom trenutku pa da propusti neki važan detalj iz izveštaja. Naravno, ako mu uz to nedostaje i poslovno znanje i iskustvo, njegovo tumačenje izveštaja može da bude nepotpuno, nekvalitetno a ponekad i potpuno pogrešno.

3.1.3 Uzroci nedostataka procesa tumačenja izveštaja SPI

Ali, koji je uzrok tome? Zašto menadžer mora ručno da formira upite i pregleda izveštaje da bi stekao sliku o poslovanju preduzeća? Moderni SPI pružaju izveštaje koji su u formi grafika, tabela, obrtomera ili skala. Čak iako je izveštaj rezultat OLAP upita, dobija se u istoj formi.

Izveštaji modernih SPI sadrže uglavnom podatke, a veoma retko kada i informacije.

Grafici i tabele, kako god složeni bili, sadrže samo podatke. U većini slučajeva su to sumirani podaci predstavljeni u odgovarajućoj formi (pogodnoj za tumačenje). Obrtomeri i skale pružaju i neke osnovne informacije utoliko što se automatski proveravaju granične vrednosti, ali je vrednost tih informacija prilično ograničena jer se ne mogu otkriti sve pojave niti njihovi uzroci.

Ova hipoteza se može dokazati na sledeći način. Grafici, tabele, obrtomeri i skale pružaju neke podatke o poslovnim pokazateljima. Primer za to je grafik na kome se prikazuje kretanje profita u toku tekuće godine. Iako je sam po sebi važan, ovaj grafik ne sadrži informacije. Potrebno je da menadžer postavi podatke iz ovog grafika u kontekst sa npr. kretanjem profita za prošlu godinu i kretanjem profita po pojedinačnim proizvodima da bi se stekao informaciju o profitu. Tek onda postoji informacija na osnovu koje se može formirati slika o poslovanju i reagovati na odgovarajući način. Time se dolazi do druge važne tvrdnje:

Korisnik (npr. menadžer) je taj koji podatke iz izveštaja SPI postavlja u kontekst i pravi od njih informacije. To važi i za slučaj kada se identifikuju pojave i kada se analiziraju i otkrivanju njihovi uzroci.

Upravo to je uzrok identifikovanih problema. Kada se koraci identifikacije pojava na osnovu izveštaja i analize uzroka ne bi radili "ručno" od strane menadžera, već automatski (od strane SPI), ne bi postojala potreba za detaljnim tehničkim znanjem da bi se iskoristili potencijali SPI. Takođe, vreme potrebno za uočavanje pojava i njihovih uzroka bi se značajno smanjilo, a kvalitet informacija bi porastao jer ne bi zavisio od koncentracije, znanja, iskustva i objektivnosti one osobe koja tumači izveštaje i, na osnovu njih, formira informacije.

3.2 Idejno rešenje

Postavlja se pitanje, kako je moguće rešiti ovaj nedostatak informacija u izveštajima SPI? Čini se da je idealno rešenje da se ovaj proces transformacije podataka u informacije automatizuje. U tom slučaju, SPI može da automatski stvori izveštaje koji sadrže i podatke i informacije i koji su mnogo korisniji. To znači da se informacije izvode bez ljudskog učešća i da su dostupne bilo gde i bilo kada uz minimum napora. Sa druge strane, menadžeri postaju rasterećeni analize velikih količina podataka, a kvalitet informacija ne varira i ne zavisi od ljudskog faktora.

Proces pretvaranja podataka u informacije se, uopšteno, može podeliti u tri koraka:

- Prikupljanje podataka
- Grupisanje kontekstualno povezanih podataka
- Stvaranje informacija

Prikupljanje podataka počinje kada menadžer želi da pogleda neki izveštaj. On ili ona zadaje odgovarajuću komandu SPI i dobija pojedinačni izveštaj koji sadrži željenje podatke - npr. "ovogodišnji profit je 100.000$".

Grupisanje kontekstualno povezanih podataka počinje kad menadžer shvati da mu je potrebno da pogleda i neke druge, povezane izveštaje i da analizira sve kontekstualno

povezane podatke. On pravi još upita i pregleda rezultate u vidu novih grafika, tabela npr. “ovogodišnji profit je 100.000$, a prošlogodišnji profit je 80.000$”. Potrebno je napomenuti da su, iako grupisani, ovo i dalje samo podaci.

Stvaranje informacija tj. analiza svih podataka i njihova transformacija u informacije je poslednji korak. Ova transformacija se odvija tako što menadžer zapravo dodaje značenje podacima i stvara informacije. On ili ona to postiže korišćenjem svog znanja, npr.: “kada je ovogodišnji profit veći od prošlogodišnjeg, preduzeće posluje dobro”. Konačno, informacija koja se dobija ukrštanjem podataka i znanja izgleda ovako “Sa obzirom na to da je prošlogodišnji profit 80.000$ a ovogodišnji 100.000$, preduzeće posluje dobro”. Navedeni iskaz čini informaciju jer se odmah na osnovu nje može odlučivati i delovati, a direktno doprinosi znanju primaoca. Ceo ovaj proces transformacije se može predstaviti i jednačinom:

PODATAK + ZNANJE POTREBNO ZA TUMAČENJE = INFORMACIJA

Pošto je već utvrđeno da važi:

PODATAK + ZNAČENJE = INFORMACIJA

Iz svega se može zaključiti da je

ZNAČENJE = ZNANJE POTREBNO ZA TUMAČENJE

Prema tome, u cilju automatizacije procesa pretvaranja podataka u informacije, potrebno je znanje potrebno za tumačenje ovih podataka formalizovati i implementirati tako da računar može da ga upotrebi za samostalno zaključivanje. A sve ovo je zapravo definicija ES.

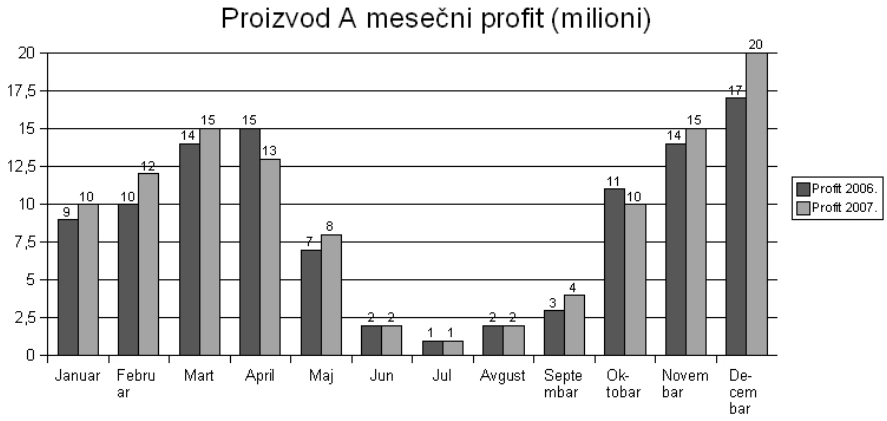
To znači da ES u kombinaciji sa SPI preuzima podatke, tumači ih, i stvara informacije. Izveštaji koje takav sistem pruža su puni informacija, ali sadrže i neke podatke. Sve ovo se može ilustrovati sledećim primerom.

Na primer, neka generalni menadžer neke firme želi da vidi godišnji izveštaj o profitu za proizvod A. Podaci koji su potrebni za izveštaj obuhvataju kretanje profita proizvoda A u toku godine (po mesecima, prosečno i ukupno), kao i vrednosti profita za prošlu godinu. Ovi podaci su dati u Tabeli 1.

Profit proizvoda A za 2007. i 2006. godinu (milioni) *Tabela 1. Podaci o profitu*														
	Jan	Feb	Mar	Apr	Maj	Jun	Jul	Avg	Sep	Okt	Nov	Dec	Uk.	Pros.
Prof. 2007	10	12	15	13	8	2	1	2	4	10	15	20	112	9,33
Prof. 2006	9	10	14	15	7	2	1	2	3	11	14	17	105	8,75

Kada bi menadžer koristio običan SPI, izveštaj koji bi dobio bi bio u formi jednog grafika i jedne tabele (Slika 18). Tada bi bilo na menadžeru da analizira podatke i uoči kako se kreće profit proizvoda A.

	PROFIT	
Proizvod A	2007	2006
Ukupan	112	105
Prosečan	9,33	8,75



Slika 18: Primer izveštaja SPI

Ako bi se, sa druge strane, koristio ES da proumači podatke to bi izgledalo ovako. Neka ES sadrži znanje potrebno za tumačenje profita u formi pravila (nije prikazano kompletno znanje za tumačenje mesečnog profita već samo ono koje pokriva situaciju kada ukupni profit raste):

Grupa pravila za tumačenje ukupnog profita

Pravilo 1

IF Ovogodišnji profit > Prošlogodišnji profit
THEN Profit je porastao (dodati objašnjenje u izveštaj)

Pravilo 2

IF Ovogodišnji profit < Prošlogodišnji profit
THEN Profit je opao (dodati obj. u izveštaj)

Grupa pravila za tumačenje mesečnog profita

Pravilo 4

IF Profit je porastao **AND** Svi ovogodišnji mesečni profiti su veći od prošlogodišnjih
THEN Profit proizvoda A se kreće izuzetno dobro iz meseca u mesec (dodati objašnjenje u izveštaj)

Pravilo 5

IF Profit je porastao **AND** Većina ovogodišnjih mesečnih profita je veća od prošlogodišnjih
THEN Profit proizvoda A se kreće dobro iz meseca u mesec (dodati objašnjenje u izveštaj)

Pravilo 6

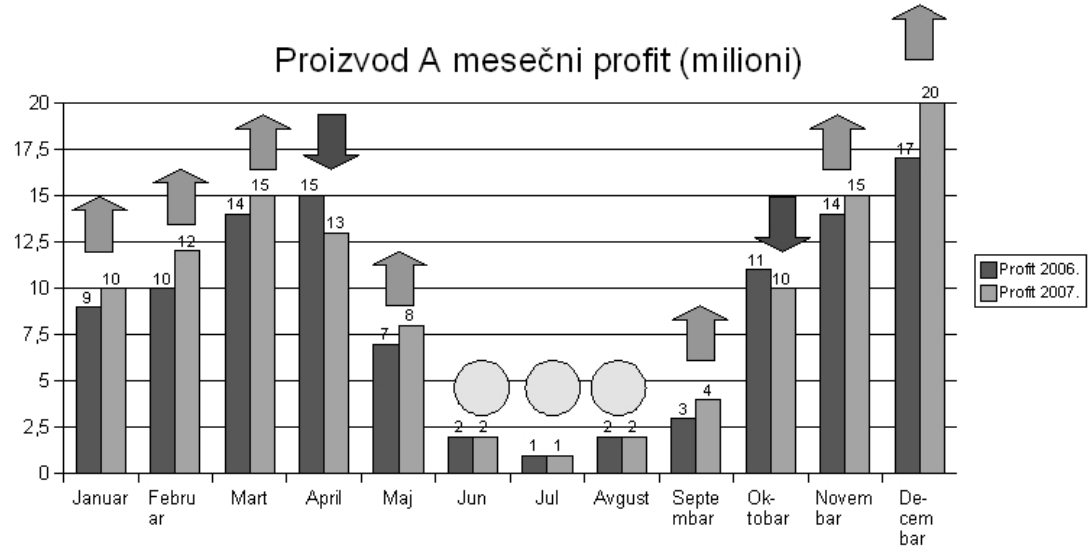
IF Profit je porastao **AND NOT** (Većina ovogodišnjih mesečnih profita je veća od prošlogodišnjih)
THEN Profit proizvoda A raste skokovito (dodati obj. u izveštaj)

Kada ES preuzme podatke o profitu proizvoda A (podaci su dati u vidu tabele) on počinje da te podatke tumači tako što ih uparuje sa pravilima iz baze znanja. ES prvo proverava da li je ovogodišnji profit veći, manji ili jednak prošlogodišnjem. Nakon što

se utvrdi da je ukupni profit porastao, okida se pravilo 1. Njegovo okidanje pokreće dva događaja: donošenje zaključka da je profit porastao i dodavanje objašnjenja o tome u izveštaj. U sledećem koraku zaključivanja, ES proverava kretanje mesčnog profita. Pošto je ovogodišnji mesečni profit veći od prošlogodišnjih za 7 meseci od 12 (januar, februar, mart, maj, septembar, novembar, decembar) okida se pravilo broj 5 jer njegova premisa zadovoljena: ukupan profit jeste porastao i većina ovogodišnjih mesečnih profita je veća od prošlogodišnjih. Okidanje pravila 5 dovodi do donošenja zaključka da je kretanje profita proizvoda A u ovoj godini dobro. Objašnjenje koje prati ovaj zaključak se dodaje u izveštaj.

Na kraju, izveštaj koji dobija menadžer izgleda ovako (Slika 19). Može se primetiti da je ovaj izveštaj pun informacija jer sadrži i zaključke i objašnjenja koja nije potrebno dalje analizirati i tumačiti da bi bili korisni. Menadžeru samo ostaje da pročita izveštaj i odmah može da donosi odgovarajuće odluke i preduzima akcije.

	PROFIT	
Proizvod A	**2007**	**2006**
Ukupan	112	105
Prosečan	9,33	8,75



Zaključak:
Profit proizvoda A ima stabilnu tendenciju rasta.

Objašnjenje:
Ukupan profit proizvoda A je **u porastu u odnosu na prošlu godinu**. Prošle godine je iznosio 105 miliona a ove 112 miliona dinara. To je **porast od 6.7%.**
Pregledanjem mesečnog kretanja profita, a u svetlu sa činjenicom da jeukupni profit porastao, može se zaključiti **relativno ravnomeran rast profita po svim mesecima** u odnosu na prošlu godinu. Mesečni profit za januar, februar, mart, maj, septembar, novembar i decembar je veći ove godine nego prošle godine, dok je za ostale mesece isti ili nešto malo manji.

Preporuka:
Obratiti pažnju i **preduzeti odgovarajuće mere za povećanje profita za jun, jul i avgust a pogotovu za april i oktobar** (jer je za ova dva meseca zabležen pad profita).

Slika 19: Primer izveštaja sa informacijama koje formira ES (NAPOMENA: informacije su izražene u formi rečenica kontrolisanog jezika)

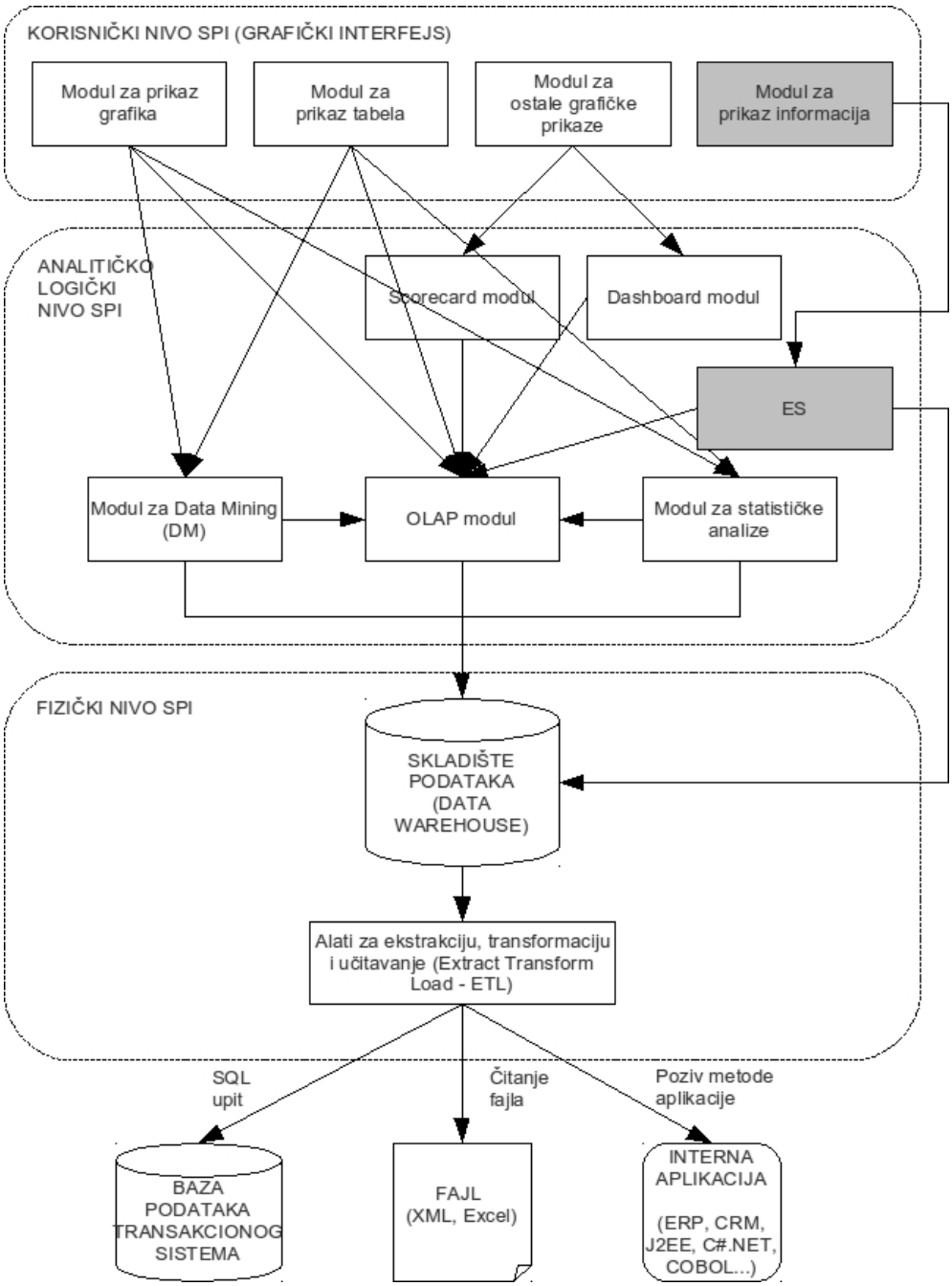
3.3 Logička analiza i projektovanje rešenja

U cilju što jasnijeg i potpunijeg prikaza logičke strukture rešenja, rezultati logičke analize i projektovanja su predstavljeni u dve celine: logička arhitektura rešenja i korišćene metode i tehnike.

3.3.1 Logička arhitektura

U principu, relativno je jasno da je predloženo rešenje zapravo dodatak postojećoj arhitekturi SPI. Razlog je taj što se mnogi već postojeći moduli SPI ionako koriste pa

nije potrebno praviti zasebnu aplikaciju. Izmenjena arhitektura SPI je prikazana na sledećoj slici (Slika 20).



Slika 20: Izmenjena SPI arhitektura i integracija rešenja u SPI

Kao što se može primetiti, najveći deo SPI arhitekture ostaje neizmenjen. ETL i DW alati su i dalje potrebni jer je neophodno izvršiti ekstrakciju, transformaciju i učitavanje podataka, kao i njihovo skladištenje u formi pogodnoj za izveštavanje. Statistički

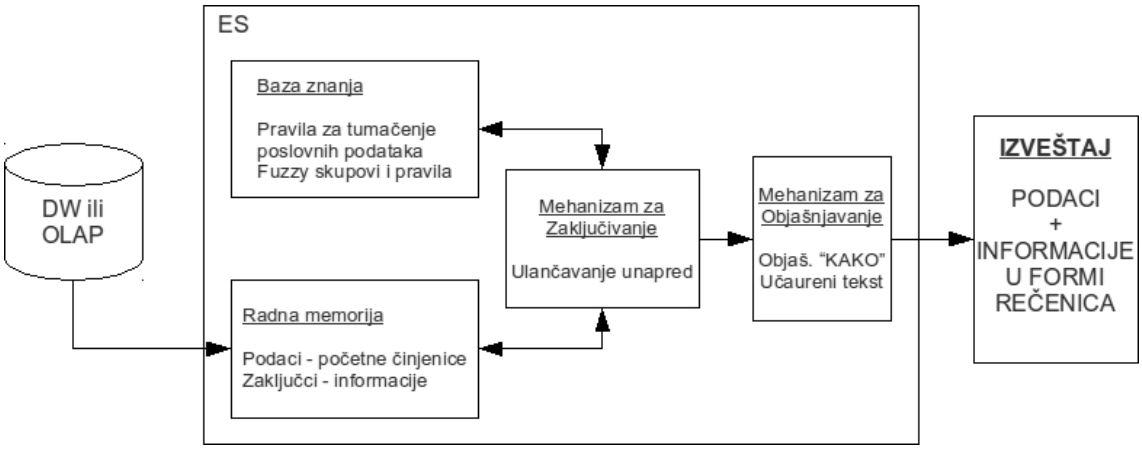
modul, DM modul i OLAP su takođe neophodni jer poslovni analitičari koriste ove module da bi izvršili analizu. Već je napomenuto da se predloženo rešenje odnosi na deo SPI koji koriste menadžeri.

Izmene SPI se prvenstveno odnose na ugradnju ES u SPI. Ovo je neophodno jer se tek na ovaj način mogu izvršiti pravila za tumačenje KPI. Svaki ES se sastoji iz tri komponente: baze znanja, radne memorije i mehanizma za zaključivanje. Baza znanja, u ovom slučaju, sadrži znanje potrebno za tumačenje KPI. Radna memorija, kao i inače, služi za skladištenje početnih činjenica i zaključaka. Početne činjenice su, zapravo, podaci (vrednosti KPI) preuzeti iz DW ili OLAP, a zaključci su informacije koje se žele napraviti. Mehanizam za zaključivanje uparuje znanje iz baze znanja i početne činjenice iz radne memorije (podaci) u cilju stvaranja zaključaka (informacija). Kao dodatak, ES sadrži i mehanizam za objašnjavanje koji pruža objašnjenja na kontrolisanom jeziku tj. predstavlja zaključke (informacije) u formi pogodnoj za menadžere.

Konačno, SPI i dalje sadrži scorecard i dashboard module ali se u prezentacionom nivou nalazi i modul za prikaz informacija tj. tekstualnih objašnjenja. Na ovaj način, pri pravljenju izveštaja, aktiviraju se svi potrebni moduli prezentacionog nivoa pa se, uz podatke, u izveštaju nalaze i informacije koje predstavljaju tumačenje ovih podataka. Grafici, tabele, BKR i komandne table i dalje zauzimaju značajan deo izveštaja i služe da bi menadžer mogao da proveri informacije u odnosu na podatke i vidi da li je zaključivanje ES bilo tačno.

3.3.2 Metode i tehnike

Detaljna arhitektura samog rešenja i metode koje se koriste su prikazane na sledećoj slici (Slika 21). Ova arhitektura, kao što je napomenuto, zapravo predstavlja arhitekturu tipičnog ES.



Slika 21: Arhitektura rešenja

Predloženo rešenje za automatizovano tumačenje vrednosti KPI se sastoji iz primene sledećih metoda i tehnika:

- Korišćenje pravila, objekata i fuzzy logike za formalizovano predstavljanje znanja potrebnog za tumačenje vrednosti KPI.
- Korišćenje ulančavanja unapred i fuzzy zaključivanja u cilju automatizovanog donošenja zaključaka.
- Korišćenje mehanizma za objašnjavanje i tehnika učaurenog teksta da bi se informacije prikazale u formi nalik kontrolisanom jeziku a sa ciljem unošenja u izveštaj i lakog razumevanja.

Cilj je da se ***unapredi kvalitet izveštaja SPI povećavanjem količine informacija u njima***. Objašnjenje date arhitekture i izbora metoda je dato u nastavku.

Već je navedeno da se, u cilju izveštavanja, poslovni podaci moraju ekstrakovati, transformisati, objediniti i učiniti dostupnim u formi DW ili OLAP. Drugim rečima, dovoljno je napraviti odgovarajući upit odgovarajućem skladištu podataka da bi se dobile početni podaci - vrednosti KPI.

Preuzeti podaci čine početne činjenice tj. osnovu za zaključivanje. Prema tome, oni se skladište u radnoj memoriji ES. Zaključci (informacije) se takođe skladište u radnoj memoriji i, zajedno sa podacima, čine osnovu za izvođenje novih zaključaka. Tehnike za predstavljanje činjenica, u ovom slučaju, nisu presudne i mogu biti objekti (klase) ili

pojedinačne promenljive.

Baza znanja sadrži znanje predstavljeno uz pomoć fuzzy skupova, fuzzy pravila i običnih pravila. Ove metode i tehnike se koriste zbog sledećeg.

U razgovoru sa domenskim ekspertima za tumačenje KPI (menadžerima), uočeno je da oni, u toku tumačenja podataka, koriste nejasne (neodređene) pojmove kao: "neutralan" profit, "dobar" profit, "odličan" profit, "novo" preduzeće, "relativno novo" preduzeće itd. Ovi pojmovi se mogu formalizovati korišćenjem fuzzy skupova. Potrebno je naglasiti da svako preduzeće ima svoj skup referentnih vrednosti, pa npr. profit od četiri posto predstavlja "odličan" profit za jedno preduzeće, a istovremeno "osrednji" profit za neko drugo. Ove vrednosti predstavljaju ciljne vrednosti za KPI i menjaju se vremenom i pod uticajem raznih internih i eksternih faktora. Na primer, u doba globalne ekonomske krize, smatra se i da je profit koji je blizu nule "odličan". Čak se i referentne vrednosti za dve različite grane privrede međusobno razlikuju. Prema tome, pored predstavljanja ovih pojmova putem fuzzy skupova, potrebno je njihove referentne vrednosti odvojiti od ostatka baze znanja. Na ovaj način se stvara određena barijera, pa se promena referentnih vrednosti ne propagira na ostatak baze znanja. Kada se podaci pretvore u fuzzy oblik, zaključivanje može da počne.

Znanje za tumačenje KPI se može izraziti korišćenjem fuzzy pravila. Ako je, na primer, potrebno iskazati da se "veoma veliki" profit smatra za "dobru" stvar ako je preduzeće "zrelo", to se može uraditi putem sledećeg pravila.

IF preduzeće je "zrelo"
AND godišnji profit je "veoma veliki"
THEN status profita je "dobar"

Uočeno je da se jedan deo ovog znanja može predstaviti i običnim pravilima.

IF prosečni profit u grani privrede < 0
THEN cela grana privrede je u krizi

Međutim, u najvećem broju situacija, potrebna je kombinacija ove dve tehnike da bi se

znanje izrazilo na formalan način. Sledeći primer ilustruje ovu tvrdnju.

IF (preduzeće je “novo” **OR** preduzeće je “relativno zrelo”)
AND godišnji profit je “negativan”
AND period povraćaja investicije nije istekao
AND profit je u nekom trenutku bio veći od nule
THEN status profita je “upozorenje”

Prva dva dela premise prethodnog pravila koja se tiču starosti preduzeća i kategorizacije godišnjeg profita sadrže fuzzy pojmove. Sa druge strane, treći i četvrti deo premise predstavljaju obične činjenice tj. period povraćaja investicije ili jeste ili nije gotov (nema nikakve neodređenosti). Zaključci prethodno navedenih pravila takođe mogu biti fuzzy (status profita je “dobar”) ili obične činjenice (cela grana industrije je u krizi).

Potrebno je istaći i to da se korišćenjem složenih premisa, pravilima mogu izraziti veoma složene relacije između više pokazatelja odjednom. Na primer, prethodno pravilo uzima u obzir datum osnivanja preduzeća, projektovani period povraćaja investicije, trenutni profit i prošle vrednosti profita. Ako bi menadžer zaključivao na osnovu istih podataka, bilo bi potrebno da naizmenično gleda nekoliko izveštaja (vremenske serije za profit u apsolutnom i relativnom iznosu, podaci o povraćaju investicije, podaci o osnivanju preduzeća i ciljne vrednosti za profit).

Izbor tehnike za zaključivanje zavisi od izabrane tehnike za predstavljanje znanja. U ovom slučaju, kombinacija fuzzy rezonovanja i ulančavanja unapred je odgovarajući izbor. Fuzzy rezonovanje je neophodno radi izvođenja fuzzy zaključaka na osnovu fuzzy pravila, a ulančavanje unapred radi izvođenja zaključaka iz običnih pravila. Ulančavanje unapred je “vođeno podacima” i, kao takvo, doprinosi izvođenju velikog broja zaključaka na osnovu ograničenog skupa početnih činjenica. Zbog toga se smatra da je ovaj algoritam zaključivanja primereniji u odnosu na ulančavanje unazad koje je “vođeno ciljem”. Drugim rečima, nije potrebno dokazati određeni cilj ili hipotezu, već treba “izvući” što više informacija iz unetih podataka. Što se tiče strategije za rešavanje konflikta, možda je najbolje obezbediti da se svako pravilo može izvršiti samo jednom u toku sesije i dodeliti prioritet pravilima.

Zaključci (i fuzzy i oni koji nisu fuzzy), u ovom ES, predstavljaju informacije o uspešnosti poslovanja a izvedene su na osnovu vrednosti KPI. Sa obzirom na to da bi ove informacije trebalo da budu prilagođene menadžerima, očigledno je da njihovo prikazivanje u vidu nečega što liči na programski kod nema mnogo smisla.

Prvo, sama struktura zaključka u vidu programskog koda (npr. status profita je "upozorenje") može da bude zbunjujuća i da dovede do pogrešnog razumevanja informacije. Smatra se da je najbolje rešenje da se informacije prikažu u formi rečenica kontrolisanog jezika. Kao takve, ih je prošitati i da odmah bude jasno šta se htelo reći. Drugo, primer za status profita koji je naveden ne sadrži objašnjenje o tome zašto je u pitanju "upozorenje" tj. kako se došlo do tog zaključka. Ovakva objašnjenja moraju da budu obezbeđena da bi menadžer mogao da proveri zaključak (u odnosu na podatke) ali i da bi stekao poverenje u sistem. Na kraju, postoji i potreba da se iste informacije prikažu na drugačiji način drugim korisnicima. Izveštaj za generalnog menadžera i za menadžera finansija može da se preklapa u nekim delovima, međutim način prikaza i detaljnost za iste delove izveštaja nisu isti. Takođe, nekad je potrebno i da se isti izveštaj prikaže na nekom drugom jeziku. Sa projektantske tačke gledišta, neophodno je da se ovi prikazi informacija odvoje od pravila tako da dodavanje novih načina prikaza ne remeti pravila iz baze znanja.

Zbog svih prethodno navedenih tvrdnji, u rešenju je upotrebljen i mehanizam za objašnjavanje. Ovaj mehanizam omogava prikaz informacija na prethodno navedeni način. Zapravo, smatra se da je najbolje prikazati informacije kao objašnjenja ES. Od svih vrsta objašnjenja, smatra se da je objašnjenje KAKO najkorisnije. Sa obzirom na to da se najčešće daje po završetku zaključivanja i da predstavlja vodič kako se došlo do zaključka "korak-po-korak", objašnjenje KAKO pruža odgovarajući uvid u poreklo svake informacije.

Što se tiče tehnika za objašnjavanje, smatra se da je potrebno koristiti i učaureni tekst i trag izvršavanja pravila. Učaureni tekst omogućava kreiranje objašnjenja u kontrolisanom jeziku koja liče na rečenice govornog jezika. Takođe, korisno je kada se u sam tekst dinamički unesu i neke vrednosti tj. kada je sam tekst objašnjenja prilagođen unosom konkretnih podataka za taj izveštaj. Nazivi pravila se onda koriste

kao identifikatori, a rečenice učaurenog teksta kao vrednosti tj. objašnjenja konkretnog zaključka koji se donosi kada se dato pravilo izvrši. Na ovaj način se lako dodaju različita objašnjenja za različite korisnike, jer se samo menja učaureni tekst, a identifikatori za objašnjenja ostaju isti. Naravno, pravila ostaju neizmenjena.

Trag izvršavanja pravila identifikuje koja su pravila izvršena u toku sesije i u kojem redosledu. Ovaj listing je veoma koristan pri testiranju sistema i zato se upotrebljava i ovde. Inženjer znanja tj. domenski ekspert može da proveri ovaj trag i utvrdi da li sistem zaključuje kako treba.

Na kraju, kada se izvedu informacije i pretvore u rečenice kontrolisanog jezika, unose se u izveštaj i to uz odgovarajuće podatke. Ovde se koriste već postojeći moduli za prikaz SPI, jer oni često omogućavaju i neke napredne opcije formatiranja izveštaja kao npr. korišćenje šablona. Šabloni za izveštaje su jako korisni jer dozvoljavaju da se isti izveštaj prikaže na različite načine, a da se ne utiče na način dobijanja sadržaja izveštaja.

Diskusija o potencijalnim pozitivnim i negativnim aspektima predloženog rešenja (pristupa) se može naći u poglavlju o evaluaciji.

3.4 Plan realizacije

U cilju što jednostavnije realizacije rešenja, formiran je odgovarajući plan koji se sastoji iz osam koraka. Prva tri koraka postavljaju neke tehničke uslove koji moraju biti ispunjeni u cilju sprovođenja narednih koraka.

1. Naći odgovarajući razvojni alat za bazu znanja (BRMS ili BRE)
2. Naći ili napraviti mehanizam za objašnjavanje
3. Integrisati prethodne dve komponente u SPI
4. Izabrati domen tj. skup KPI čije vrednosti se žele tumačiti
5. Pribaviti i formalizovati znanje za tumačenje KPI
6. Napraviti odgovarajuća objašnjenja
7. Testirati sistem
8. Izvršiti evaluaciju sistema

Prvo, potrebno je izabrati odgovarajući alat za razvoj i testiranje baze znanja. Sa obzirom na to da se znanje predstavlja putem pravila i fuzzy pravila, ovaj alat mora da podržava ove dve tehnike kao i ulančavanje unapred i fuzzy zaključivanje. Kao prirodno rešenje se nameće da je taj alat neki BRMS ili BRE sa mogućnosti neizvesnog zaključivanja. Poželjno je i to da alat bude besplatan za akademsko korišćenje kao i da je otvorenog koda. Radi lakše integracije rešenja u postojeće SPI, neophodno je da ovaj alat bude napisan u nekom od popularnih jezika za objektno orijentisano programiranje, npr. Javi.

Sledeći korak je da se pronađe odgovarajući mehanizam za objašnjavanje (ako ga razvojni alat već ne sadrži). On mora da podržava tehnike učaurenog teksta (sa dinamičkim unošenjem vrednosti na nekim mestima) i traga izvršavanja pravila, kao i da pruža objašnjenje KAKO. Objašnjenja bi trebalo da budu takva da se mogu prilagoditi različitim korisnicima, da eventualno budu na različitim jezicima, a ceo mehanizam bi trebalo da bude jednostavan za integraciju u izabrani alat za razvoj baze znanja. Konkretno, trebalo bi da postoji mogućnost da se prave delovi objašnjenja kako se izvrši koje pravilo.

Treći korak je pronalaženje načina za integraciju sistema koji se pravi uz pomoć ovih alata u postojeći SPI. Logički gledano, u pitanju su dve dodirne tačke sistema sa SPI. Na početku je potrebno preuzeti podatke iz DW ili OLAP izvora podataka, pretvoriti ih u odgovarajući oblik (npr. objekte) i proslediti ih sistemu. Ovo u praksi najčešće znači da bi trebalo postaviti odgovarajući upit i iskoristiti neku transformaciju iz relacione forme podataka u objekte. Posle zaključivanja i formiranja objašnjenja potrebno je informacije uneti u izveštaj zajedno sa podacima. To znači da bi trebalo ostvariti neku povezanost sa ostalim modulima SPI za prikaz izveštaja. Ukratko, možda je najbolje da se iskoriste svi potencijali gotovih izveštaja iz SPI i da se samo na odgovarajuća mesta (uz odgovarajuće grafike, tabele isl.) dodaju tekstualna objašnjenja.

Narednih pet koraka se tiču konkretne primene datog rešenja. Prvo je potrebno izabrati odgovarajući domen koji se želi obraditi. To znači da je potrebno naći neki skup usko povezanih KPI kojima se mere performanse preduzeća ili nekog njegovog dela. Uz to, potrebno je da i postoji jasna procedura kako se njihove vrednosti tumače tj. kako utiču

jedna na drugu da bi znanje za njihovo tumačenje moglo da se prikupi i formalizuje.

Posle izbora domena, neophodno je prikupiti i formalizovati znanje potrebno za tumačenje vrednosti ovih KPI. U ovom slučaju, postoje dva izvora znanja: eksperti i literatura. U zavisnosti od domena, postoji mogućnost da univerzitetski udžbenici, studije slučaja i naučni časopisi pružaju dovoljno znanja o tumačenju KPI.

Svaki bitan zaključak sistema bi trebalo da prati odgovarajuće objašnjenje - informacija. Da bi to bilo moguće, potrebno je napraviti delove učaurenog teksta i povezati ih sa pravilima. Ovaj tekst bi trebalo da bude prilagođen konkretnom korisniku, pa treba predvideti i mogućnost da se napravi više verzija objašnjenja koje bi se aktivirale u zavisnosti od nivoa znanja korisnika, jezika na kojem želi informacije itd. Takođe, potrebno je ostaviti na određenim mestima u tekstu prostor za unos dinamičkih vrednosti, a sve u cilju prilagođavanja objašnjenja.

Detaljno testiranje bi trebalo da usledi nakon prethodnih koraka. Testiranje je potrebno obaviti na nekoliko načina. Prvo, potrebno je proveriti konzistentnost pravila iz baze znanja i izbaciti ili izmeniti sva suvišna pravila, nezadovoljiva ili konfliktna pravila, itd. Nakon toga, testiranje izvršava i ekspert. I ekspert i sistem bi dobili isti skup vrednosti KPI, i trebalo bi proveriti da li se njihovi zaključci poklapaju i da li su objašnjenja tačna i precizna. U toku ove provere je potrebno konsultovati i trag izvršavanja pravila da bi se uočilo da li se pravila aktiviraju i da li je to u odgovarajućem redosledu. Konačno, testiranje drugih aspekata sistema se može izvršiti korišćenjem standardnih tehnika za testiranje softvera.

Poslednji korak čini evaluacija sistema i ona se može uraditi sa dva aspekta. Prvo, predloženo rešenje se može uporediti sa klasičnim SPI sistemom, ili sa nekim sličnim rešenjem i mogu se uočiti značajne sličnosti i razlike kao i prednosti i mane. Drugi aspekt evaluacije čini rad sa krajnjim korisnicima. Bilo bi zanimljivo izmeriti utisak korisnika o korisnosti i jasnoći informacija koje se pružaju kao i o korišćenju sistema uopšte. Nešto strožija i formalnija evaluacija bi mogla da obuhvati eksperiment u kojem bi jedna grupa korisnika koristila običan SPI a druga predloženo rešenje i u kojem bi bilo potrebno da, nakon čitanja izveštaja, odgovore na određena pitanja u vezi sa poslovanjem. Cilj ovog eksperimenta bi bio da se utvrdi da li je tačnost i, eventualno,

brzina davanja odgovora bila veća u onoj grupi u kojoj se koristi dati sistem ili ne.

4 Mehanizam za objašnjavanje

Već je navedeno da je prvi razvojni korak pronalaženje odgovarajućeg razvojnog okruženja za bazu znanja (BRE ili BRMS) a da je, nakon toga, potrebno pronaći ili napraviti i odgovarajući mehanizam za objašnjavanje (ako ga ovo okruženje već ne sadrži). Komparativna analiza savremenih BRE i BRMS napravljenih u Javi (videti sledeće poglavlje) je pokazala da analizirana okruženja ne sadrže mehanizam za objašnjavanje namenjen krajnjim korisnicima niti da implementiraju tražene tehnike za objašnjavanje. Stoga, odlučeno je da se prvo završi drugi razvojni korak, a to je pronalaženje ili razvoj mehanizma za objašnjavanje. Rezultati ovog dela razvoja su tema ovog poglavlja. Evaluacija rezultirajućeg mehanizma za objašnjavanje je data u posebnom poglavlju o evaluaciji celokupnih rezultata istraživanja.

4.1 Komparativna analiza mehanizama za objašnjavanje BRE i BRMS

Većina popularnih BRE-a i BRMS-ova napravljenih u Javi nemaju mehanizam za objašnjavanje namenjen krajnjim korisnicima [95]. Ovo je zaključak komparativne analize mehanizama za objašnjavanje kojom su obuhvaćeni *JESS 7.1* [71], *Exsys Corvid 5.1* [72] i *Drools Expert 5.0* [73] BRE-ovi, kao i *Drools Guvnor 5.0* [74], *Blaze Advisor 6.9* [75] i *JRules 7.0* [76] BRMS-ovi. Samo Exsys Corvid poseduje mehanizam za objašnjavanje za krajnje korisnike. Rezultati pregleda se mogu videti u sledećoj tabeli (Tabela 2).

Tabela 2. Komparativna analiza osobina mehanizama za objašnjavanje BRE i BRMS napravljenih u Javi [95]

	Drools Expert, Drools Guvnor	***Blaze Advisor***	***JRules***	***JESS***	***Exsys Corvid***
Licenciranje i otvorenost koda	Besplatan, otvoren kod	Plaća se, zatvoren kod	Plaća se, zatvoren kod	Plaća se*, zatvoren kod	Plaća se, zatvoren kod
KAKO	DA (samo za testiranje i debag namene)	DA (samo za testiranje, debag namene i kod modela sa ocenom)	DA (samo za testiranje i debag namene)	DA (samo za testiranje i debag namene)	DA
ZAŠTO	NE	NE	NE	NE	DA
STRATEGIJA	DA (samo za testiranje i debag namene)	DA (samo za testiranje i debag namene)	DA (samo za testiranje i debag namene)	DA (samo za testiranje i debag namene)	DA
Trag izvrš. pravila	DA	DA	DA	DA	DA
Učaureni tekst	NE	NE (samo kod mod. sa ocenom)	NE	NE	DA
Domensko znanje	NE	NE	NE	NE	NE
Objašnjenja za tehničke korisnike	DA	DA	DA	DA	DA

	Drools Expert, Drools Guvnor	***Blaze Advisor***	***JRules***	***JESS***	***Exsys Corvid***
Objašnjenja za krajnje korisnike	NE	NE (samo kod mod. sa ocenom)	NE	NE	DA

Drools Expert je besplatni BRE otvoren koda, sa ulančavanjem unapred koji koristi Java objekte kao činjenice, i koji takođe manipuliše Java objektima u samim pravilima. Poseduje specijalnu sintaksu za pisanje pravila koja je veoma slična Javi – Drools Rule Language. Drools Guvnor je BRMS kreiran nad Drools Expert-om i koji ima dodatne mogućnosti neophodne za upravljanje i implementaciju velikih baza znanja u distribuiranim okruženjima, ali ima iste osobine što se tiče mehanizama za objašnjavanje. Oba ova okruženja podržavaju objašnjenja KAKO i STRATEGIJA. Objašnjenje je u obliku detaljnog traga izvršavanja pravila (Audit view) i može biti aktivirano samo tokom testiranja i debagovanja. Svako aktiviranje pravila, promena činjenica, aktivacija meta-pravila i, generalno, događaj je prikazan u ovom tragu. Trag izvršavanja pravila je takođe podržan od strane drugih pogleda koji prikazuju pravila i grupe pravila koja uzimaju u obzir za izvršavanje (Agenda view) kao i činjenice radne memorije (Working memory view). U Drools-u, proces zaključivanja se vodi aktiviranjem *grupa pravila* (Agenda group) uz pomoć meta-pravila. Ovo znači da su objašnjenja KAKO i STRATEGIJA spojena. Tehnički govoreći, objašnjenja su predstavljena kao pseudo kod (zapravo, stablo) i namenjena inženjerima znanja i domenskim ekspertima – tehničkim korisnicima. Krajnji korisnici su ostavljeni bez opcija objašnjenja.

Blaze Advisor je komercijalni BRMS zatvorenog koda koji takođe uključuje ulančavanje unapred i koji može da koristi i manipuliše Java objektima u samim pravilima. Sintaksa koja se koristi za pisanje pravila se naziva Structured Rule Language. Što se tiče mogućnosti objašnjavanja, Blaze Advisor je veoma sličan Drools-

u. On pruža objašnjenja KAKO i (implicitno) STRATEGIJA u formi detaljnih tragova izvršavanja pravila namenjenih testiranju i debagovanju (od strane tehničkih korisnika). Razlika je u tome da on može prikazati grafičku reprezentaciju svih pravila (Execution browser) u kojoj je jasno označen aktuelni trag izvršavanja pravila. Blaze Advisor omogućava modelovanje određenih problema verovatnoće koji koriste modele ocena (score models). Namenjeni su za predviđanje izvesnosti nekih budućih događaja, a rezultat je dat u vidu ocene. I tehnički i krajnji korisnici mogu dobiti objašnjenje kako (KAKO) je ocena ostvarena. Kod rezonovanja (Reasoning code) se dodeljuje svakom pravilu iz modela (u formi učaurenog teksta) i objašnjava razlog zašto je dodeljena određena ocena tokom izvršavanja. Na kraju, listing koda rezonovanja objašnjava finalne ocene, korak po korak. Ovaj tip objašnjenja je moguć samo za modele ocena i ne može biti korišćen od strane regularnih pravila.

JRules je takođe komercijalni BRMS zatvorenog koda sa ulančavanjem unapred i mogućnostima manipulacije Java objektima. Koristi "ILOG Rule Language" za pisanje pravila i ima iste mogućnosti objašnjavanja kao Drools. Objašnjenje je detaljni trag izvršavanja pravila (Debug view), podržan dodatnim tragovima izvršavanja pravila (Agenda view, Working memory view) i namenjen je debagovanju i testiranju. Za razliku od Drools-a i Blaze Advisor-a, JRules poseduje opciju revizije sesija memorišući svaku sesiju kroz svoj trag izvršavanja pravila (Rule execution server). Ovo može biti korisno za pregled prethodnih odluka i zaključivanja, i namenjen je pre svega za optimizaciju revizije i zaključivanja. Međutim, ovo može biti izvršeno samo od strane tehničkog osoblja.

JESS je komercijalni BRE zatvorenog koda, koji se može koristiti besplatno u akademske svrhe. Za razliku od drugih alata prikazanih do sad, poseduje ulančavanje i unapred i unazad. Međutim, kao i svi drugi alati, može manipulisati Java objektima i pružiti objašnjenja KAKO i STRATEGIJA u formi traga izvršavanja pravila. Trag izvršavanja pravila se aktivira tokom testiranja i debagovanja, i sadrži sve bitne događaje. Grupe pravila se nazivaju moduli u JESS-u, a meta-pravila se koriste za određivanje toga koji moduli trebaju biti aktivirani. Izlaz je, stoga, kombinacija KAKO i STRATEGIJA tipova objašnjenja. Nema objašnjenja namenjenog krajnjim korisnicima.

Exsys Corvid je komercijalni BRE zatvorenog koda sa ulančavanjem unapred i unazad. Razlika između njega i drugih alata iz ovog pregleda je da podržava sve tipove objašnjenja i implementira sve tehnike za objašnjavanje osim domenskog znanja. Druga važna razlika je da ne koristi Java objekte kao činjenice, već ima interne promenljive. Ovde je dat je kratak opis mehanizma za objašnjavanje Exsys Corvid-a, a detaljna lista karakteristika je prikazana na kraju sekcije evaluacije u ovom poglavlju. Razlog je taj što smo Exsys Corvid koristili kao reper za evaluaciju naših rezultata.

Takođe je važno napomenuti da neki od prikazanih alata imaju (alternativne) kontrolisane jezike koje mogu biti korišćene za pisanje pravila. Pored "Drools Rule Language" jezika, Drools podržava *domenski specifične jezike* (Domain Specific Language) koji mogu biti kreirani od strane inženjera znanja i prilagođeni određenom domenu. Business Action Language je alternativa za ILOG Rule Language u JRules-u. Blaze Advisor-ov Structured Rule Language je više formalizovan u odnosu na prethodna dva, ali je izražajniji i precizniji. S obzirom na to da su pravila pisana kao izrazi u kontrolisanom jeziku, ideja je da se omogući domenskim ekspertima i krajnjim korisnicima da održavaju bazu pravila. Trag izvršavanja pravila koji sadrži definicije pravila u kontrolisanom jeziku može izgledati kao objašnjenje namenjeno krajnjim korisnicima. Međutim, ovo nije slučaj. Trag izvršavanja pravila definisan na ovaj način ne može pružiti objašnjenje na isti način kao kada se koristi učaureni tekst ili domensko znanje iz tri razloga. Prvo, trag izvršavanja pravila i dalje izgleda tehnički - kao listing pseudo-koda. Svi izrazi imaju jasne IF i THEN delove i činjenice su, do nekog nivoa, izražene u programskoj sintaksi. Drugo, objašnjenja se ne mogu sastojati iz nekoliko rečenica ili pasusa po pravilu. Konačno, ne postoji način za objašnjenje zašto je određeno pravilo uvedeno u bazu znanja.

Druga važna napomena je da većina prikazanih alata ima takođe i napredne mogućnosti izveštavanja. Međutim, ovi izveštaji nisu zamišljeni da budu, i ne mogu biti, pretvoreni u objašnjenja bez dosta dodatnog programiranja. Stoga, ne mogu biti razmatrani kao ugrađene mogućnosti objašnjavanja.

Nakon ove komparativne analize je sprovedena temeljna pretraga za mehanizmom za objašnjavanje u vidu aplikacijskog okvira ili biblioteke koji bi se mogao koristiti sa

navedenim BRE i BRMS. Pretraga nije dala nikakve rezultate. Koliko je autoru poznato, ne postoji mehanizam za objašnjavanje koji zadovoljava ove kriterijume.

4.2 Ciljevi

Pošto odgovarajući mehanizam za objašnjavanje (koji se može koristiti sa BRE i BRMS napisanim u Javi) nije mogao biti pronađen, jedina preostala opcija je bila da se napravi.

Mehanizam za objašnjavanje je nazvan *JEFF – Java Explanation Facility Framework*. Odlučeno je da JEFF bude besplatan i otvorenog koda, i da pruža KAKO i, ako je moguće, STRATEGIJA vrste objašnjenja. Sa obzirom na to da se za predstavljanje informacija u vidu rečenica kontrolisanog jezika traže samo ovi tipovi objašnjenja, odlučeno je da se objašnjenje ZAŠTO ne implementira. Takođe, definisan je skup primarnih ciljeva:

1. Jednostavnost korišćenja.
2. Jednostavna integracija sa bilo kojim ES, BRE ili BRMS napisanim u Javi.
3. Objašnjenja u formi rečenica kontrolisanog jezika, kao i traga izvršavanja pravila.
4. Objašnjenja trebaju, opcionalno, uključiti i druge vrste sadržaja osim teksta (slike, tabele, grafike, itd).
5. Mogućnost pružanja objašnjenja na različitim jezicima bez modifikacije samog mehanizma.
6. Mogućnost pružanja objašnjenja u različitim izlaznim formatima.
7. Proširivost.

Prvi cilj ne zahteva objašnjenje, ali drugi implicitno uslovljava da JEFF bude implementiran kao *aplikacijski okvir* u Javi (software framework). Svi BRE i BRMS iz komparativne analize imaju opciju za izvršavanje Java koda u samim pravilima (obično u THEN delu pravila). Stoga, najdirektniji način za korišćenje JEFF-a u ovim razvojnim okruženjima je pozivanje JEFF-ovih (Java) metoda od strane samog okruženja.

Kao što je naglašeno u prethodnim sekcijama, objašnjenje mora biti prilagođeno korisniku. Treći cilj uključuje nameru da se pruže objašnjenja tehničkim korisnicima (inženjeri znanja i eksperti), kao i krajnjim korisnicima. Objašnjenja u kontrolisanom jeziku su najpogodnija za ove druge jer ih mogu čitati i razumeti bez potrebe za daljim objašnjenjima, dok je trag izvršavanja pravila obično dovoljan za ove prve. Ovaj cilj implicitno definiše da mehanizam za objašnjavanje treba podržati aktivnosti testiranja i debagovanja, takođe.

Nekim korisnicima je tekstualni sadržaj najjednostavniji za razumevanje, dok drugi preferiraju grafičke ili druge vrste prezentacije kako bi bolje razumeli tvrdnje iznete u objašnjenjima. Stoga, JEFF treba da pruži opciju za ubacivanje drugih tipova sadržaja u objašnjenja kao što su slike ili podaci u vidu tabela i grafika.

Jedan od ciljeva je internacionalizacija JEFF-a. Mora imati mogućnost generisanja objašnjenja u različitim jezicima, bez potrebe za reprogramiranjem, redizajnom ili bilo kojom drugom izmenom samog aplikacijskog okvira.

Sledeći cilj je usmeren kao povećavanju upotrebljivosti jer različiti izlazni formati omogućavaju različite načine upotrebe. Na primer, PDF može biti pogodan kao finalni izlaz za krajnje korisnike, dok XML može biti koristan za kompleksne transformacije u druge formate tražene od strane programera i drugog tehničkog osoblja.

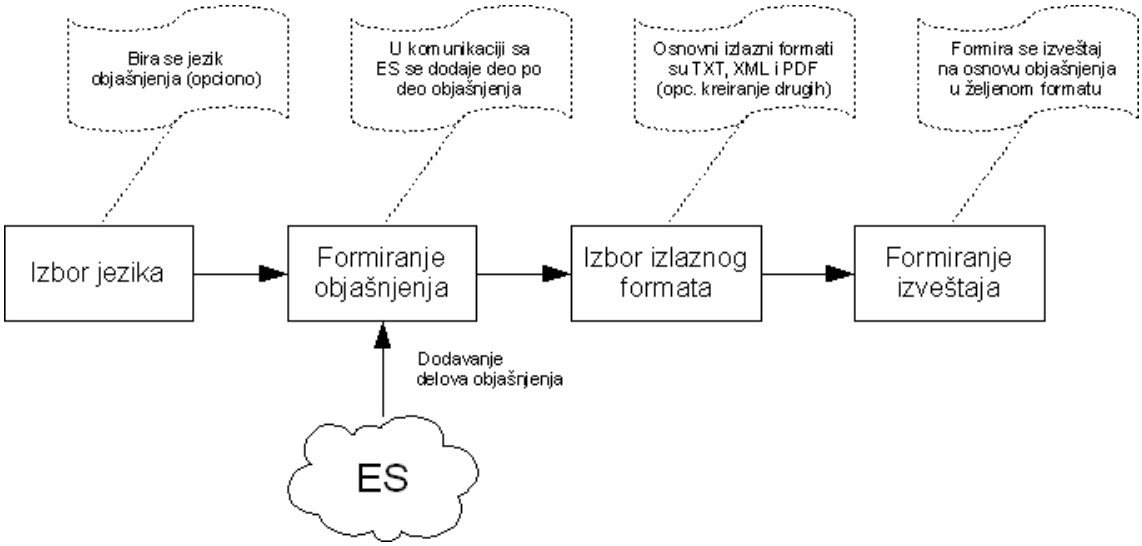
Poslednji cilj definiše da JEFF mora biti proširiv. Dodavanje novih elemenata ili izmena postojećih bi trebalo da bude jednostavno i lako.

4.3 Način rada

Način rada JEFF mehanizma za objašnjavanje se može opisati u četiri koraka (Slika 22).

U prvom koraku, korisnik definiše svoje jezička podešavanja izborom željenog jezika i zemlje. Ovo poslednje pomaže kod razlikovanja između dijalekata istog jezika (npr. britanski engleski i američki engleski). U slučaju da nema potrebe za pružanjem objašnjenja na različitim jezicima, ovaj korak može biti preskočen izborom podrazumevanog jezika i zemlje. Kada je ovaj korak završen, tj. kad sledeći počne, podešavanja jezika ne mogu biti izmenjena.

Drugi korak je najkompleksniji jer uključuje formiranje objašnjenja u traženom jeziku. U ovom koraku, JEFF komunicira sa BRE-om, BRMS-om ili ES-om i proces formiranja objašnjenja se dešava paralelno procesu zaključivanja. Kreiranje objašnjenja počinje kada proces zaključivanja startuje i pojedinačni *delovi objašnjenja* (explanation chunks) bivaju ubačeni u objašnjenje dok se pravila izvršavaju. Kada se zaključivanje zaustavi, objašnjenje je gotovo. Delovi objašnjenja su stoga dodati sekvencijalno objašnjenju i mogu predstavljati neki tekst (jedan ili više pasusa), podatke (reprezentovane tabelom ili grafikom) ili sliku. Njihov redosled predstavlja redosled u kojem se vrše individualna zaključivanja i jedan deo objašnjenja (ili više) treba da objasni svaki važan zaključak.



Slika 22: JEFF - način rada

Izlaz iz drugog koraka je objašnjenje u sirovoj formi koje je nepogodno za čitanje. Zbog toga ga je neophodno transformisati u čitljiv izveštaj. U trećem koraku korisnik bira format izveštaja. U ovom trenutku, JEFF može kreirati izveštaje u fori običnog teksta, XML-a i PDF-a.

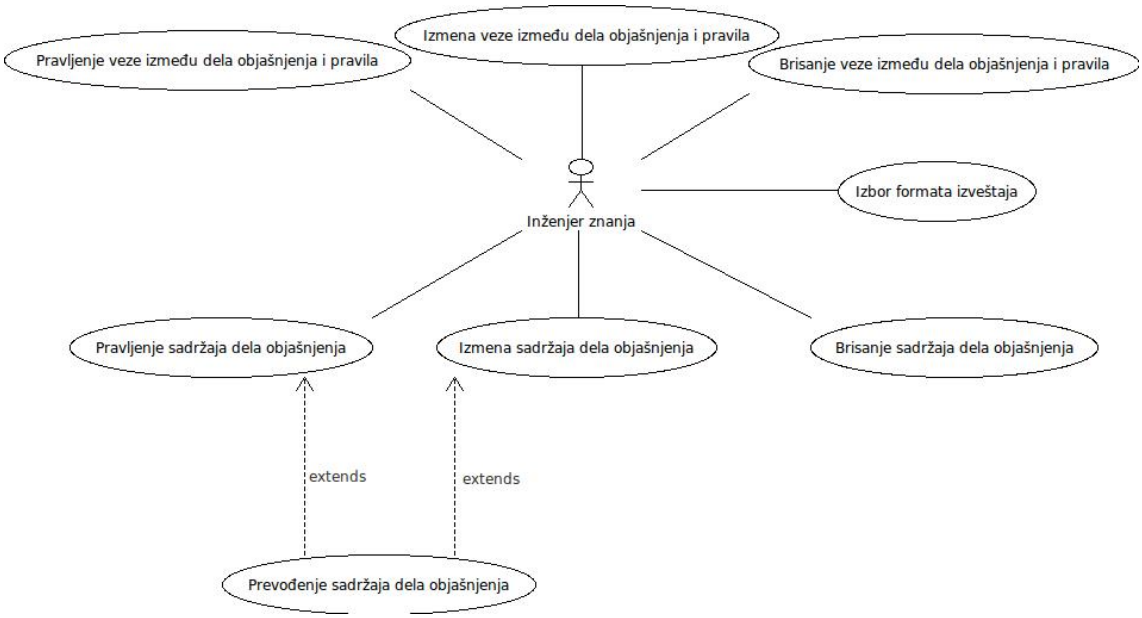
Poslednji korak uključuje transformisanje objašnjenja u izveštaj u izabranom formatu. Ovaj izveštaj može biti sačuvan kao fajl (XML, PDF ili tekstualni) ili prosleđen nekom izlaznom toku radi prezentacije u browser-u, korišćenja od strane Web servisa ili čak prikazivanja na standardnom izlaznom toku – *konzoli*. Prikaz objašnjenja na konzoli se može koristiti kao izvršni log ili u svrhe testiranja.

4.4 Slučajevi korišćenja

Prethodno predstavljeni način rada implicitno definiše tri uloge (aktora) i nekoliko slučajeva korišćenja za svaku od njih. Uloge su: inženjer znanja, krajnji korisnik i BRE/BRMS. Pri tome, poslednja uloga zapravo predstavlja softverski sistem BRE ili BRMS koji poziva JEFF, a ne neku osobu.

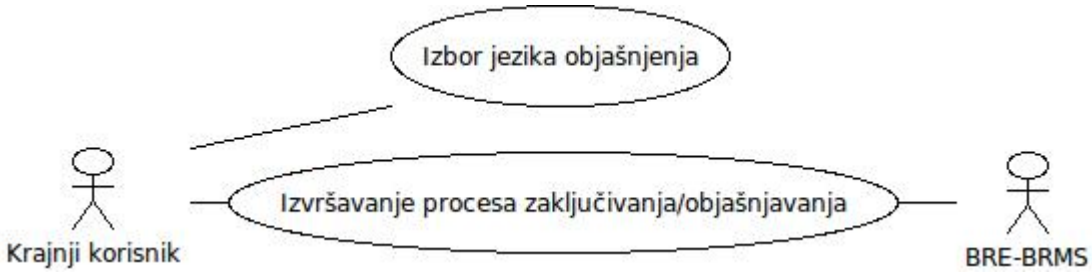
Inženjer znanja projektuje sadržaj objašnjenja i kreira veze između tog sadržaja i odgovarajućih zaključaka (zapravo, pravila) iz baze znanja BRE tj. BRMS (Slika 23). Zbog toga, kada BRE ili BRMS pokrene svoje zaključivanje, objašnjenja mogu biti formirana automatski dodajući ranije definisani sadržaj objašnjenju kada se svaki važan zaključak donese tj. kada se aktivira odgovarajuće pravilo. U slučaju korišćenja za pravljenje sadržaja dela objašnjenja, inženjer znanja kreira sadržaj koji će biti upotrebljen za objašnjenje bez obzira da li je to neki tekst, slika ili predefinisani podatak. Ako postoji potreba za pružanjem objašnjenja na različitim jezicima, neophodno je prevesti deo sadržaja objašnjenja. Koraci su u ovom slučaju isti, osim što se tekst, naslovi slika i naslovi tabela prvo prevode, a onda unose. Prirodno, inženjer znanja poseduje opciju izmene i brisanja dela sadržaja objašnjenja.

Kada se sadržaj dela objašnjenja pripremi, inženjer znanja može praviti, menjati i brisati veze izmedju dela objašnjenja (zapravo njegovog sadržaja) i pravila. Ove veze omogućavaju automatsko dodavanje odgovarajućeg sadržaja u objašnjenja kako se pravilo aktivira. Jedno pravilo može biti spojeno sa nula, jednim ili više instanci delova objašnjenja, time ne ograničavajući objašnjenje nekog zaključka na samo jedan tip sadržaja. Konačno, inženjer znanja bira i format izveštaja. Razlog za to je što on ima uvid u to koji format je najpogodniji za primaoca izveštaja. Treba imati u vidu da ovaj slučaj korišćenja odgovara trećem koraku načina rada opisanog u prethodnom poglavlju.



Slika 23: Slučajevi korišćenja inženjera znanja

Krajnji korisnik je primalac izveštaja (Slika 24). Na početku, korisnik mora izabrati jezik (iz liste ponuđenih) u kojem će izveštaj biti kreiran. Ovaj slučaj korišćenja predstavlja prvi korak načina rada opisanog u prethodnom poglavlju, tako da neće biti detaljno objašnjen ovde. Po završetku, proces zaključivanja/objašnjavanja može da se izvrši. Pored krajnjeg korisnika, *BRE/BRMS* predstavlja drugu ulogu u ovom slučaju korišćenja. Izvršavanje slučaja korišćenja procesa zaključivanja/objašnjavanja počinje pokretanjem procesa zaključivanja. On može biti pokrenut od strane krajnjeg korisnika ili od samog BRE/BRMS (kroz neki događaj koji nije kreiran od strane krajnjeg korisnika). Proces izrade objašnjenja se izvršava paralelno sa procesom zaključivanja (videti korak dva iz prethodnog poglavlja) i, kada se završi, izvršava se proces izrade izveštaja (videti korak četiri). Slučaj korišćenja se završava isporukom izveštaja u traženom jeziku i formatu krajnjem korisniku.

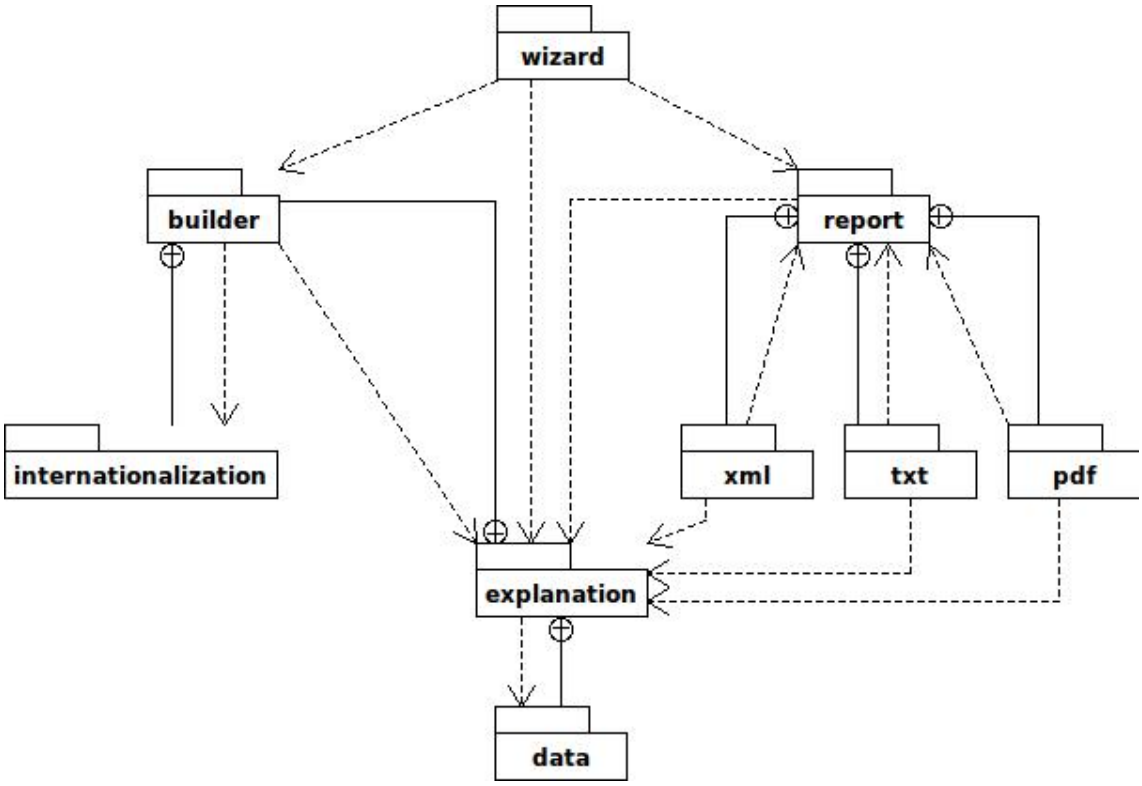


Slika 24: Slučajevi korišćenja krajnjeg korisnika i BRE/BRMS

4.5 Arhitektura i klase

Arhitektura JEFF-a se može videti na sledećem UML dijagramu paketa (Slika 25). Ona ilustruje pakete u JEFF-u, njihovu hijerarhiju, kao i zavisnosti. Zavisnosti su predstavljene strelicama, kako je to uobičajeno u UML-u, a hijerahija je predstavljena na nestandardan način kako bi bila čitljivija. Svaka veza u hijerarhiji je predstavljena sa linijom koja počinje krugom koji ima znak plus unutra. Paket na početku linije sadrži paket na kraju linije. Na primer, *builder* paket sadrži *internationalization* paket, ali *explanation* paket sadrži *builder* paket. Takođe, *wizard* paket ne sadrži ni jedan paket niti je deo nekog drugog paketa. Potrebno je napomenuti da su se, u toku projektovanja arhitekture i klasa JEFF-a koristili *softverski paterni* (software pattern)[96][97][98] i to sa ciljem obezbeđivanja lake nadogradivosti, proširivosti ali i održavanja sistema.

Explanation paket je osnovni paket jer sadrži sve elemente (klase i interfejse) potrebno za predstavljanje objašnjenja i delova objašnjenja. Može se primetiti da skoro svi drugi paketi i njihovi elementi zavise od elemenata ovog paketa. Elementi *explanation* paketa poseduju zavisnost prema elementima *data* paketa. *Data* paket je deo *explanation* paketa i sadrži sve elemente neophodne za predstavljanje podataka u objašnjenjima.



Slika 25: Arhitektura JEFF-a

Na levoj strani dijagrama se može videti *builder* paket. Ovaj paket predstavlja deo *explanation* paketa i njegovi elementi implementiraju funkcionalnost potrebnu za pravljenje objašnjenja tokom procesa zaključivanja. Njegovi elementi se oslanjanju na elemente u *internationalization* paketu, kako bi pružili prevode objašnjenja u traženom jeziku.

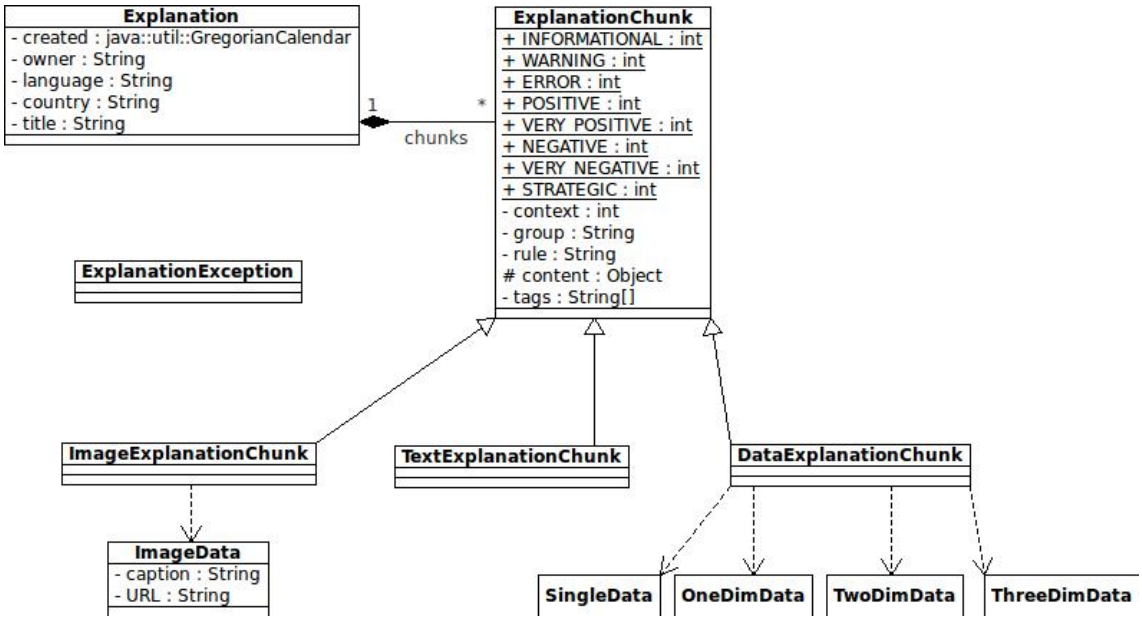
Report paket i njegovi podpaketi (*xml*, *txt* i *pdf)* sadrže sve elemente neophodne za transformisanje objašnjenja u izveštaj u traženom formatu. Kao što je naglašeno, trenutno postoje tri formata: XML, običan tekst i PDF.

Konačno, paket *wizard* i njegovi elementi su uvedeni kako bi se omogućilo jednostavnije korišćenje. Cilj je omogućiti inženjerima znanja da iskoriste potpunu funkcionalnost JEFF mehanizma za objašnjavanje kroz jedinstvenu ulaznu tačku (jednu klasu) i kroz komadne koje trebaju biti što jednostavnije.

Dijagram klasa koji predstavlja neke od elemenata iz *explanation* paketa se može videti na sledećoj slici (Slika 26). Sve metode u ovim klasama se koriste za preuzimanje i

postavljanje vrednosti atributa (tzv. *getXXX* i *setXXX* metode za enkapsulaciju atributa), tako da su uklonjeni sa dijagrama. *Explanation* klasa predstavlja objašnjenje. Ona sadrži generalne podatke objašnjenja o tome kada je kreirano (*created* atribut), ko može da ga čita (*owner*), u kom jeziku je napisano (*language*) i iz koje zemlje (*country*), i koji je naslov objašnjenja (*title*).

Svako objašnjenje može imati nula, jedan ili više delova objašnjenja (apstraktna klasa *ExplanationChunk*). Svaki deo ima svoj kontekst (*context* atribut), koji objašnjava koje je generalno značenje tog sadržaja. Ovo znači da deo može predstavljati upozorenje (*WARNING* konstanta), grešku (*ERROR*), neku pozitivnu poruku (*POSITIVE*), veoma pozitivnu poruku (*VERY_POSITIVE*), negativnu poruku (*NEGATIVE*), veoma negativnu poruku (*VERY_NEGATIVE*), strateško usmeravanje procesa zaključivanja (*STRATEGIC*) ili može predstavljati neku informaciju (*INFORMATIONAL*). *STRATEGIC* kontekst se koristi kada se formira objašnjenje STRATEGIJA, dok se drugi konteksti koriste za objašnjenje KAKO. Pošto se JEFF koristi i za debagovanje i testiranje, uvedeni su *rule* i *group* atributi. Oni predstavljaju identifikator pravila na koje se odnosi deo objašnjenja kao i identifikator grupe pravila kojoj pravilo pripada (ako su pravila podeljena u grupe). Ovo omogućava formiranje jednostavnog traga izvršavanja pravila koji se može koristiti za testiranje i debagovanje. Međutim, samo će pravila koja imaju pridružene delove objašnjenja biti uključena u trag izvršavanja pravila. Atribut *tags* predstavlja niz ključnih reči i izraza koji omogućavaju fleksibilnije opise ili klasifikaciju za svaki deo objašnjenja. Oni su opcioni i mogu biti korišćeni da označe sadržaj u slučaju da objašnjenje (ili izveštaj) treba da bude transformisan na kompleksan način korišćenjem nekih kriterijuma. Atribut *content* predstavlja sadržaj dela objašnjenja. Pošto je *ExplanationChunk* apstraktna klasa, njene konkretne podklase određuju tip sadržaja implementirajući apstraktnu metodu *setContent*. U *TextExplanationChunk* podklasi, sadržaj je običan niz slova (instanca Java klase String). Sadržaj *ImageExplanationChunk*-a je instanca *ImageData* klase koja sadrži informacije o nazivu slike i njenom URL-u. Konačno, instance *DataExplanationChunk* klase sadrže podatke i predstavljene su kao klase iz data paketa: *SingleData*, *OneDimData*, *TwoDimData* i *ThreeDimData*.



Slika 26: Elementi explanation paketa

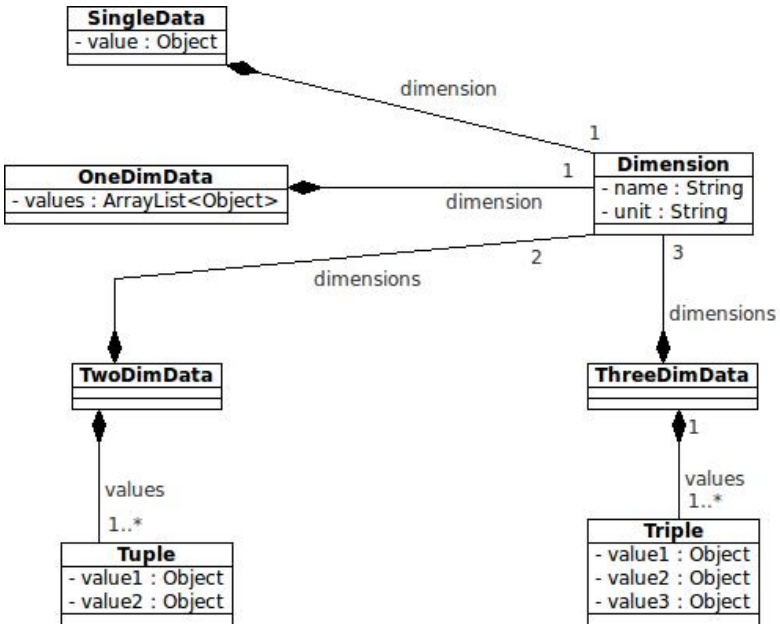
Data paket (Slika 27) sadrži sve klase potrebne za čuvanje podataka koji se unose u objašnjenje. Dakle, one samo čuvaju podatke koji bi trebalo da budu predstavljeni putem tabele (npr. lista cena) ili grafika, ali ne i one podatke koji bi trebalo da budu uključeni u tekst (npr. „temperatura je 17 stepeni“). Osnovne klase u ovom paketu su *SingleData*, *OneDimData*, *TwoDimData* i *ThreeDimData*. One su međusobno veoma slične, a razlika između je u dimenzionalnosti podataka koje sadrže. Klase *SingleData* i *OneDimData* sadrže jednodimenzionalne podatke, *TwoDimData* se odnosi na dvodimenzionalne podatke, a *ThreeDimData* može sadržati trodimenzionalne podatke. Sve ove klase poseduju relaciju kompozicije sa *Dimension* klasom (kardinalnost ove relacije određuje dimenzionalnost). Ova klasa sadrži naziv dimenzije (npr. „cena“ ili „dužina“) i opcionalno, naziv merne jedinice te dimenzije (npr. „dolar“ ili „metar“).

SingleData klasa je najjednostavnija i time najlakša za razumevanje. Može sadržati samo jednu vrednost i njene dimenzije. Na primer, ako nešto košta 20 dolara, ova klasa može biti iskorišćena da čuva taj podatak. Naziv dimenzije bi bio „cena“, a naziv jedinice „dolar“. Vrednost podataka može biti bilo kog tipa: integer, decimal, character, string, boolean ili čak objekat.

OneDimData predstavlja jednodimenzionalni niz vrednosti. Na primer, lista svih

proizvoda koji zadovoljavaju određeni kriterijum može biti prikazana korišćenjem ove klase. U ovom slučaju, naziv dimenzije bi bio „naziv proizvoda“, a jedinica bi bila izostavljena.

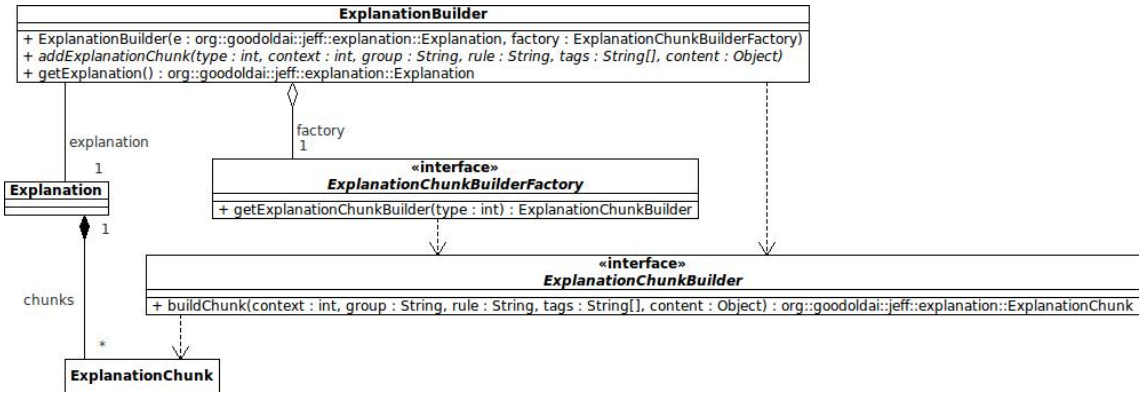
Kao što je naglašeno, klase *TwoDimData* i *ThreeDimData* se koriste za predstavljanje dvodimenzionalnih i trodimenzionalnih nizova podataka. Vrednosti podataka su predstavljene nizom dvojki (*Tuple* klasa) ili trojki (*Triple* klasa) respektivno. Stoga, ako je neophodno predstaviti listu cena vina, *TwoDimData* klasa će se koristiti. Prva dimenzija bi bila „vino“ i ne bi imala jedinicu, dok bi naziv druge dimenzije bio „cena“ koja bi imala „dolar“ kao jedinicu. Vrednosti bi bile predstavljene dvojkama: {“Monterra”, 57.50}, {“Maryvale”, 87.50}, itd. Ako bi postojala i potreba za predstavljanjem godine kada je vino flaširano, koristila bi se *ThreeDimData* klasa. Prve dve dimenzije bi bile iste, dok treća dimenzija pod nazivom „godina“ ne bi imala jedinicu, a vrednosti bi bile predstavljene trojkama: {“Monterra”, 57.50, 1999}, {“Maryvale”, 87.50, 1983}, itd.



Slika 27: Elementi data paketa

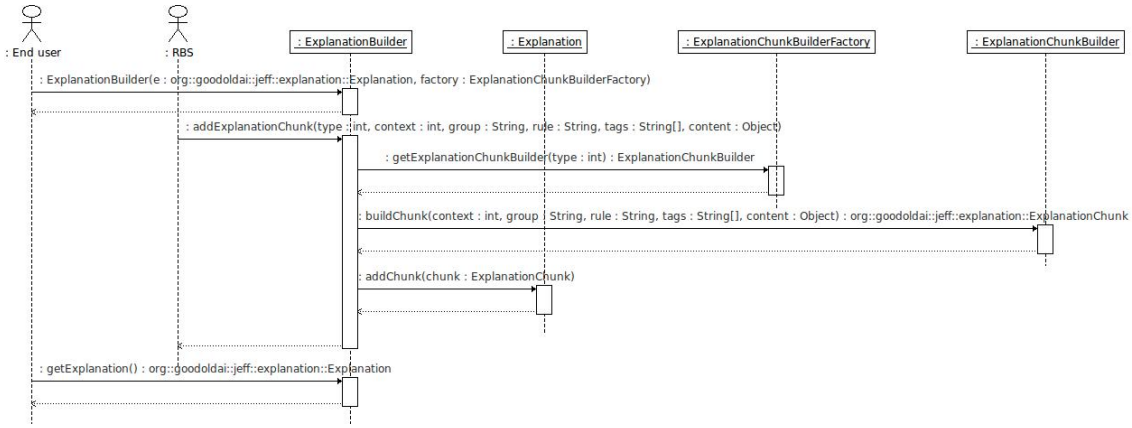
Osnovni elementi *builder* paketa se mogu videti na sledećem dijagramu (Slika 28). Oni sadrže apstraktne klase i interfejse. Podrazumevana implementacija ovih elemenata je prisutna u samom paketu, ali nije predstavljena na dijagramu. *ExplanationBuilder*

apstraktna klasa je osnovna klasa odgovorna za kreiranje objašnjenja tokom procesa zaključivanja i nastala je kao posledica primene *Builder* paterna. Svaka instanca *ExplanationBuilder* klase je odgovorna za kreiranje tačno jedne instance *Explanation* klase. Proces izgradnje je postepen, i kako se pravila aktiviraju, instance *ExplanationChunk* klase se kreiraju i umeću u objašnjenje. Klase *ExplanationChunkBuilder* i *ExplanationChunkBuilderFactory* su nastale kao posledica primene paterna *Builder* i *Factory*.



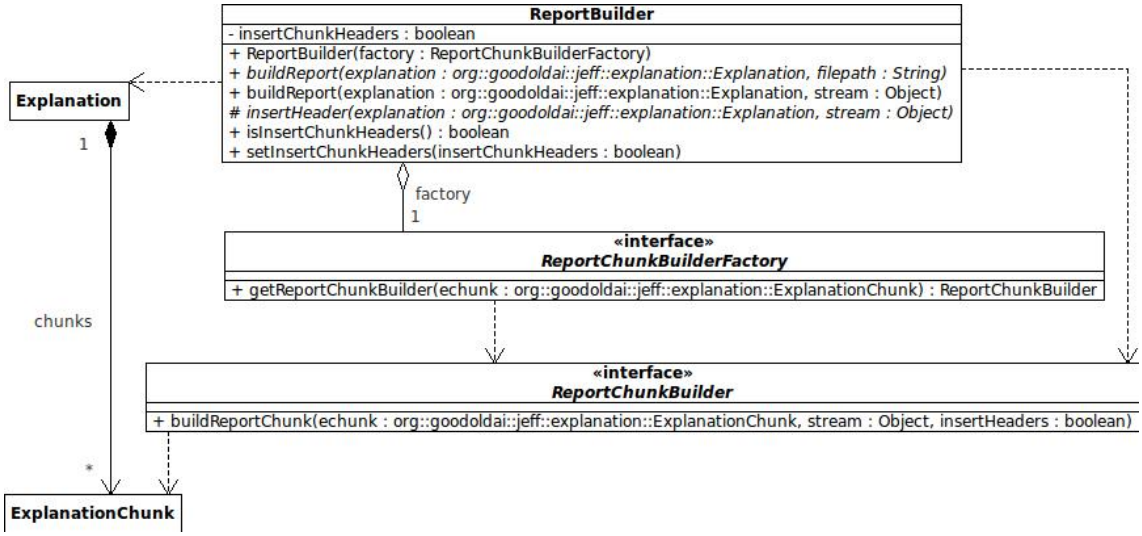
Slika 28: Elementi builder paketa

Ovaj proces je prikazan na sledećem dijagramu sekvenci (Slika 29). Metoda *addExplanationChunk* iz *ExplanationBuilder* klase prima podatke potrebne za izradu dela objašnjenja i poziva metodu *getExplanationChunkBuilder* iz *ExplanationChunkBuilderFactory* interfejsa. Nakon toga, metoda koristi primljenu instancu *ExplanationChunkBuilder*-a i njegovu *buildChunk* metodu za izradu odgovarajućeg tipa dela objašnjenja na osnovu ulaznih podataka. Ako sadržaj treba da se prevede, *buildChunk* metoda koristi pomoćne klase iz *internationalization* paketa. Konačno, *addExplanationChunk* metoda unosi kreirani deo u objašnjenje. Novi deo se dodaje na kraj objašnjenja, čuvajući time red donošenja zaključaka (a i objašnjenja). Kada je objašnjenje završeno, može se dobiti pozivanjem metode *getExplanation*. Povratna vrednost je instanca *Explanation* klase koja sadrži nula, jednu ili više instanci *ExplanationChunk* klase sa sadržajem na traženom jeziku.



Slika 29: Dijagram sekvence formiranja objašnjenja

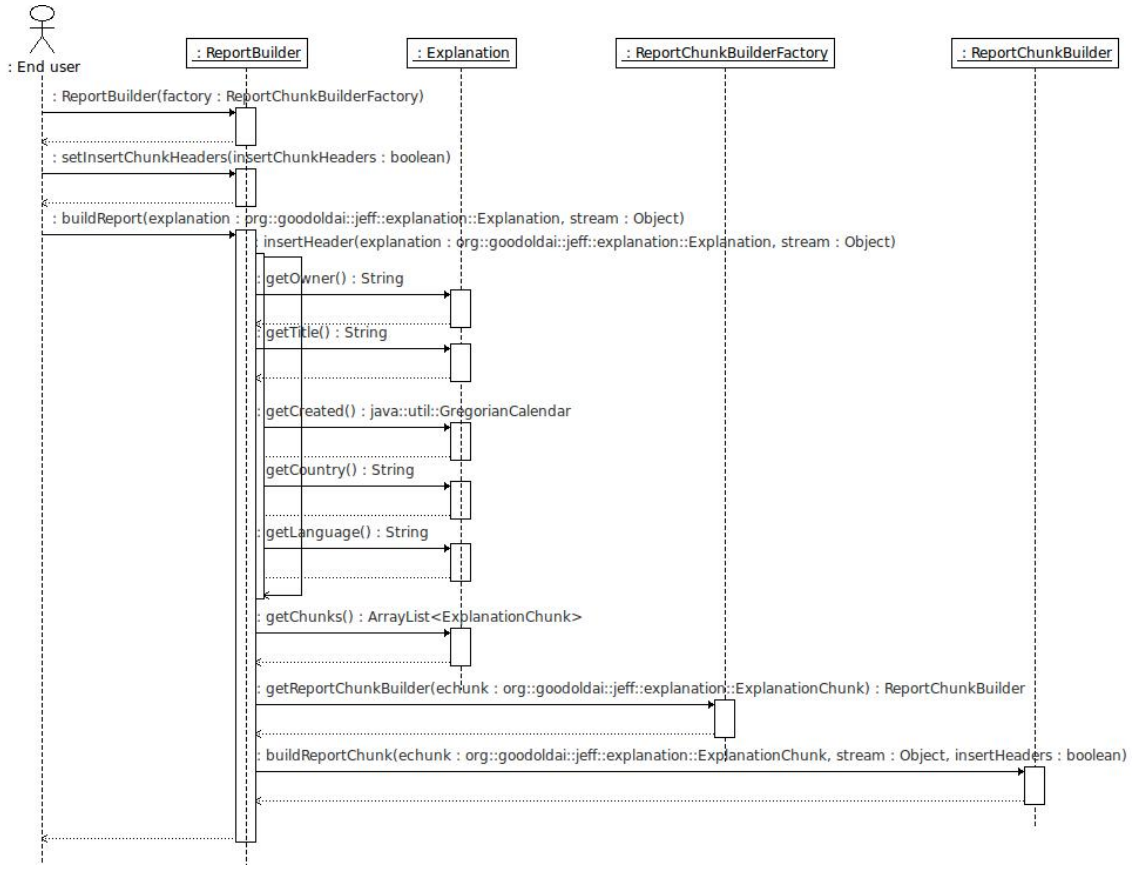
Elementi u *report* paketu imitiraju elemente u *builder* paketu do nekog nivoa a nastale su kao posledica upotrebe istih paterna (Slika 30). Apstraktna klasa *ReportBuilder* je odgovorna za kreiranje izveštaja baziranih na pruženom objašnjenju. Atribut *insertChunkHeader* označava da li izveštaj treba da sadrži trag izvršavanja pravila (true) ili ne (false), tj. da li će se izveštaj koristiti za testiranje i debagovanje ili ne. Dve *buildReport* metode koje se koriste za formiranje izveštaja rade na sličan način, ali jedna kreira fajl koji sadrži izveštaj, dok druga prosleđuje izveštaj ka nekom izlaznom toku.



Slika 30: Elementi report paketa

Proces formiranja izveštaja se može videti na sledećem dijagramu sekvence (Slika 31). Kada se pozove, *buildReport* metoda prvo formira naslov izveštaja pozivajući *insertHeader* metodu. Ova metoda uzima opšte podatke objašnjenja (vlasnik, naslov,

datum i vreme kreiranja, jezik i zemlja), kreira naslov koristeći ove podatke i unosi ih u izveštaj. Potom *buildReport* metoda uzima sve delove objašnjenja i izvršava sledeću proceduru za svaki. Poziva *getReportChunkBuilder* metodu iz *ReportChunkBuilderFactory* interfejsa i koristi postojeću instancu *ReportChunkBuilder*-a i njegovu *buildReportChunk* metodu za transformaciju dela objašnjenja u deo izveštaja. Kada se ovo završi za sve delove objašnjenja, izveštaj se vraća krajnjem korisniku. Važno je primetiti da konkretne implementacije ovih klasa i interfejsa zavise od formata izveštaja. Postoje tri podrazumevane implementacije: za PDF, običan tekst i XML formate, a nalaze se u *pdf*, *txt* i *xml* paketima respektivno.

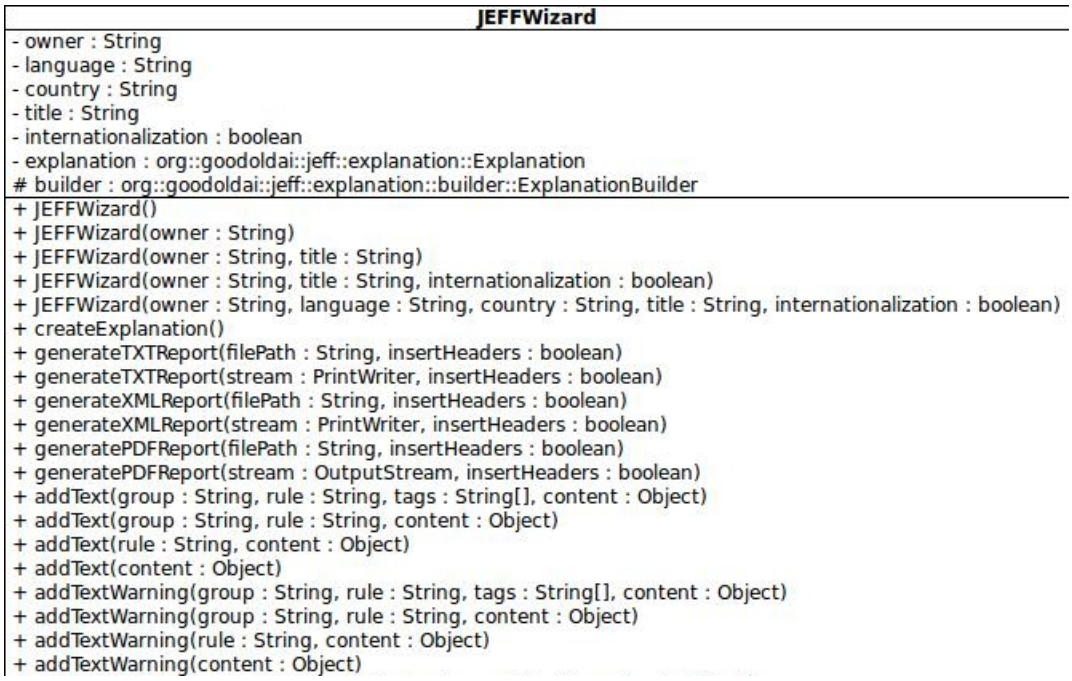


Slika 31: Dijagram sekvence formiranja izveštaja

Wizard paket sadrži samo jednu klasu – *JEFFWizard*. Sledeći dijagram klasa prikazuje samo deo ove klase, jer ima mnogo metoda, a svaka ima mnogo varijacija (Slika 32). *JEFFWizard* klasa je uvedena kako bi pružila jedinstvenu ulaznu tačku u ceo sistem tj. nastala je kao posledica primene paterna *Facade*. Ova klasa pruža najjednostavniji način

inženjerima znanja da kreiraju objašnjenja i da prave različite izveštaje bazirane na datom objašnjenju. Sve se nalazi na jednom mestu, tako nema potrebe za upoznavanjem sa celim aplikacijskim okvirom i njegovim funkcionalnostima. Nazivi metoda su intuitivniji i lakši za pamćenje, i inženjer znanja ne mora da brine da li radi sa pravim klasama, metodama itd.

Što se tiče atributa, *JEFFWizard* sadrži podatke o vlasniku objašnjenja, jeziku, zemlji i naslovu i kreira instancu *Explanation* klase sa ovim podacima kada se pozove *createExplanation* metoda. Atribut *internationalization* označava da li postoji potreba za kreiranjem objašnjenja na više jezika (true) ili ne (false). U zavisnosti od te vrednosti, različite implementacije *ExplanationBuilder* apstrakne klase će biti instancirane u *builder* atributu i korišćene za izradu objašnjenja. Sve *addXXX* metode se koriste za dodavanje instanci odgovarajuće *ExplanationChunk* podklase u objašnjenje. Na primer, *addText* metoda se koristi za dodavanje *TextExplanationChunk* instance sa *INFORMATIONAL* kontekstom, a *addTextWarning* metoda se takođe koristi za dodavanje *TextExplanationChunk* instance, ali sa *WARNING* kontekstom. Svaka od *addXXX* metoda ima nekoliko varijacija u zavisnosti od ulaznih parametara. Konačno, kada je potrebno napraviti izveštaj, odgovarajuća *generateXXXReport* metoda se poziva, u zavisnosti od traženog izlaznog formata. Inženjer znanja jedino treba da izabere da li izveštaj treba da bude sačuvan u fajl ili prosleđen izlaznom toku. Svaka *generateXXXReport* metoda ima dve varijacije, jednu za svaki od slučajeva. Naravno, uvek postoji opcija da se *JEFFWizard* ne koristi, gde u tom slučaju inženjer znanja koristi regularne klase iz svih drugih paketa.



Slika 32: Elementi wizard paketa – JEFFWizard klasa

4.6 Implementacija i testiranje

JEFF je besplatan i otvoren koda, tako da se nalazi na Internetu gde može preuzeti besplatno i bez registracije [87]. Aktuelna verzija, JEFF 1.0.0 beta i neka njena skorašnja poboljšanja su predstavljeni u ovom radu. Prethodna verzija, JEFF 0.1.0 prealpha, se može preuzeti sa iste Internet lokacije i delimično je opisana u jednom kraćem radu [88].

JEFF je implementiran kao aplikacijski okvir u programskom jeziku Java. Jedan od ciljeva JEFF-a je jednostavna integracija sa postojećim BRE, BRMS, i ES napisanim u Javi. Svi oni alati koji su obuhvaćeni u komparativnoj analizi imaju opcije pozivanja Java metoda u samim pravilima, rada sa Java objektima i izvršavanje Java komandi unutar i van baze znanja. Najdirektniji način korišćenja JEFF-a u ovim razvojnim okruženjima je kreiranje poziva Java metoda do i od njega od strane okruženja i to je razlog zašto je JEFF implementiran kao aplikacijski okvir u Javi. On nema *grafički korisnički interfejs* (GUI) i može mu se pristupiti kroz njegov *API* (Application Programming Interface). Alati sa kojima se JEFF koristi imaju veoma različite GUI-je,

pa jedinstven pristup u kreiranju GUI-a koji može biti integrisan sa ovim alatima nije lako ostvarljiv.

Jedan od najvećih izazova u fuzzy implementacije je omogućavanje *internacionalizacije* za JEFF. To podrazumeva identifikaciju svih sadržaja koji trebaju biti prevedeni, njihovo uklanjanje iz softverskog okruženja i enkapsuliranje u neke module ili jedinice koje se mogu dodati, menjati ili ukloniti bez uticaja na softversko okruženje. Identifikovani sadržaj se sastoji od tekstualnih opisa, naslova objašnjenja, naziva slika, naziva dimenzija i jedinica.

Srećom, Java poseduje mogućnost internacionalizacije (*i18n* u kratko) [89], tako da se koristi u JEFF-u. Rešenje se sastoji od pružanja skupa *Java property fajlova* koji čuvaju sadržaj koji treba da se prevede, i potom korišćenjem neke od predefinisanih Java klasa za uzimanje ovog sadržaja, kada je potrebno. Novi jezici mogu biti podržani dodavanjem fajlova sa prevedenim sadržajem. Property fajlovi su obični tekstualni fajlovi, ali sa određenom strukturom. Sastoje se od više *parova ključ-vrednost*, gde se ključ nalazi na početku praćen znakom jednakosti, a vrednost je nakon ovog znaka. U ovom slučaju, vrednost predstavlja sadržaj koji bi trebalo prevesti. Prevod se čuva u drugom property fajlu i ima isti ključ (koji se koristi za njegovu identifikaciju), ali je vrednost prevedeni sadržaj. Ovi property fajlovi imaju striktne konvencije imenovanja. Naziv se sastoji od osnovnog naziva (unetog od strane korisnika), praćenog kratkim nazivom jezika i kodnim nazivom zemlje [89]. Java i18n pruža dodatne mogućnosti, kao što su, umetanje dinamičkih vrednosti u predefinisana mesta u prevodu, pa je i to iskorišćeno.

U JEFF-u, četiri property fajla se koriste za podršku svakog jezika. Prvi sadrži kompletan tekst sadržaja dela objašnjenja, drugi sadrži sve nazive slika zajedno sa naslovom objašnjenja, a treći i četvrti sadrže nazive dimenzija i jedinica, respektivno. Ako je baza znanja velika i mora biti podeljena u grupe pravila (module), nazivi tih grupa mogu biti korišćeni za podelu kompletnog teksta sadržaja objašnjenja na više property fajlova.

Pored običnog teksta, JEFF može kreirati izveštaje u XML i PDF formatima. Za kreiranje XML-a JEFF koristi dom4j aplikacijski okvir [90], dok za PDF koristi iText

aplikacijski okvir [91]. Oba su besplatna i otvorenog koda, i mogu se koristiti pod sličnim opcijama licenciranja kao i JEFF.

Kada se kreiraju izveštaji kao obični tekstualni fajlovi, nameću se neka ograničenja: slike se ne mogu prikazati a podaci mogu biti prikazani samo u improvizovanoj tabelarnoj formi. Međutim, u slučaju da postoji potreba da se pruži brz i jednostavan ali čitljiv izlaz (kao kod testiranja i debagovanja), običan tekst je najpogodniji. Što se tiče prezentacije, XML izveštaji imaju slična ograničenja. Međutim, ovaj format pruža značajnu fleksibilnost kada postoji potreba za prilagođavanjem izveštaja ili vršenjem kompleksnih transformacija (korišćenjem XSLT-a i CSS-a). Na primer, XML izveštaji sadrže nizove delova objašnjenja prezentovane kao XML elementi (i atributi), u redu u kojem su ovi delovi nastali. Korišćenjem XSLT-a, moguće je promeniti njihov redosled (npr. prikazati delove sa ERROR kontekstom prvo), ili ih filtrirati u prema nekim ključnim rečima (npr. prikazati samo one delove koji imaju pridružen “product” tag), i generalno, transformisati ih rezultujući različitim XML dokumentom. PDF izveštaji nemaju ograničenja kod prezentacije i mogu prikazati slike i tabele podataka ako je potrebno. Ovaj format je najkorisniji za izveštaje koji su namenjeni krajnjim korisnicima ali, s obzirom na to da je ovo binarni format, imaju najmanji transformacioni potencijal. Primeri izveštaja u sva tri formata se mogu videti u narednom poglavlju.

Konačno, JEFF je testiran korišćenjem JUnit alata [92], u skladu sa procedurama testiranja i preporukama datim ovde [93]. Svi testovi se mogu besplatno preuzeti zajedno sa JEFF-om sa iste internet adrese.

4.7 Primer korišćenja

Kao primer korišćenja JEFF-a, biće upotrebljen dobro poznati ES za savetovanje pri izboru vina. Ovaj ES pomaže korisnicima da izaberu vino uz obrok u slučaju kada nisu sigurni koju vrstu da kupe. Zaključivanje je bazirano na relativno malom broju ulaznih podataka: da li će vino biti konzumirano pre večere, posle večere, uz glavno jelo, dezert ili sir, koja vrsta glavnog jela/dezerta/sira će biti servirana, da li korisnici vole penušavo vino ili ne, kakva se flaša i boja svidja korisniku, itd. Izlaz je preporuka o

odgovarajućem generičkom tipu vina ili o konkretnom varijetetu.

Baza znanja za ovaj ekspertni sistem je preuzeta u vidu listinga koda [94] i delimično implementirana korišćenjem Drools Expert BRE kako bi se napravio primer. Dva od 28 implementiranih pravila su prikazana na sledećoj slici (Slika 33 - levo).

```
rule "Fruit-based dessert"
    lock-on-active true
    agenda-group  "dessert"
when
    wr: WineRequest (consumationTime == "to accompany dessert" &&
                     dessert == "fruit or primarily fruit")
then
    String[] wine = {"Riesling"};
    wr.setSuggestedVarietalWine(wine);

    update(wr);
end



rule "Sweet dessert"
    lock-on-active true
    agenda-group  "dessert"
when
    wr: WineRequest (consumationTime == "to accompany dessert" &&
                     dessert == "very sweet such as chocolate")
then
    wr.setRecommendedGenericWineType("port");

    update(wr);
end
```

```
rule "Fruit-based dessert"
    lock-on-active true
    agenda-group  "dessert"
when
    wr: WineRequest (consumationTime == "to accompany dessert" &&
                     dessert == "fruit or primarily fruit")
then
    String[] wine = {"Riesling"};
    wr.setSuggestedVarietalWine(wine);

    Object [] content = new Object[1];
    content[0] = "Riesling";
    wizard.addText("dessert", "Fruit-based dessert", null, content );
    update(wr);
end

rule "Sweet dessert"
    lock-on-active true
    agenda-group  "dessert"
when
    wr: WineRequest (consumationTime == "to accompany dessert" &&
                     dessert == "very sweet such as chocolate")
then
    wr.setRecommendedGenericWineType("Port");

    Object [] content = new Object[1];
    content[0] = "Port";
    wizard.addText( "dessert", "Sweet dessert", null, content );
    ImageData imageData = new ImageData("/images/port.jpg", "A bottle of Port");
    wizard.addImage("dessert", "Sweet dessert",null, imageData );
    update(wr);
end
```

Slika 33: Pravila u Drools Expert-u bez objašnjenja (levo) i sa objašnjenjem u JEFF-u (desno)

Pravila pružaju preporuke u slučaju da se vino servira sa dezertom. Ako se dezert sastoji isključivo ili primarno od voća, *Fruit-based dessert* pravilo se aktivira a, ako je dezert veoma sladak (npr. čokolada), *Sweet dessert* pravilo se aktivira. U prvom slučaju, ekspertni sistem predlaže Riesling, dok u drugom predlaže Port. Java objekat *wr* je instanca *WineRecommendation* klase i čuva inicijalne činjenice i zaključke kao vrednosti atributa: vreme konzumiranja, tip dezerta, predložene sorte vina, generički tip vina, itd. Novi zaključci se postavljaju u THEN delu pravila a radna memorija se ažurira *update* metodom. Takođe, treba imati na umu da je baza znanja podeljena u *grupe pravila* (agenda group) i da oba pravila pripadaju grupi pravila koja se odnose na dezert (*dessert*). Meta-pravila koja usmeravaju proves zaključivanja se izvršavaju na početku i aktiviraju odgovarajuće grupe pravila.

Ista dva pravila, ali sa dodatim objašnjenjima se mogu videti na slici (Slika 33 – desno). Zaključak koji je izveden kada se *Fruit-based dessert* pravilo aktivira je objašnjen tekstualnim delom objašnjenja. Prvo, sadržaj koja treba biti dinamički umetnut u statički

deo objašnjenja se inicijalizuje (Riesling). Potom, delovi objašnjenja se kreiraju i dodaju kao objašnjenje pozivanjem *addText* metode instance *JEFFWizard* klase. Ova instanca je inicijalizovana pre nego što je počeo proces zaključivanja i uvedena je u radnu memoriju kao globalna promenljiva, čineći je dostupnom svim pravilima. Parametri *addText* metode obuhvataju naziv grupe pravila (*dessert*), naziv pravila (*Fruit-based dessert*), tagove (null zbog toga što su izostavljeni), i sadržaj. Kontekst je *INFORMATIONAL*. Kada se *Sweet dessert* pravilo aktivira, dva dela će biti uneta u objašnjenje. Prvi je tekstualni deo i veoma je sličan prethodnom objašnjenju, s tim da je dinamički deo sadržaja „Port“ i da je naziv pravila *Sweet dessert*. Drugi deo je slika. Instanca *ImageData* klase sadrži URL slike („images/port.jpg“) i naslov slike („A bottle of Port“). Kako bi se kreirao ovaj deo, poziva se *addImage* metoda, a ulazni parametri su naziv grupe (*dessert*), naziv pravila (*Sweet dessert*), tagovi (izostavljeno) i sadržaj (instanca *ImageData*). Kontekst je takođe *INFORMATIONAL*.

Statička tekstualna objašnjenja se čuvaju u property fajlovima. Sa obzirom na to da ovaj primer treba da pruži objašnjenja na Engleskom (podrazumevani jezik) i Srpskom („srb“) koji se govori u Republici Srbiji („RS“), potrebna su dva skupa property fajlova, po jedan za svaki jezik. I pošto su pravila podeljena u grupe pravila, svaka grupa ima sebi pridružen property fajl a naziv grupe se koristi kao deo naziva fajla (videti sledeći listing).

 # sadržaj fajla “dessert_text.properties”
Fruit-based\ dessert = With any kind of fruit-based dessert, we recommend {0} as one of the best.
Sweet\ dessert = If it is sweet that you like, then a glass of {0} must accompany it.

 # sadržaj fajla “dessert_text_srb_RS.properties”
Fruit-based\ dessert = Sa bilo kojim desertom na bazi voća, mi preporučujemo {0} kao bolji izbor.
Sweet\ dessert = Ako slatke deserte volite, onda čaša {0} najviše prija.

Naziv prvog fajla je „dessert_text.properties” i on sadrži tekstualna objašnjenja za

dessert grupu pravila na podrazumevanog jeziku. Drugi fajl, „dessert_text_srb.RS.properties", sadrži tekstualna objašnjenja za ista pravila, ali prevedena na Srpski jezik. U oba fajla se nazivi pravila koriste kao ključevi (svim praznim znakovima u nazivu ključa prethodi obrnuta kosa crta), a tekstualna objašnjenja su vrednost. Treba napomenuti da će dinamički deo tekstualnih objašnjenja biti umetnut na poziciju gde se nalaze vitičaste zagrade ("{0}"). Redni broj između vitičastih zagrada označava redni broj dinamičke vrednosti iz niza unetih vrednosti (*content* parametar *addText* metode). U oba slučaja, postoji samo jedna dinamička vrednost i to je naziv vina (Riesling tj. "Port" u zavisnosti od izvršenog pravila).

Primer izveštaja u tekstualnom formatu (običan tekst) na Engleskom jeziku može se videti na sledećoj slici (Slika 34 – gore). On pokazuje objašnjenje KAKO (kako je zaključak dobijen), korak po korak, ali i STRATEGIJA objašnjenje. Ovaj izveštaj počinje prikazom datum kreiranja i imena vlasnika izveštaja, potom ide naslov ("Wine recommendation") praćen objašnjenjem. Prvi pasus prikazuje stratešku odluku da vodi proces zaključivanja isključivo ka dezertnim vinima. Ovo je postignuto korišćenjem meta-pravila. Drugi i treći pasus objašnjavaju konačnu preporuku. Kao što je naglašeno, slike se ne mogu prikazatu i tekstualnim fajlovima, tako da unosi samo njihov naslov i URL. Isti izveštaj, ali sa uključenim tragom izvršavanja pravila, može se videti na istoj slici (Slika 34 – dole). Ovaj izveštaj je namenjen testiranju i debagovanju, tako da prikazuje sve delove objašnjenja koji su uneti: kontekst, pravilo, grupu pravila i tagove.

```
Creation date: 7/16/10 9:26 AM
Report owner is: Bojan Tomic

Wine recommendation

Dessert wines are, in general, sweet to some extent and can be sparkly. The focus of
the search is on them.

If it is sweet that you like, then a glass of Port must accompany it

Caption is: A bottle of Port
The path to this image is: /images/port.jpg
```

```
Creation date: 7/16/10 9:30 AM
Report owner is: Bojan Tomic

Wine recommendation

The context is: strategic
The rule that initiated the creation of this chunk: Activate only rules for dessert wines
Dessert wines are, in general, sweet to some extent and can be sparkly. The focus of
the search is on them.

The context is: informational
The rule that initiated the creation of this chunk: Sweet dessert
The group to which the executed rule belongs: dessert
If it is sweet that you like, then a glass of Port must accompany it

The context is: informational
The rule that initiated the creation of this chunk: Sweet dessert
The group to which the executed rule belongs: dessert
Caption is: A bottle of Port
The path to this image is: /images/port.jpg
```

Slika 34: Izveštaj u tekstualnom formatu na Engleskom bez (gore) i sa (dole) tragom izvršavanja pravila

Ako je potrebno napraviti isti izveštaj na Srpskom jeziku, to bi izgledalo ovako (Slika 35). Naslov, sva tekstualna objašnjenja, naslovi slika i naslovi tabela podataka su prevedeni. Takođe, podaci o jeziku i zemlji su uključeni u izveštaj, sa obzirom na to da se podrazumevani jezik (u ovom slučaju Engleski) ne koristi.

```
Creation date: 7/16/10 10:15 AM
Report owner is: Bojan Tomic
The language used: srb
The country is: RS

Preporuceno vino

Dezertna vina su, uopsteno gledano, slatka a mogu biti i penusava. Fokus daljeg
zakljucivanja je na ovim vinima.

Ako slatke deserte volite, onda casa Port najvise prija

Caption is: Boca Porta
The path to this image is: /images/port.jpg
```

Slika 35: Izveštaj u tekstualnom formatu na Srpskom

Ako je traženi izlazni format XML, izveštaj će izgledati kao na sledećoj slici (Slika 36). Delovi objašnjenja, i njihov sadržaj su predstavljeni XML elementima, dok su svi drugi delovi podataka i podaci o generalnom objašnjenju predstavljeni XML atributima. Prema tome, koreni element je *explanation* i on predstavlja celo objašnjenje. Njegovi atributi su opšti podaci o objašnjenju: datum nastanka, vlasnik, jezik i zemlja. Njegovi podelementi su delovi objašnjenja i mogu, u zavisnosti od tipa, biti predstavljeni elemetima *textualExplanation*, *dataExplanation* ili *imageExplanation*. Svaki od ovih podelemenata ima, kao atribute, definisane naziv pravila, grupe pravila i kontekst.U korišćenom primeru nisu navedeni tagovi, ali da jesu, pojavili bi se kao posebni elementi u okviru svakog dela objašnjenja (*tags* element sa više tag podelemenata). Podelementi delova objašnjenja zavise od tipa dela, ali su, u svakom slučaju, ograničeni *content* elementom. U slučaju tekstualnog dela objašnjenja, *content* element sadrži samo tekst. U slučaju slike kao dela objašnjenja, *content* element sadrži element *imageUrl* koji kao vrednost ima URL slike, a kao atribut njen naslov (*caption*). U slučaju podataka kao delova objašnjenja, *content* element sadrži podelemente *SingleData*, *OneDimData*, *TwoDimData* i *ThreeDimData* u zavisnosti od dimenzionalnosti podataka. O ovome će biti više reči u narednih par strana.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<explanation date="7/16/10 9:59 AM" owner="Bojan Tomic" title="Wine recommendation">
    <textualExplanation rule="Activate only rules for dessert wines" context="strategic">
        <content>
            Dessert wines are, in general, sweet to some extent and can be sparkly. The focus of
            the search is on them.
        </content>
    </textualExplanation>
    <textualExplanation rule="Sweet dessert" group="dessert" context="informational">
        <content>
            If it is sweet that you like, then a glass of Port must accompany it
        </content>
    </textualExplanation>
    <imageExplanation rule="Sweet dessert" group="dessert" context="informational">
        <content>
            <imageUrl caption="A bottle of Port">/images/port.jpg</imageUrl>
        </content>
    </imageExplanation>
</explanation>
```

Slika 36: XML izveštaj

Konačno, primer običnog PDF izveštaja je prikazan u produžetku (Slika 37). U ovom formatu, slike i tabele podataka mogu biti pravilno prikazani. Potrebno je napomenuti i to da je ovaj PDF izveštaj onakav kakav će se prikazati bez korišćenja opcija za formatiranje izveštaja.

Wine recommendation

Dessert wines are, in general, sweet to some extent and can be sparkly. The focus of the search is on them.

If it is sweet that you like, then a glass of Port must accompany it

IMAGE: A bottle of Port

Slika 37: PDF izveštaj

Kao rezultat skorašnjeg razvoja [99], JEFF podržava i mogućnost formatiranja izgleda samog PDF izveštaja. To podrazumeva podešavanja strane (veličina strane, rotacija

strane, margine), podešavanja slova (font, boja veličina) ali i izbor tehnike prezentacije podataka. Naime, podrazumevana opcija za prikaz podataka je tabela, ali se oni sada mogu prikazati i putem nekoliko vrsta grafika. Sva podešavanja imaju neku podrazumevanu vrednost a željene izmene se unose u odgovarajući konfiguracioni property fajl. Delovi jednog takvog fajla se mogu videti na nekoliko sledećih slika (Slika 38, Slika 39 i Slika 40). Prvi deo se odnosi na formatiranje stranice, drugi na formatiranje tabela, a treći na formatiranje teksta objašnjenja i slika. Po potrebi, tekst objašnjenja se može formatirati u skladu sa kontekstom.

```
#------------------------------------------------------------
#PAGE PROPERTIES
#
#NOTE: you can specify page height and width manual or you
#      can use predefined formats through page format property

#specify height and width
;page_height=800
;page_width=600

#specify page format
#allowed values are: A0, A1, A2, A3, A4, A5, A6, A7, A8, A9, A10
page_format=A4

#specify page rotation
#allowed values are: true, false
page_rotation=true

#specify page margins
margin_left=10
margin_right=10
margin_top=10
margin_bottom=10

#specify margin mirror
#allowed values are: true, false
margin_mirror=true
```

Slika 38: Deo property fajla sa podešavanjima stranice [99]

```
#------------------------------------------------------------
#DATA PROPERTIES
#

#specify color for the table column value
#allowed values are: black, red, white, blue, yellow
table_column_value_color=red

#specify size for the table column value
table_column_value_size=12

#specify font for the table column value
#allowed values are: times roman, courier, helvetica
table_column_value_font=helvetica

#specify color for the table header background
#allowed values are: black, red, white, blue, yellow
table_header_background_color=yellow

#specify color for the table header value
#allowed values are: black, red, white, blue, yellow
table_header_value_color=red

#specify size for the table header value
table_header_value_size=10

#specify font for the table header value
#allowed values are: times roman, courier, helvetica
table_header_value_font=courier

#specify alignment for the table
#allowed values are: center, left, right
table_alignment=center

#specify space before and after the table
table_space_before=10
table_space_after=10
```

Slika 39: Deo property fajla sa podešavanjima za tabele [99]

```
#-----------------------------------------------------------
#TEXT PROPERTIES
#
#NOTE: to set special text format properties for specific context replace word defualt with context name
#      for example to use text_informational_color to specify color for the informational context or
#      use text_strategic_font to set the font for the strategic context

#specify color for the default text
#allowed values are: black, red, white, blue, yellow
text_default_color=blue

#specify size for the default text
text_default_size=12

#specify font for the default text
#allowed values are: times roman, courier, helvetica
text_default_font=helvetica

#-----------------------------------------------------------
#IMAGE PROPERTIES
#

#specify color for the image caption text
#allowed values are: black, red, white, blue, yellow
image_caption_color=red

#specify size for the image caption text
image_caption_size=10

#specify font for the image caption text
#allowed values are: times roman, courier, helvetica
image_caption_font=courier
```

Slika 40: Deo property fajla sa podešavanjima teksta i slika [99]

Kada se pokrene formiranje PDF izveštaja, rezultat je dokument koji je u skladu sa navedenom specifikacijom. Strana izveštaja je u željenoj veličini, orijentaciji i sa definisanim marginama, tekst je željene boje, fonta i veličine (Slika 41). Naslov slike je, takođe, željene boje, veličine i fonta (Slika 42).

CONTEXT: INFORMATIONAL
GROUP: dessert
RULE: Sweet dessert
Ako slatke deserte volite, onda casa Port najvise prija

Slika 41: Formatirani tekst [99]

Slika 42: Formatirana slika [99]

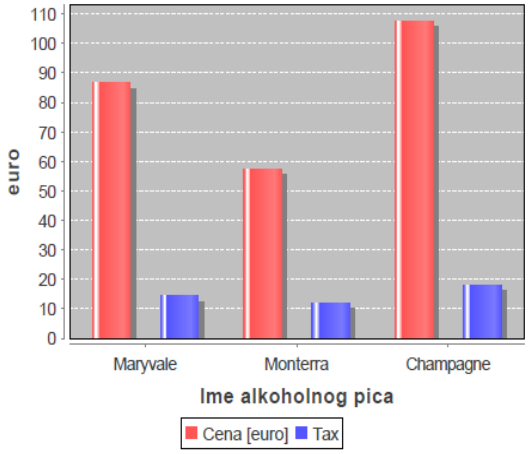
Međutim, zanimljivo je videti šta se dešava sa podacima i koje su mogućnosti njihovog prikaza. Naime, kada se primene opcije formatiranja tabela na konkretne podatke, dobija se tabela kao na sledećoj slici (Slika 43 - gore). Zaglavlje tabele je određene boje,

izgled teksta zaglavlja i tela tabele je u skladu sa navedenim opcijama. U ovom slučaju, u tabeli je prikazana instanca klase *ThreeDimData* koja sadrži trojke sa podacima o nazivu vina, ceni i porezu. Instance klase *Dimension* sadrže podatke koji se unose u zaglavlje tabele, dok se podaci iz tela tabele dobijaju preko liste instanci klase *Triple*. Redosled kolona je definisan redosledom definisanih dimenzija, a redosled redova u tabeli redosledom podataka unetih u listu. Kada je potrebno prikazati iste ove podatke u vidu grafika, celokupna porcedura se sastoji u tome da se u sam deo objašnjenja doda tag *chart*, a JEFF će, na osnovu toga, napraviti odgovarajući grafik (Slika 43 - dole).

CONTEXT: INFORMATIONAL
GROUP: after dinner
RULE: After dinner
TAGS:

Ime alkoholnog pica	Cena [euro]	Tax
Maryvale	86.50	14.50
Monterra	57.50	12
Champagne	107.50	18

CONTEXT: INFORMATIONAL
GROUP: after dinner
RULE: After dinner
TAGS: 'chart'



Slika 43: Primer podataka prikazanih u vidu tabele i grafika [99]

5 Prototip sistema za automatizovano tumačenje podataka o profitu

Već je navedeno da jedan od bitnih očekivanih rezultata ovog istraživanja i razvoj sistema (prototipa), kojim se u praksi demonstriraju svi do sada navedeni principi. Osnovna ideja je da ovaj prototip bude mala ali dobro osmišljena softverska aplikacija koja omogućava automatizovano tumačenje nekoliko međusobno povezanih KPI i drugih pokazatelja. Takođe, zamišljeno je da ovaj prototip bude isproban na nekom realnom problemu, ma kako jednostavnom.

Kao rezultat, napravljena je *Enterprise Profit Interpretation Tutor* (PT nadalje) aplikacija. Ova aplikacija tumači podatke o godišnjem profitu preduzeća i obezbeđuje, kao izlaz, automatski generisane izveštaje. U cilju pružanja detaljne analize profita PT, osim sadašnje i prošlih vrednosti profita, uzima u obzir i neke blisko povezane pokazatelje kao npr. ukupan prihod, ukupne troškove, period povraćaja investicije itd.

Upotreba aplikacija u poslovnom okruženju zahteva da one budu dovedene na visok nivo funkcionalnosti i efikasnosti, pa je odlučeno da se PT proba kao dodatno nastavno sredstvo na studijama menadžmenta. Svrha PT aplikacije je da omogući studentima da na konkretnim primerima nauče kako se tumači godišnji profit preduzeća. Naime, svaki student bi trebalo da osmisli konkretne primere poslovnih situacija unošenjem odgovarajućih podataka, da pokrene aplikaciju i da vidi koji su zaključci. Nakon toga, on tj. ona bi trebalo da uporedi svoje zaključke sa zaključcima koje je doneo PT i da uoči eventualna odstupanja. Svaki zaključak koji PT donese je, naravno, dat u obliku rečenica kontrolisanog jezika i prikazan zajedno sa podacima tako da je informacije dovoljno pročitati i nikakvo dalje tumačenje nije neophodno.

Evaluacija PT aplikacije je data u posebnom poglavlju i obuhvata rezultate ankete sprovedene među studentima nakon odslušanog kursa ekonomike preduzeća u okviru diplomskih studija menadžmenta.

5.1 Domensko znanje

Prvi korak u kreiranju PT aplikacije je bila analiza domenskog znanja. Rezultat je izbor odgovarajućih KPI i drugih pokazatelja čije vrednosti je potrebno obuhvatiti sistemom. Nakon toga se moglo pristupiti prikupljanju znanja potrebnog za tumačenje njihovih vrednosti. Izvori znanja su, u oba slučaja, bili univerzitetski udžbenici [102] [103][104], a konsultovan je i domenski ekspert. Rezultat je projektovana baza znanja koja je podeljena u nekoliko modula, a formiran je i redosled po kojem se ovi moduli aktiviraju. Konačno, isprojektovana su i objašnjenja za krajnje korisnike koja predstavljaju informacije prikazane rečenicama kontrolisanog jezika.

5.1.1 Izabrani pokazatelji

Cilj PT aplikacije je da omogući tumačenje godišnjih vrednosti profita za preduzeća. Zbog toga je bilo jako bitno da se napravi izbor odgovarajućih pokazatelja koji će biti uvršćeni u analizu profita. To su:

1. Profit (u apsolutnom i relativnom iznosu)
2. Ukupni prihod
3. Ukupni troškovi
4. Diferencijalni profit
5. Period povraćaja investicije
6. Starost preduzeća (u godinama)
7. Prosečan profit u privrednoj grani

Formula kojom se izražava *godišnji profit preduzeća* (P) je veoma jednostavna i predstavlja razliku između *ukupnog godišnjeg prihoda* (UP) i *ukupnih godišnjih troškova* (UT) preduzeća:

$$P = UP - UT$$

Ovako izračunat profit je iskazan u *apsolutnom iznosu* (novcu) a formula za izračunavanje *relativnog profita* (u procentima) je:

$$P_{\%} = P / UP * 100 = (UP - UT) / UP * 100$$

Ove dve formule čine osnovu za utvrđivanje koliko je preduzeće dobro poslovalo u toku jedne godine. Pri tome, profit je KPI jer preduzeća često postavljaju ciljne vrednosti koje je potrebno ostvariti. Na primer, godišnji profit od 5% je vrednost koja se smatra za odličan profit za preduzeće A, ali je ciljna vrednost sve preko 4%.

Pokazatelji koji su u najbližoj vezi sa profitom su *ukupan prihod* i *ukupni troškovi*. Ova dva pokazatelja direktno određuju vrednost profita pa ih je potrebno analizirati da bi se utvrdilo koji su uzroci povećanja, stagnacije ili smanjenja profita. Svaki od ova dva pokazatelja ima svoje pojedinačne činioce koji određuju njegovu vrednost, ali je, radi jednostavnosti, odlučeno da ovi činioci ne budu obuhvaćeni PT aplikacijom u ovom trenutku.

I *prethodne vrednosti profita* su uzete u obzir kao pokazatelji. Njihovo tumačenje zapravo predstavlja analizu vremenskih serija i može da ukaže na trendove u smislu kretanja profita ali i opšte poslovne slike. Analiza vremenske serije, u opštem smislu, predstavlja poređenje više uzastopnih vrednosti nekog pokazatelja, a vrši se izračunavanjem diferencijalnih vrednosti između dve uzastopne vrednosti profita. U smislu profita, u pitanju je *diferencijalni profit* (ΔP), a izračunava se u relativnom iznosu:

$\Delta P_{n\%} = P_{Tn} - P_{T(n-1)}$, pri čemu je u oba slučaja Tn n-ta godina, a T(n-1) godina pre nje.

Jedna vrednost diferencijalnog profita ukazuje na to da li se profit povećao, smanjio ili ostao isti u odnosu na prethodnu godinu. Međutim, na osnovu poređenja nekoliko vrednosti diferencijalnog profita, može se utvrditi da li je, u toku tih nekoliko godina, profit rastao, padao stagnirao ili oscilovao. Zbog toga je u analizi obuhvaćeno više vrednosti diferencijalnog profita.

Period povraćaja investicije (return on investment period) je ključni pokazatelj koji utiče na tumačenje vrednosti profita u slučaju da je preduzeće novo tj. tek osnovano. On označava vremenski period za koji se očekuje da će preduzeće povratiti sredstva koja su uložena u njega pri osnivanju tj. do kada se očekuje da će preduzeće početi da posluje pozitivno. Period povraćaja investicije se uvek izražava u godinama a, kada su

preduzeća u pitanju, računa se po sledećoj formuli:

$t_{god} = I / NP$, pri čemu je t rok vraćanja u godinama, I su uložena sredstva a NP je godišnji neto prihod

Kao što se može videti iz prethodne formule, da bi se znalo da li je periodom povraćaja investicije istekao, potrebno je znati i starost preduzeća odnosno koliko godina preduzeće posluje. Stoga je i ovaj pokazatelj uzet u razmatranje.

Konačno, tumačenje profita je u vezi i sa nekim spoljnim pokazateljima. U ovom slučaju, uzet je u obzir *prosečan profit u privrednoj grani* jer je to jedna gruba ali dostupna i prilično efikasna mera kojom se preduzeće poredi sa konkurentima u grani (kada je profit u pitanju). Prosečan profit u grani se uvek izražava u apsolutnom iznosu a računa se prema sledećoj formuli:

$P_{pg} = \Sigma P_{pi} / N$ (i = 1..N), pri čemu je N broj preduzeća u grani, a P_{pi} profit i-tog preduzeća.

5.1.2 Postupak prikupljanja znanja

Nakon što su izabrani odgovarajući pokazatelji, prešlo se na prikupljanje znanja. Cilj je bio da se utvrdi tačno kako se koji pokazatelj (tj. njegova vrednost) tumači ali, još važnije, kako vrednosti drugih pokazatelja utiču jedni na druge i koji je opšti redosled u kojem se ovi pokazatelji tumače. Znanje je, kao što je već napomenuto, prikupljeno iz dva izvora: univerzitetske udžbeničke literature i od domenskog eksperta. Domenski ekspert je, u ovom slučaju, bio magistar organizacionih nauka smera za menadžment koji je nekoliko godina radio na sličnim poslovima u privredi, ali sada predaje ekonomiku preduzeća. Ceo postupak se odvijao u nekoliko osnovnih faza:

1. Utvrđivanje opšteg redosleda u kojem se (izabrani) pokazatelji tumače
2. Prikupljanje znanja potrebnog za tumačenje pojedinačnih vrednosti pokazatelja
3. Prikupljanje znanja potrebnog za tumačenje uticaja vrednosti pokazatelja jednih na druge
4. Prikupljanje znanja potrebnog za formiranje objašnjenja

Utvrđivanje opšteg redosleda u kojem se (izabrani) pokazatelji tumače je započeto tako što je napravljeno nekoliko konkretnih primera poslovne slike preduzeća sa vrednostima izabranih pokazatelja, a domenskom ekspertu dat zadatak da protumači rezultate poslovanja. Primeri su se odnosili na neke uobičajene poslovne situacije, ali i na neke ekstremne situacije koje su retke, ali se dešavaju. Ovi primeri su kasnije korišćeni i u drugim fazama prikupljanja znanja. Sve vreme u toku rešavanja problema, od domenskog eksperta se tražilo da navede šta će da uradi sledeće tj. koji pokazatelj analizira i zašto. Pošto su, zbog dugogodišnjeg iskustva, eksperti skloni tome da "zaborave" kako zapravo postupno rešavaju neki problem, istovremeno je praćeno i šta on zaista radi i koje grafike tj. tabele posmatra. Razlog zašto je sve ovako urađeno je taj što se smatralo da je najbolje da PT prati prirodni redosled kojim domenski ekspert rešava problem. Rezultat je traženi redosled tumačenja pokazatelja.

Prikupljanje znanja potrebnog za tumačenje pojedinačnih vrednosti pokazatelja je bila druga faza. Osnova za ovo znanje je uzeta iz literature, a onda je domenski ekspert dato znanje proverio, dopunio i izmenio po potrebi. Važno je napomenuti da je sve ovo rađeno tako da se još uvek ne uzimaju u obzir uticaji vrednosti drugih pokazatelja, već (teorijska) situacija da pred sobom ima samo vrednost analiziranog pokazatelja i ništa više. Znanje koje je prikupljeno u ovoj fazi se najviše odnosilo na utvrđivanje nekih osnovnih graničnih, ciljnih ali i ekstremnih vrednosti svakog pokazatelja pojedinačno, a prethodno spremljeni primeri su pomogli u identifikovanju ovih vrednosti.

Prikupljanje znanja potrebnog za tumačenje uticaja vrednosti pokazatelja jednih na druge je bila poslednja faza. Ovde se, zapravo, razmatralo kako određene vrednosti jednog pokazatelja menjaju način na koji se tumače neki drugi pokazatelji. I ovde je, kao osnova, poslužila literatura ali je domenski ekspert imao mnogo veću ulogu. Naime, veliki deo ovog znanja čine *iskustvena* (heuristička) pravila koja se uglavnom ne mogu naći u literaturi, jer su stečena praktičnim radom i nisu standardizovana. Manji deo ovih pravila se mogao naći u okviru studija slučajeva objašnjenih u udžbenicima. Ova faza je ujedno bila i najteža jer se zahtevalo da se ispitaju sve moguće kombinacije međusobnih uticaja dva ili više pokazatelja. Problem "kombinatorne eksplozije" je izbegnut zahvaljujući usmeravanju izvođenja ove faze od strane eksperta ka onim pokazateljima

koji zaista utiču jedni na druge, ali i tome da je broj izabranih pokazatelja relativno mali.

Na kraju, potrebno je napomenuti i to da je **prikupljeno znanje potrebno za formiranje objašnjenja** KAKO tj. reprezentacije samih informacija. Sama objašnjenja prate način na koji bi ekspert objasnio svoje rezonovanje.

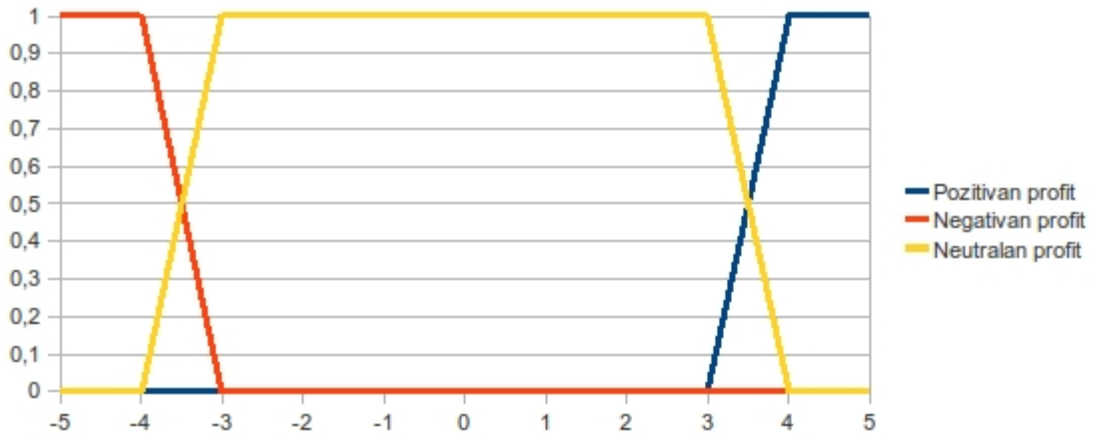
5.1.3 Prikupljeno znanje (baza znanja)

Redosled tumačenja izabranih pokazatelja koji je rezultat prikupljanja znanja se može grubo podeliti na pet faza koje oslikavaju prirodan redosled kojim bi ekspert uradio isti zadatak. Osnovni princip koji se može uočiti je taj da ekspert prvo pregleda veoma mali broj opštih pokazatelja da bi stekao uopšten utisak o poslovanju, pa tek onda analizira ostale pokazatelje da bi utvrdio uzrok takvog stanja. Prema tome, dobijena baza znanja sadrži pet modula koji se aktiviraju jedan za drugim u pet opštih koraka:

1. Tumačenje profita za trenutnu godinu
2. Tumačenje profita u kombinaciji sa ROI periodom
3. Tumačenje profita u kombinaciji sa prosečnim profitom u grani privrede
4. Tumačenje vremenske serije profita
5. Tumačenje profita u kombinaciji sa ukupnim prihodom i ukupnim troškovima

Tumačenje profita za trenutnu godinu je prvi korak i podrazumeva da se u obzir ne uzima nijedan podatak osim vrednosti profita za trenutnu godinu (i u apsolutnom i u relativnom iznosu) i postavljenih ciljnih vrednosti. Dakle, ne razmatraju se čak ni prošle vrednosti profita. Jedina informacija koja se može izvesti u ovom koraku je da li preduzeće ostvaruje profit ili ne i da li je ciljna vrednost profita postignuta. Sa obzirom na to da cu ove ciljne (referentne) vrednosti drugačije za svako preduzeće i da se menjaju u toku vremena, odlučeno je da se predstave putem nekoliko fuzzy kategorija koje su napravljene nad skalom relativnog profita (Slika 44): *pozitivan* (positive), *neutralan* (neutral) i *negativan* (negative) profit. Sve tri kategorije su predstavljene trapezoidnim fuzzy skupovima, a konkretne granične vrednosti su uzete na osnovu

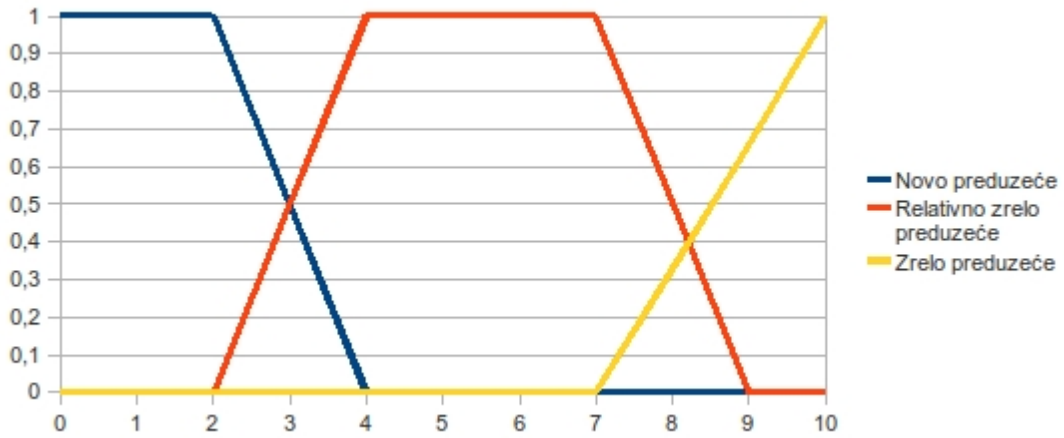
preporuke domenskog eksperta. Pod pozitivnim profitom se smatra onaj koji je dostigao ciljnu vrednost (u ovom slučaju to je četiri posto), pod neutralnim onaj koji je oko nula posto a pod negativnim onaj koji je niži od minus četiri posto. Tako, smatra se da je profit od četiri posto pozitivan sa vrednošću funkcije pripadnosti jednakoj jedan, negativan sa vrednošću funkcije pripadnosti jednakoj nula i neutralan sa vrednošću funkcije pripadnosti jednakoj nula. Profit od 3,5 posto se, recimo, smatra pozitivnim sa vrednošću funkcije pripadnosti od 0,5 ali isto tako i neutralnim sa vrednošću funkcije pripadnosti od 0,5 i negativnim sa vrednošću finkcije pripadnosti jednakoj nula. Na ovaj način se modeluju one situacije u kojima npr. profit jeste veći od nule, ali ipak nije baš da je dostigao ciljnu vrednost. Naravno, granične vrednosti ovih skupova tj. tačke trapeza koje ih definišu se mogu menjati po potrebi a da ostatak baze znanja ostane neizmenjen.



Slika 44: Fuzzy kategorije profita

Drugi korak je **tumačenje profita u kombinaciji sa ROI periodom**. Kada preduzeće ostvaruje pozitivan profit, uvek se smatra da je to dobro. Međutim, negativan ili neutralan profit ne mora uvek da bude protumačen kao loša stvar. To se dešava u onim situacijama kada je preduzeće tek krenulo sa radom tj. kad još uvek vraća uložena investiciona sredstva i kad projektovani ROI period nije istekao. Situacija postaje loša kada taj period istekne, a preduzeće još uvek nije počelo pozitivno da posluje. Da bi se pravilno opisala ova situacija, uvedene su tri fuzzy kategorije: *novo preduzeće* (new enterprise), *relativno zrelo preduzeće* (relatively established enterprise) i *zrelo*

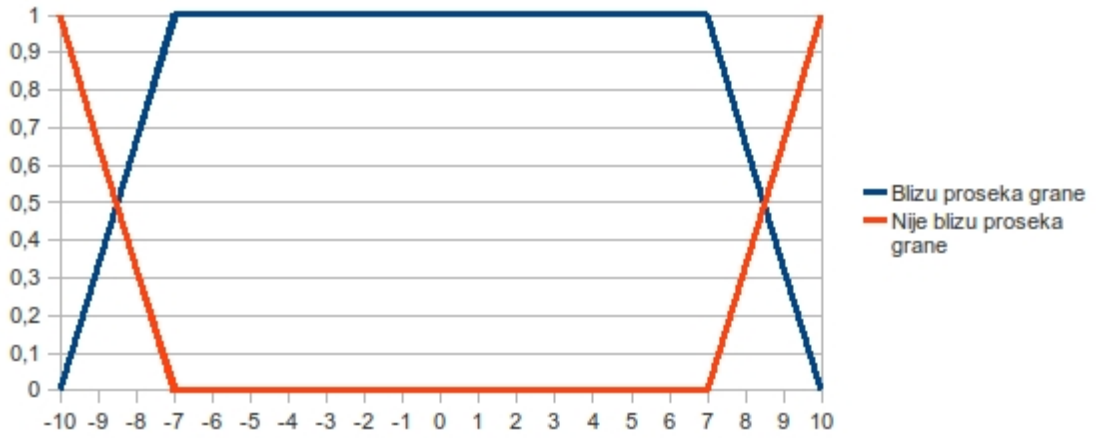
preduzeće (established enterprise). Sve tri kategorije su zasnovane na starosti preduzeća u godinama i izražene trapezoidnim fuzzy skupovima (Slika 45). Kada se profit (odnosno njegove kategorije iz prethodnog koraka) tumači u kombinaciji sa ROI periodom i starošću preduzeća, to onda izgleda ovako. Zrela preduzeća moraju da ostvaruju pozitivan profit i ROI period se uopšte ne razmatra. Kod novih preduzeća, negativan i neutralan profit se toleriše ako ROI period nije istekao. Ako jeste, onda se očekuje da preduzeće ostvari makar neutralan profit. Relativno zrela preduzeća mogu da ostvaruju neutralan i negativan profit ako ROI period nije istekao, ali nakon toga moraju da ostvaruju pozitivan profit.



Slika 45: Fuzzy kategorije starosti preduzeća

U trećem koraku se **tumači profit u kombinaciji sa prosečnim profitom u grani privrede**. Prosečne vrednosti nisu baš najbolji pokazatelji ako se koriste kao jedino merilo jer prikazuju prosek, a pojedinačne vrednosti koje ih čine mogu da veoma mnogo variraju. Međutim, neki važni zaključci ipak mogu da se donesu ako se profit preduzeća uporedi sa prosečnim profitom u grani. Opšta premisa je da, ako je profit preduzeća veći od prosečnog profita u grani privrede, to znači da preduzeće posluje bolje od konkurenata (u proseku) i obrnuto. Takođe, prosečan profit u grani privrede koji je blizu nule ili manji od nule može da ukazuje na neki ozbiljan poremećaj tržišta i da objasni uzrok negativnog poslovanja preduzeća. Prema tome, uvedene su još dve fuzzy kategorije koje služe da opišu da li je profit preduzeća *blizu proseka grane* (near average) ili nije *blizu proseka grane* (not near average). Skupovi koji opisuju ove dve

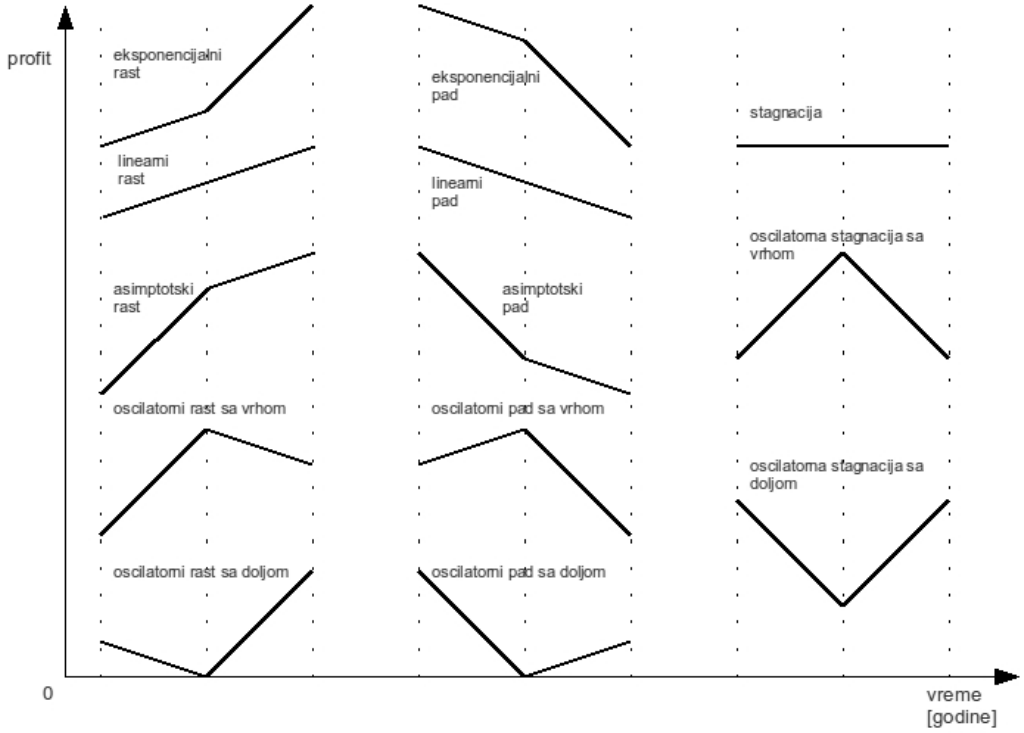
kategorije se mogu videti na sledećij slici (Slika 46). Razlika između profita i prosečnog profita u grani izražena u procentima, zapravo, predstavlja osu na kojoj su definisani dati fuzzy skupovi.



Slika 46: Fuzzy kategorije blizine profita proseku u grani

Četvrti korak je **tumačenje vremenske serije profita**. U pitanju je tumačenje tekuće vrednosti profita u kombinaciji sa prošlim vrednostima profita. Izvedeni pokazatelj koji se koristi u ove svrhe je diferencijalni profit kojim se porede vrednosti profita za dve uzastopne godine. Tumačenje vremenskih serija ima za cilj da utvrdi neke opšte trendove tj. kretanja posmatranog pokazatelja i često je ograničena na neki vremenski period. U ovom slučaju, domenski ekspert je obrazložio da je najsvrsishodnije posmatrati profit za tri uzastopne godine. Razlog je taj da se uslovi poslovanja a i ciljevi mogu drastično promeniti ukoliko je posmatrani period duži, pa poređenje rezultata npr. prve i poslednje godine nije merodavno.

Osnovni princip od kojeg se polazi je taj da je potrebno prvo utvrditi koji je opšti trend kretanja (rast, stagnacija ili opadanje), a onda i koja je podvrsta trenda u pitanju (eksponencijalni, linearni, asimptotski ili oscilujući). Pošto su podaci vremenske serije profita diskretni i, u ovom slučaju, uvek sadrže samo tri vrednosti, sve kombinacije trenda kretanja i podvrste trenda se mogu videti na sledećoj slici (Slika 47) i ima ih ukupno trinaest.

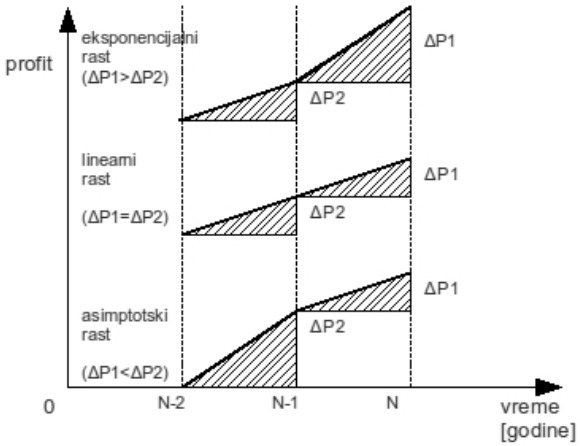


Slika 47: Trendovi i podtrendovi vremenskih serija profita

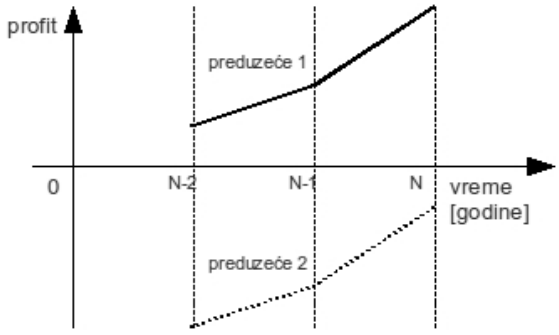
Prvo, rast profita može biti eksponencijalan (ubrzan), linearan (ravnomeran), asimptotski (usporen) ili oscilujući. Oscilujući rast je karakterističan po tome što se između najranije i poslednje vrednosti profita može uočiti rast, ali je srednja vrednost profita bila ili viša ili niža od ove dve. U prvom slučaju, oscilacija je "naviše", pa postoji "vrh" a u drugom slučaju oscilacija je "naniže" pa postoji "dolja". Slična je situacija i sa padom profita koji može biti eksponencijalni, linearan, asimptotski, usporen ili oscilujući. Oscilujući pad profita karakteriše to da je najranija vrednost profita veća od poslednje ali da je srednja vrednost bila viša ("vrh") ili niža ("dolja") od ove dve. Konačno, postoji i mogućnost da je profit stagnirao. U tom slučaju, stagnacija može biti obična (horizontalna linija na grafiku) ali i oscilujuća (sa "vrhom" ili "doljom"). Potrebno je napomenuti i to da utvrđivanje trenda nije domenski zavisna aktivnost, pa se ovaj deo baze znanja može upotrebiti i u svim onim situacijama kada je potrebno utvrditi trend bilo kojeg pokazatelja predstavljen preko tri diskretne vrednosti.

Utvrđivanje konkretnog trenda i njegovog podtrenda se vrši putem provere vrednosti diferencijalnog profita (Slika 48). Tako se, na primer, rast utvrđuje tako što su sve

vrednosti diferencijalnog profita (između svake dve uzastopne godine) veći od nule. Ako profit opada onda su sve vrednosti diferencijalnog profita negativne, ako stagnira onda su blizu nule, a ako je u pitanju i neka oscilacija, vrednosti su negativne i pozitivne naizmenično. Dalje, eksponencijalni rast se utvrđuje tako što, osim što je svaka vredost diferencijalnog profita veća od nule, važi i to da je svaka naredna vrednost diferencijalnog profita veća od prethodne. Linearni rast karakterišu jednake vrednosti diferencijalnog profita koje su, istovremeno, sve pozitivne itd. Za svaki od uočenih trinaest trendova je napravljena odgovarajuća fuzzy kategorija, a napravljen je i skup fuzzy pravila kojima se dolazi do zaključka o najverovatnijem trendu. To je urađeno na ovaj način jer je potrebno da za svaki trend postoji određena vrsta tolerancije. Tako je, na primer, veoma malo verovatno da će da postoji apsolutna stagnacija gde se vrednost profita nije promenila nimalo ili da će se pojaviti apsolutno linearni rast gde je rast profita potpuno ravnomeran. Tada se, iz praktičnih razloga, može reći da se sve vrednosti diferencijalnog profita do npr. 0.5% smatraju za stagnaciju. Ovako se, zapravo, najpribližnije modeluje situacija u kojoj domenski ekspert gleda vremensku seriju i procenjuje trend.



Slika 48: Trendovi rasta profita i diferencijalni profit



Slika 49: Primer eksponencijalnog trenda

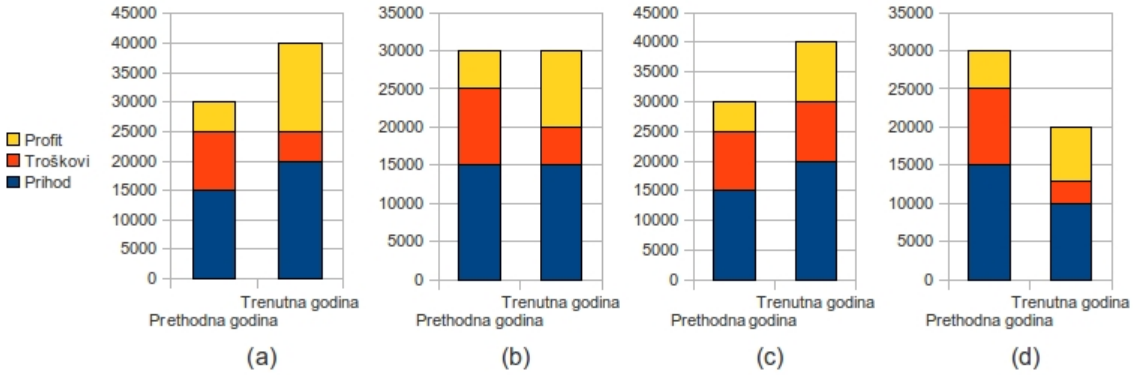
Kada se utvrdi trend i podtrend, još jedna stvar se uzima u obzir pri tumačenju vremenske serije profita. U pitanju je provera da li su pojedinačne vrednosti profita iz vremenske serije veće od nule, manje od nule ili jednake nuli. Na primer, eksponencijalni rast može da bude takav da su sve vrednosti veće od nule i to se tumači kao odličan rezultat poslovanja (Slika 49 - vremenska serija za preduzeće 1). Međutim, eksponencijalni rast može da se pojavi i ako su sve tri vrednosti profita manje od nule (Slika 49 - vremenska serija za preduzeće 2). U tom slučaju, to je loš rezultat poslovanja jer preduzeće posluje sa gubitkom, bez obzira na to što je taj gubitak sve manji tj. što profit ubrzano teži nuli.

Poslednji korak je **tumačenje profita u kombinaciji sa ukupnim prihodom i ukupnim troškovima**. Prihod i troškovi direktno određuju vrednost profita jer se on dobija kao njihova razlika. U ovom koraku se, zapravo, utvrđuje zašto je profit porastao, stagnirao ili se smanjio u odnosu na prošlogodišnji i to se ostvaruje poređenjem vrednosti prihoda i troškova u odnosu na prošlogodišnje.

Rast profita može biti uzrokovan rastom prihoda i smanjenjem troškova u odnosu na prethodnu godinu (Slika 50 - a). Ovo je optimalna situacija jer se zaključuje se da se zbog rasta prihoda povećala *efektivnost* (sposobnost preduzeća da plasira svoje usluge tj. proizvode na tržištu) a da se, zbog pada troškova, povećala i *efikasnost* preduzeća (sposobnost preduzeća da sa što manje ostvarenih troškova pruži te usluge tj. napravi proizvode). Međutim, rast profita može da se pojavi i u slučaju da:

- Prihod stagnira a da su se troškovi smanjili (Slika 50 - b)
- Prihod raste a da troškovi stagniraju (Slika 50 - c)

- Prihod opada a i troškovi opadaju samo još brže (Slika 50 - d)

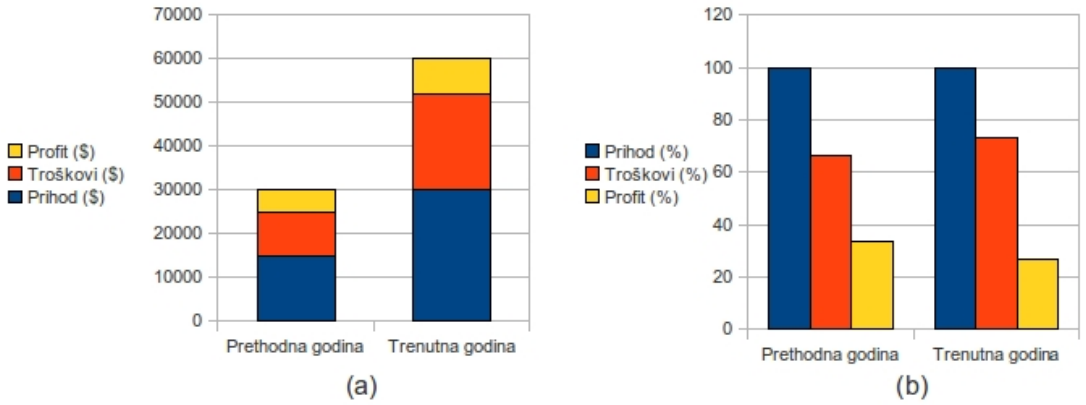


Slika 50: Profit, prihodi i troškovi

U prvom slučaju, preduzeće je samo postalo efikasnije (smanjilo je svoje troškove) a zadržalo je isti nivo efektivnosti - ukupan iznos prodatih proizvoda ili pruženih usluga. U drugom slučaju, preduzeće je efektivnije (npr. osvojilo je neko novo tržište) ali nema promene u smislu efikasnosti. U poslednjem slučaju smanjila se efektivnost (preduzeće prodaje manje) ali je efikasnost puno porasla pa je kompenzovala smanjene prihode, a efekat je uvećan profit. Ova situacija se, bez obzira na porast profita, posmatra kao upozorenje jer najavljuje potencijalnu opasnost od daljeg smanjenja efektivnosti. Slična stvar se dešava i kada profit stagnira ili opada pa se tumačenje i ovde svodi na utvrđivanje uzroka u smislu povećanja ili smanjenja efikasnosti tj. efektivnosti. Fuzzy kategorije koje su uvedene zbog poređenja profita, prihoda i troškova se svode na to da utvrde da li je taj pokazatelj *veći*, *manji* ili *približno jednak* u odnosu na prošlu godinu.

Osim efektivnosti i efikasnosti, u ovom koraku se prati kretanje i apsolutne i relativne vrednosti profita u odnosu na prethodnu godinu kao i uzrok eventualnog neslaganja trenda kretanja ova dva pokazatelja. Naime, moguće je da profit poraste u apsolutnom iznosu (novcu), ali da zapravo opadne u relativnom iznosu (procentima). Primer ovakve situacije je kad preduzeće ostvaruje veći nivo i prihoda i troškova u odnosu na prošlu godinu (pa je apsolutna razlika svakako veća, Slika 51 - a) ali su troškovi porasli većom stopom nego prihod (pa je relativna razlika zapravo manja, Slika 51 - b). Ovakve situacije nisu retke, a potrebno ih je uočiti jer mogu da otežaju uočavanje određenih pojava. Ako profit u relativnom iznosu poraste, a u apsolutnom se smanji, to najčešće

znači da preduzeće posluje efikasnije ali u manjem obimu nego pre što, konačno, vodi smanjenju učešća na tržištu. Ako bi se samo posmatrala relativna vrednost profita, izgledalo bi da je situacija bolja nego što jeste.



Slika 51: Apsolutni i relativni iznos profita

5.1.4 Objašnjenja za krajnje korisnike

Poslednji rezultat prikupljanja znanja su objašnjenja KAKO se došlo do određenih zaključaka. Ova objašnjenja su projektovana u skladu sa tim kako bi domenski ekspert objasnio svoje rezonovanje i u potpunosti prate proces zaključivanja iz prethodnog poglavlja. Pri tome, dodatno je obraćena pažnja na to da objašnjenja:

- Budu izražena jasnim i preciznim rečenicama.
- Budu takva da je lako uočljiv njihov smisao.
- Budu prilagođena nivou znanja krajnjeg korisnika.
- Sadrže (na odgovarajućim mestima) dinamički umetnute podatke koji dopunjavaju objašnjenje.
- Navode zaključak pa tek onda objašnjenje kako se došlo do tog zaključka.
- Budu takva da se može napraviti siže najvažnijih zaključaka.

Da bi objašnjenja predstavljala informacije na odgovarajući način, velika pažnja je bila posvećena tome da **budu izražena jasnim i preciznim rečenicama**. Cilj je bio da informacije budu shvatljive odmah po čitanju i da nikakva dodatna objašnjenja ne budu potrebna. Pri tome su naročito izbegavane dugačke rečenice pune stilskih izraza pa su,

kao rezultat, dobijena objašnjenja koja liče na rečenice u *novinarskom stilu pisanja* - prvo kratak i jezgrovit opis sa istaknutim ključnim delovima, a onda razrada datog opisa. Primer jednog objašnjenja je dat u nastavku:

Profit is SIGNIFICANTLY LARGE.
The enterprise is making money and profit is SIGNIFICANTLY LARGE which is GOOD.

Pri tome, vodilo se računa i o tome da objašnjenja budu takva da se **lako može uočiti njihov smisao**. U tu svrhu, ključne reči i izrazi su napisani velikim slovima, a sva objašnjenja su obojena odgovarajućim bojama i napisana odgovarajućom veličinom slova u zavisnosti od konteksta. Svi pozitivni zaključci su obojeni zelenom bojom, svi negativni crvenom, dok su upozorenja napisana žutom bojom. Ako je neki zaključak okarakterisan od strane sistema kao veoma pozitivan ili veoma negativan, onda su upotrebljena i veća slova. Primer nekoliko objašnjenja i načina njihovog prikaza se može videti u produžetku:

Profit is SIGNIFICANTLY LARGE. POZITIVAN

Profit is NEUTRAL (close to zero). UPOZORENJE

Profit is NEGATIVE!!! NEGATIVAN

Već je napomenuto da objašnjenja moraju da **budu prilagođena krajnjem korisniku** da bi efekat bio pozitivan tj. da bi korisnik izgradio poverenje u zaključke koje je sistem doneo. U ovom slučaju, PT ima studente kao krajnje korisnike pa su projektovana objašnjenja detaljna i objašnjavaju i osnovne stvari i zaključke koji bi iskusnijim korisnicima (npr. menadžerima) bili suvišni. Prema tome, kada bi PT bio namenjen menadžerima, bilo bi potrebno napraviti složenija (stručnija) objašnjenja sa, najverovatnije, mnogo manje detalja a više stručnih reči i izraza.

Da bi zaključak mogao da se proveri, bitno je da objašnjenje **sadrži dinamički umetnute podatke** koji se odnose na dati zaključak i dopunjavaju objašnjenje. Zamišljeno je da se ovo umetanje vrši dvojako: u okviru samog teksta objašnjenja, ali i putem grafika i tabela sa podacima koji bi bili prikazani uz sam tekst. Primer dela

objašnjenja sa vrednostima umenutim u tekst je dat u nastavku (vrednosti 100.000$ i 6.78 su umetnute):

Current profit is 100.000$ or, when expressed relatively 6.78%.

Prateći osnovnu ideju formiranja objašnjenja u novinarskom stilu pisanja, ona su projektovana tako da **navode zaključak pa tek onda objašnjenje kako se došlo do tog zaključka**. To se može videti iz sledećeg primer u kojem se u prvoj rečenici navodi osnovni zaključak, a potom sledi njegovo objašnjenje. Ceo zaključak predstavlja upozorenje pa je tekst žute boje:

Current profit is SIGNIFICANTLY LARGER than last years profit BUT ONLY IN PERCENTAGES.

When comparing absolute profit values, profit has DECREASED since last year. Both total income and total cost have decreased since last year, but the costs decreased a lot more than income thus creating an illusion that profit is on the rise (when viewed in percentages). The enterprise has reduced costs (it is now more efficient) but also lost some of its income. The situation must be viewed as a WARNING, and close monitoring is advised because income can become even smaller in the future.

Konačno, uočeno je da domenski ekspert na kraju zaključivanja posmatra celu poslovnu situaciju tako da izvodi **sižе najvažnijih zaključaka**. Zbog toga su objašnjenja projektovana u dve verzije: kratkoj i dugačkoj. Kratka verzija sadrži samo ono što je najvažnije i može se predstaviti na početku (ili kraju) izveštaja. Razlog za ovaj postupak ima i praktičnu prirodu: ovako formiran izveštaj na početku sadrži najvažnije stavke potrebne za upoznavanje sa poslovnom situacijom, a njegovim detaljnim čitanjem se mogu otkriti uzroci ovakvom stanju. Primer kratek i dugačke verzije objašnjenja je dat u produžetku:

Current profit is NEGATIVE (although close to zero), and is SIGNIFICANTLY LARGER than last years profit BUT ONLY IN PERCENTAGES. When comparing absolute profit values, profit has DECREASED since last year.

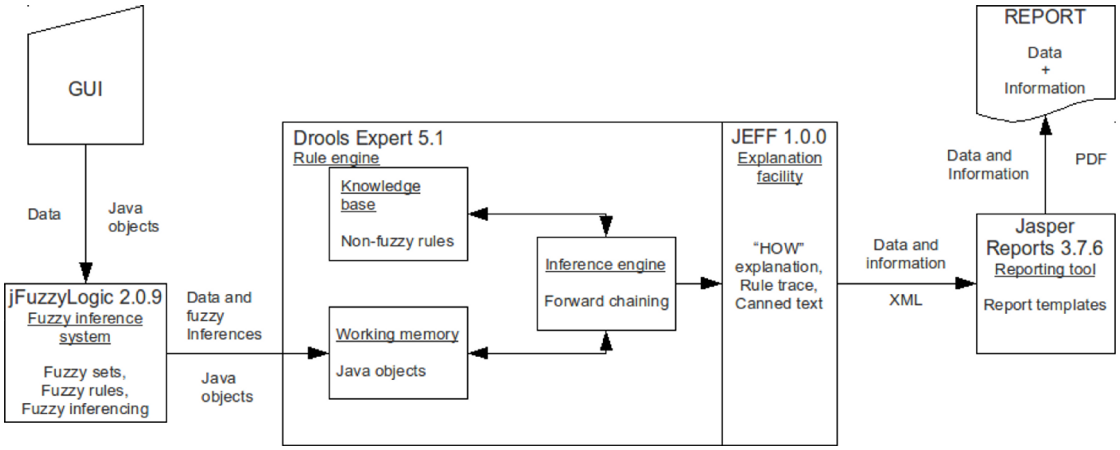
Current profit is SIGNIFICANTLY LARGER than last years profit BUT ONLY IN

PERCENTAGES. When absolute values are concerned, profit is still NEGATIVE (although close to zero) which is MOST ALARMING!!! The reason why it has increased is that both total income and total cost have decreased since last year, but cost has decreased more than income. However, total income is still smaller than total cost and enterprise is still loosing money!!!

5.2 Implementacija

Nekoliko faktora je moralo biti uzeto u obzir prilikom implementacije PT. Prvo, pošto je zamišljeno da PT bude upotrebljen kao nastavno sredstvo, sama distribucija, instalacija i upotreba alata su morali da budu jednostavni. Razlog je taj što bi se, u suprotnom, moglo izgubiti interesovanje studenata a sama funkcionalnost aplikacije bi bila u drugom planu zbog tehničkih problema. Drugo, studenti koriste veliki broj različitih operativnih sistema na svojim računarima, pa je i interoperabilnost bila veoma važna. Konačno, veliki broj softverskih alata koji su besplatno dostupni za upotrebu i koje već koristimo za razvoj sličnih sistema su napisani na programskom jeziku Java.

Prvobitno je zamišljeno da PT bude implementiran kao web aplikacija tj. da studenti mogu da mu pristupaju putem browser-a. Prednosti ovog pristupa su očigledne: nije potrebno distribuirati aplikaciju niti je instalirati na računar i ne postoji problem interoperabilnosti. Međutim, pojavili su se drugi problemi tehničke prirode koji su onemogućili primenu ove zamisli. Naime, samo zaključivanje PT je veoma procesorski zahtevan proces i bilo bi potrebno da serveri na kojima se PT nalazi budu izuzetno velike procesorske moći da bi se mogao podržati istovremeni pristup bar dve stotine studenata. Nažalost, resursi koji su mogli biti ustupljeni na korišćenje od strane fakulteta nisu bili dovoljno hardverski snažni da omoguće ovo.



Slika 52: Arhitektura PT i softverski alati korišćeni u implementaciji

Posle dodatnog razmatranja, odlučeno je da PT ipak bude implementiran kao desktop aplikacija u programskom jeziku Java i to korišćenjem više besplatnih softverskih alata napisanih u Javi (Slika 52). Iako instalacija i distribucija moraju da se izvrše, ovaj način implementacije ipak prevazilazi problem interoperabilnosti jer Java programi mogu da se izvršavaju na bilo kojem poznatijem operativnom sistemu. Osim toga, ceo alat se distribuira u vidu jednog izvršivog fajla (JAR - Java ARchive) koji je, posle kopiranja, dovoljno samo pokrenuti. Jedini preduslov je da na računaru postoji instalirana *JVM* (Java Virtuelna Mašina). Konačno, pošto računar pokreće samo jednu kopiju programa, utiče se i na celokupni utisak o brzini aplikacije.

Prva stvar koja se može uočiti je da ne postoji DW ili OLAP iz kojeg se preuzimaju podaci. Podaci se unose preko veoma jednostavnog grafičkog interfejsa (GUI - Graphical User Interface) i skladište u vidu običnih Java objekata. Očigledno je da su, na ovaj način, stvari pojednostavljene jer nema potrebe za instalacijom, konfiguracijom i pokretanjem složenog DW ili OLAP softvera. Podaci koje PT koristi za zaključivanje se sastoje od samo jedanaest vrednosti (i još šest drugih koje se automatski izvode), pa je unos podataka isključivo preko GUI-ja opravdan. U trenutku kad se podaci unesu, kreira se objekat klase *Enterprise* i puni se odgovarajućim podacima. Dijagram na kojem se vidi ova klasa i njeni elementi je dat na slici (Slika 53).

Slika 53: Klasa Enterprise

Klasa *Enterprise* sadrži veliki broj atributa ali i *getXXX* i *setXXX* metoda koje služe za preuzimanje tj. postavljanje vrednosti tih atributa. Takođe, može se uočiti da, osim atributa koji predstavljaju ulazne podatke (npr. *profit*, *cost*, *income*), sadrži i atribute koji se odnose na zaključke (*profitTrendCategory*, *currentProfitCategory* itd.). Drugim rečima, klasa *Enterprise* se koristi za smeštanje ulaznih podataka, ali i rezultata zaključivanja koji se koriste u sledećim iteracijama zaključivanja ili kao izlazne informacije.

Softverski alati koji su korišćeni za implementaciju su besplatni i otvorenog koda. Međutim, prirodna posledica toga je da oni imaju određena ograničenja koja komercijalni alati ne moraju da imaju. Zbog nemogućnosti da se pronađe jedinstveni softverski alat koji bi mogao da izvršava i fuzzy i obična pravila, baza znanja je podeljena u dva dela, a zaključivanje se vrši uz pomoć dva alata u dve faze.

Prva faza zaključivanja se odnosi isključivo na fuzzy zaključivanje. U tu svrhu se koristi *JFuzzyLogic* aplikacijski okvir [100] koji je besplatan, otvorenog koda i omogućava predstavljanje fuzzy skupova i pravila i zaključivanje nad istim. Sve fuzzy kategorije i skupovi iz PT kao npr. *novo preduzeće*, *pozitivan profit* itd. su izraženi u jFuzzyLogic sintaksi i mogu se videti na sledećoj slici (Slika 54). U istom alatu su izražena i fuzzy pravila za određivanje trenda vremenske serije. Prednost ovog alata je jednostavnost upotrebe i brz rad, a mana ta što ne može direktno da radi sa Java objektima već podaci moraju da se unose kao pojedinačne promenljive.

```
FUNCTION_BLOCK indicatorsPIR

VAR_INPUT
    profit_category : REAL;
    enterprise_category : REAL;
    average_profit_in_sector_difference : REAL;
    profit_difference : REAL;
    income_difference : REAL;
    cost_difference : REAL;
END_VAR

FUZZIFY profit_category
    TERM negative := (-100, 1) (-4, 1) (-3, 0);
    TERM near_zero := (-4, 0) (-3, 1) (3, 1) (4, 0) ;
    TERM positive := (3, 0) (4, 1) (100, 1);
END_FUZZIFY

FUZZIFY enterprise_category
    TERM new := (0, 1) (2, 1) (4, 0);
    TERM relatively_established := (2, 0) (4, 1) (7, 1) (8, 0) ;
    TERM established := (7, 0) (10, 1) (11, 1);
END_FUZZIFY
```

Slika 54: Deo baze znanja implementiran u jFuzzyLogic alatu

Ova faza zaključivanja funkcioniše na sledeći način. Poslovni podaci uneti preko GUI-ja se prvo preuzimaju iz objekta *Enterprise* klase i unose u fuzzy bazu znanja kao obične promenljive (npr. *profitCategory*). Kada se unesu, podaci se svrstavaju u fuzzy kategorije i vrši se zaključivanje. Na primer, relativna vrednost profita od četiri posto se

može okarakterisati kao pozitivan profit (*positive*) sa vrednošću funkcije pripadnosti jednakoj jedan (Slika 54). Kada se fazi zaključivanje završi, svi zaključci se unose nazad u instancu *Enterprise* klase i prosleđuju u drugu fazu zaključivanja zajedno sa svim ulaznim podacima. Bitno je napomenuti to da se pamti samo jedna fuzzy kategorija po promenljivoj - ona sa najvećom vrednosti funkcije pripadnosti. Na primer, preduzeće koje posluje dve i po godine se može smatrati *novim* sa vrednošću funkcije propadnosti od 0,75 ali i *relativno zrelim* sa vrednošću funkcije pripadnosti od 0,25 (Slika 54). Ipak, preduzeće će biti okarakterisano kao *novo*, i samo ovaj zaključak će se sačuvati u atributu *enterpriseCategory* instance *Enterprise* klase.

U drugoj fazi zaključivanja se kombinuju obična pravila, ulančavanje unapred ali i prethodno doneti fuzzy zaključci da bi se doneli konačni zaključci. Tek u ovoj fazi se poziva mehanizam za objašnjavanje i to u cilju pretvaranja konačnih zaključaka u objašnjenja u formi kontrolisanog jezika. Alati koji se koriste su *Drools Expert* BRE i JEFF mehanizam za objašnjavanje (o kojem je već bilo reči u prethodnim poglavljima).

Drools Expert je okruženje za razvoj sistema zasnovanih na pravilima koje je besplatno, otvorenog koda i napisano je kao aplikacijski okvir u Javi. Od tehnika zaključivanja podržava ulančavanje unapred a konflikti u izvršavanju pravila se mogu razrešiti dodeljivanjem prioriteta pravilima ili obezbeđivanjem da se određeno pravilo može izvršiti samo jednom u toku sesije. Međutim, možda je najvažnije to što Drools Expert koristi Java objekte za skladištenje činjenica, pa je integracija sa drugim aplikacijama napisanim u Javi veoma jednostavna. Razlog zašto je izabrano baš ovo okruženje za implementaciju je taj što je besplatno za sve vrste korišćenja, otvorenog koda i što podržava sve tražene metode i tehnike za predstavljanje znanja i zaključivanje. Deo baze znanja implementiran u Drools Expert BRE se može videti na sledećoj slici (Slika 55).

```
rule "new_or_relatively_established_enterprise_ROI_period_over_positive_profit"
lock-on-active true
salience 90
when
    e:Enterprise (currentProfitCategory == "positive"  &&
      (enterpriseCategory == "new" || enterpriseCategory == "relatively_established" )&&
      ROIPeriodOver == true)
then
    ef.addTextVeryPositive(null,"new_or_relatively_established_enterprise_ROI_period_over_positive_profit",
    new String[]{"current profit","ROI"},null);
    ef.addTextVeryPositive(null,"summary_new_or_relatively_established_enterprise_ROI_period_over_positive_profit",
    new String[]{"summary","current profit","ROI"},null);
end

rule "new_or_relatively_established_enterprise_ROI_period_over_positive_near_zero_profit"
lock-on-active true
salience 90
when
    e:Enterprise (currentProfit >= 0 && currentProfitCategory == "near_zero"  &&
      (enterpriseCategory == "new" || enterpriseCategory == "relatively_established" )&&
      ROIPeriodOver == true)
then
    ef.addTextPositive(null,"new_or_relatively_established_enterprise_ROI_period_over_positive_near_zero_profit",
    new String[]{"current profit","ROI"},null);
    ef.addTextPositive(null,"summary_new_or_relatively_established_enterprise_ROI_period_over_positive_near_zero_profit"
    new String[]{"summary","current profit","ROI"},null);
end
```

Slika 55: Deo baze znanja implementiran u Drools Expert alatu

Drools Expert sintaksa uslovljava da svako pravilo mora da ima jedinstveni identifikator koji se piše posle rezervisane reči *rule*. Oba pravila koja su predstavljena imaju isti prioritet (*salience*) i on iznosi 90. Takođe, podešeno je i da svako od ova dva pravila može da se izvrši samo jednom u toku sesije (*lock-on-active*). Početak premise pravila se uvek označava rezervisanom reči *when*, dok se posledica pravila piše u vidu običnih Java komandi i to posle rezervisane reči *then*. Kraj svakog pravila se označava rezervisanom reči *end*.

Oba pravila sa slike (Slika 55) se odnose na situaciju kada je preduzeće *novo* (new) ili *relativno zrelo* (relatively established) a ROI period je istekao. Prvo pravilo se izvršava onda kada je, osim prethodno navedenih uslova, profit ocenjen kao *pozitivan* (positive), a drugo kada je profit veći od nule ali se smatra za *neutralan* (neutral). Sada je potrebno obratiti pažnju na nekoliko stvari.

Ova pravila koriste i obične činjenice ali i fuzzy činjenice u premisi. Na primer, u prvom pravilu, iz instance klase *Enterprise* se uzima vrednost atributa *currentProfitCategory* i proverava se da li je njegova (string) vrednost postavljena na fuzzy vrednost *positive*. Takođe, proverava se da li atribut *enterpriseCategory* ima fuzzy vrednost *new* ili *relatively established* ali i da li atribut *ROIPeriodOver* ima vrednost *true* (ovo nije fuzzy činjenica). Već je navedeno da su sve fuzzy činjenice izvedene u prvoj fazi zaključivanja i unete u instancu klase *Enterprise*.

Zaključci koji se izvode ovim pravilima su, zapravo, konačna rešenja i ne koriste se u narednim ciklusima zaključivanja. Zbog toga se ne skladište u radnoj memoriji kao nove činjenice, već je jedina aktivnost koja se preduzima u *then* delu pravila pozivanje metoda mehanizma za objašnjavanje (preko objekta *ef*). Iz toga sledi da akcije iz *then* dela pravila uzrokuju dodavanje novog dela objašnjenja u konačno objašnjenje i ništa više.

Na primer, u *then* delu prvog pravila, upućuju se dva poziva mehanizmu za objašnjavanje. Metoda *addTextVeryPositive* pravi i dodaje tekstualni deo objašnjenja u objašnjenje a kontekst tog dela objašnjenja je označen kao *veoma pozitivan* (very positive). Praktično, to znači da se *pozitivan* profit koji je ostvarilo *novo* ili *relativno zrelo* preduzeće nakon isteka ROI perioda smatra za *veoma pozitivnu* stvar. Kontekstualno označavanje sadržaja delova objašnjenja u JEFF-u je veoma korisna stvar jer se, na primer, *negativna* objašnjenja mogu prikazati drugačijom bojom (recimo crvenom) ili drugačijim oblikom ili veličinom slova. Ova metoda ima četiri parametra i njihovo značenje je potrebno objasniti. Prvi parametar je naziv grupe pravila kojoj pravilo pripada. Koristi se za podelu objašnjenja na grupe u slučaju da je baza znanja podeljena na grupe pravila. Svaka grupa objašnjenja je u po jednom property fajlu a naziv grupe je deo naziva fajla. U ovom slučaju, baza nije podeljena u grupe pravila pa ovaj parametar ima vrednost *null*. Drugi parametar predstavlja ključ za učaureni tekst (ključnu reč). U ovom slučaju, identifikator (naziv) pravila se koristi kao ključ u okviru samog property fajla, pa se svako pravilo može na jedinstven način povezati sa odgovarajućim tekstom objašnjenja. Treći parametar je skup *tagova* kojima se mogu označiti delovi objašnjenja i koji se kasnije mogu koristiti za npr. preuređivanje redosleda delova objašnjenja, njihovu selekciju isl. Poslednji parametar je niz dinamičkih vrednosti koje je potrebno uneti na određena mesta u okviru samog teksta objašnjenja. U ovom slučaju to nije potrebno, pa je vrednost ovog parametra takođe *null*.

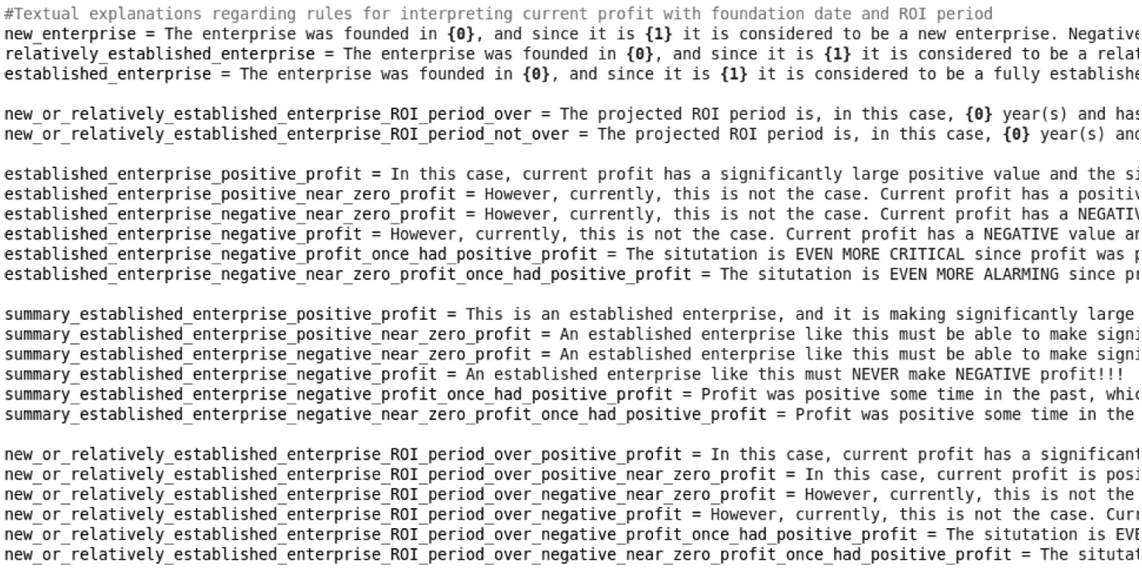
```
rule "current_profit_ROI"
lock-on-active true
salience 96
when
    e:Enterprise (profit != null)
then
    ArrayList<Triple> values = new ArrayList<Triple>();
    values.add(new Triple(e.getCurrentYear(), e.getEnterpriseFoundationYear(), e.getROIPeriod()));
    ef.addData(null, "current_profit_ROI", new String[]{"current profit","ROI"},
      new ThreeDimData( new Dimension("Current year"),new Dimension("Enterprise foundation year"),
      new Dimension("ROI period [years]"),values));
end

rule "new_enterprise"
lock-on-active true
salience 95
when
    e:Enterprise (enterpriseCategory == "new")
then
    Object[] content = new Object[3];
    content[0] = ""+e.getEnterpriseFoundationYear();
    content[1] = ""+e.getCurrentYear();
    ef.addText(null,"new_enterprise",new String[]{"current profit","ROI"},content);
end
```

Slika 56: Deo baze znanja implementiran u Drools Expert alatu

Primer još dva pravila iz dela baze znanja implementirane u Drools Expert alatu se može videti na slici (Slika 56). Pravilo *current_profit_ROI* služi za proveru da li su podaci o profitu uopšte uneti ili ne. Njegovim aktiviranjem se, kao posledica, u objašnjenje unosi jedan deo objašnjenja ali koji predstavlja podatke. Podaci su trodimenzionalni i odnose se na to koja je trenutna godina, koje godine je osnovano preduzeće i koliko je projektovani ROI period. Ovi podaci se kasnije koriste u objašnjenju i mogu da se prikažu tabelarno ili grafikom. Pravilo *new_enterprise* služi da doda objašnjenje o tome kako se došlo do zaključka da je preduzeće novo. Taj zaključak je, zapravo, donet u prethodnoj fazi fuzzy zaključivanja ali se tek ovde objašnjava. Objašnjenje je tekstualno, a sadrži i neke delove koji se dinamički unose na određena mesta u tekstu. Konkretno, niz objekata klase *Object* koji se zove *content* služi tome da te podatke prenese u objašnjenje. Elementi niza mogu biti, efektivno, bilo kojeg tipa. Prvi element niza je godina osnivanja preduzeća, dok je drugi element niza trenutna godina. Kada se *addText* metoda pozove u ovom pravilu, ova dva podatka će biti uneta na ona mesta u tekstu objašnjenja koja su označena vitičastim zagradama.

Već je napomenuto da JEFF skladišti učaureni tekst objašnjenja u običnim property fajlovima. Deo objašnjenja PT aplikacije se može videti na sledećoj slici (Slika 57).



Slika 57: Deo objašnjenja napisanih u JEFF alatu

Svaki deo objašnjenja je predstavljen parom ključ-vrednost i, kao što je navedeno, identifikator pravila predstavlja ključ. Naravno, ovo ne mora da bude tako, već se delovi objašnjenja mogu identifikovati preko bilo kojih ključnih reči. Vrednost iz ovog para predstavlja sam tekst dela objašnjenja koji je u vidu statičkih rečenica kontrolisanog teksta. Dinamički deo objašnjenja se unosi direktno u tekst i označen je vitičastim zagradama i rednim brojem. Na primer, objašnjenje sa ključem *new_enterprise* ima dva mesta na kojima se dinamički unosi sadržaj. Na prvom mestu (označeno sa {0}) se unosi godina osnivanja, a na drugom (označeno sa {1}) trenutna godina. Pravilo koje je povezano sa ovim objašnjenjem je prikazano na prethodnoj slici (Slika 56).

Konačno, potrebno je skrenuti pažnju na to da, iako se koriste fuzzy činjenice u premisama pravila, zaključivanje nije fuzzy tj. nema FAM i koristi se obično ulančavanje unapred.

Konačni izlaz iz ove faze zaključivanja je XML dokument koji predstavlja objašnjenje i sadrži i podatke i informacije. Primer ovakvog dokumenta se može videti na sledećoj slici (Slika 58). Razlog zašto je izabrano da izlazni format posle zaključivanja bude XML a ne recimo PDF je taj što se XML dokumenta lako mogu transformisati i koristiti (od strane alata za izveštavanje) u svrhe generisanja različitih izveštaja. Na ovaj način se sadržaj objašnjenja odvaja od načina njegovog predstavljanja. Potrebno je napomenuti i

to da se ovim dokumentom prenose i svi podaci u izveštaj pa ponovni upit ka izvoru podataka nije potreban. Ovo nije naročito bitno za samu PT aplikaciju jer se podaci preuzimaju sa grafičkog interfejsa, ali bi isti princip bio primenjen i da je izvor podataka DW ili OLAP. Time bi se značajno smanjilo opterećenje nad izvorima pdoataka jer se upiti ne bi postavljali dva puta - pre zaključivanja i posle zaključivanja a pre pravljenja izveštaja.

Struktura dobijenog XML dokumenta je sledeća. Koreni element XML dokumenta je *explanation* (koji predstavlja celo objašnjenje), a svi delovi objašnjenja su njegovi podelementi. Osnovni atributi ovog elementa su datum i vreme kada je nastao (*date*), jezik (*language*) i zemlja (*country*), kao i naslov objašnjenja (*title*). Delovi objašnjenja koji predstavljaju tekst su uokvireni *textualExplanation* elementom, oni koji predstavljaju podatke *dataExplanation* elementom, dok su oni koji predstavljaju slike uokvireni *imageExplanation* XML elementom. Svaki deo objašnjenja ima pridružene atribute koji se direktno preslikavaju iz objektnog modela podataka: naziv pravila (*rule*), naziv grupe pravila (*group*) i kontekst (*context*). U slučaju da neki od ovih atributa nije naveden, ne unosi se u XML dokument. Svaki deo objašnjenja ima i tagove (*tags*), pa je tako za npr. prvi deo objašnjenja koji predstavlja podatke vezan samo jedan tag: “current profit”.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<explanation date="6/11/11 5:00 PM" language="en" country="US" title="Profit interpretation">
    <dataExplanation rule="current_profit" context="informational">
        <tags>
            <tag>current profit</tag>
        </tags>
        <content dimensionName="Year" dimensionName2="Current profit [$]" dimensionName3="Current profit [%]">
            <tripleValue>
                <value1>2010</value1>
                <value2>3327.0</value2>
                <value3>15.583138173302109</value3>
            </tripleValue>
        </content>
    </dataExplanation>
    <textualExplanation rule="current_profit" context="informational">
        <tags>
            <tag>current profit</tag>
        </tags>
        <content>Current profit is 3,327$ or, when expressed relatively 15.583%.</content>
    </textualExplanation>
    <textualExplanation rule="positive_profit" context="positive">
        <tags>
            <tag>current profit</tag>
        </tags>
        <content>The enterprise is making money and profit is SIGNIFICANTLY LARGE which is GOOD.</content>
    </textualExplanation>
    <textualExplanation rule="summary_positive_profit" context="positive">
        <tags>
            <tag>summary</tag>
            <tag>current profit</tag>
```

Slika 58: Deo XML dokumenta koji predstavlja objašnjenje

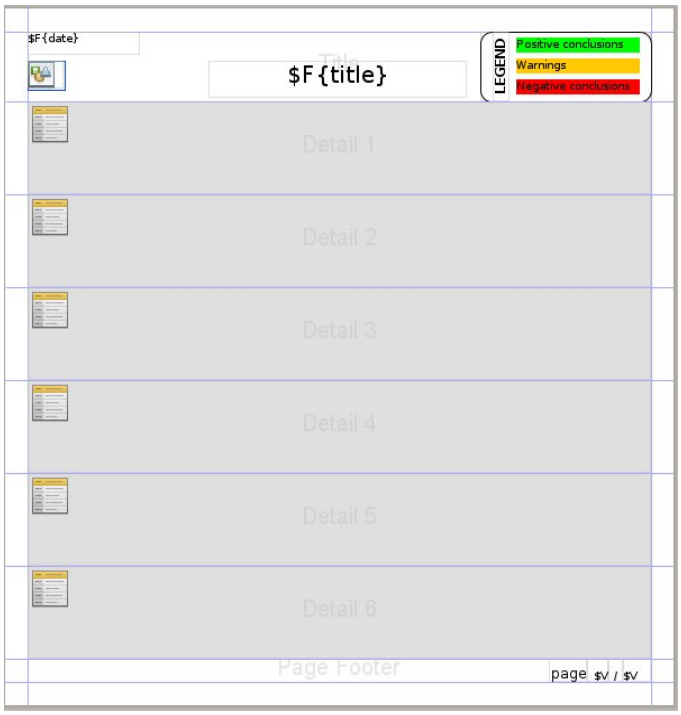
Svaki deo objašnjenja ima i odgovarajući sadržaj (*content*) čije predstavljanje u XML-u zavisi od tipa sadržaja tj. dela objašnjenja. Tekstualni delovi objašnjenja kao sadržaj imaju običan tekst preveden na željeni jezik i sa unetim svim dinamičkim vrednostima. Na primer, drugi deo objašnjenja sa slike (Slika 58) je tekstualni, a sadržaj je sledeći tekst: “Current profit is 3,327$ or, when expressed relatively 15.583%”. Delovi objašnjenja koji predstavljaju podatke u sadržaju imaju podelemente koji predstavljaju vrednosti podataka, a dimenzije podataka i njihove jedinice su označeni direktno u *content* elementu atributima *dimensionName* i *dimensionUnit*. U zavisnosti od dimenzionalnosti podataka, podelementi mogu biti *value* (jednidimenzionalni podaci), *tupleValue* (dvodimenzionalni podaci) i *tripleValue* (trodimenzionalni podaci). Prvi deo objašnjenja iz istog XML dokumenta sadrži trodimenzionalne podatke od koji prvi predstvlja godinu (“2010”), drugi apsolutnu vrednost trenutnog profita izraženu u dolarima (“3327.0”) a treći relativnu vrednost trenutnog profita (“15.583”).

Prethodno predstavljeni XML dokument se transformiše u izveštaj uz pomoć *Jasper Reports* [101] alata. Jasper Reports je aplikacijski okvir u Javi koji je besplatan i otvorenog koda a služi za pravljenje izveštaja u raznim izlaznim formatima kao PDF, XML, HTML, XHTML, Microsoft Word 2003 (DOC), Microsoft Word 2007 (DOCX),

Microsoft Excel 2003 (XLS), Microsoft Excel 2007 (XLSX) itd. Izvori podataka na osnovu kojih se generiše izveštaj mogu biti obični fajlovi (npr. XML ili XLS), ali i relacione baze podataka, web servisi itd. Svaki izveštaj se, zapravo, pravi u vidu praznog šablona u *iReport* grafičkom alatu, a onda se izvršavanjem odgovarajućeg upita nad izvorom podataka popunjava podacima i pretvara u željeni oblik prema šablonu. Distribucija izveštaja se može vršiti putem servera za izveštavanje ali i pokretanjem same osnove servera na jednom računaru u vidu desktop aplikacije.

U ovom slučaju, JasperReports kao ulaz dobija XML fajl, izlaz su dva PDF izveštaja, a ceo proces formiranja izveštaja se odvija u okviru PT aplikacije tj. pokreće se samo osnova servera pozivanjem odgovarajućih Java komandi. Preuzimanje podataka iz XML fajla se vrši preko nekoliko predefinisanih XPath putanja. Prvi izveštaj sadrži samo podatke predstavljene u tabelarnoj i grafičkoj formi, dok drugi izveštaj sadrži te iste podatke (u grafičkoj formi) ali i informacije. Razlog zašto je izabrano da konačni izlaz iz PT aplikacije bude baš PDF format je taj što on podržava širok spektar opcija za formatiranje, a i interoperabilan je.

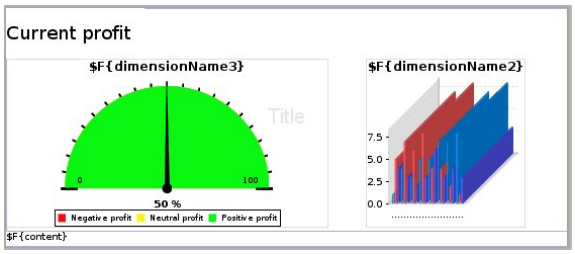
Primer JasperReports šablona koji je korišćen u PT aplikaciji se može videti na sledećoj slici (Slika 59). U pitanju je šablon za izveštaj sa informacijama koji se sastoji iz glavnog dokumenta (prikazan na slici) ali i šest dodatnih dokumenata (podizveštaja) koji čine njegovo telo. Glavni dokument sadrži odrednice za veličinu strane, margine a ima i polja za datum izveštaja, naslov i logo. Ova polja se popunjavaju u trenutku formiranja izveštaja. Glavni dokument sadrži i legendu koja objašnjava boje koje su upotrebljene za označavanje konteksta. Konačno, podešeno je i da svaka strana ima svoj broj ali i da na njoj bude prikazan ukupan broj strana izveštaja.



Slika 59: JasperReports šablon za PT izveštaj - glavni dokument

Slika 60: JasperReports šablon za PT izveštaj - podizveštaj 1

Prvi podizveštaj se prikazuje pri samom početku i sadrži siže najvažnijih zaključaka - informacija. Kao što se na slici vidi (Slika 60), naslov je “Summary” a sav preostali tekst je dat po stavkama. XPath putanja za ovaj deo izveštaja je podešena tako da se u polju sa sadržajem nađu samo oni tekstualni delovi objašnjenja koji sadrže tag “summary”. Ovde, ali i u svim ostalim podizveštajima, važi princip bojenja teksta prema kontekstu pa su pozitivne informacije obojene zelenom bojom, negativne crvenom a upozorenja žutom.

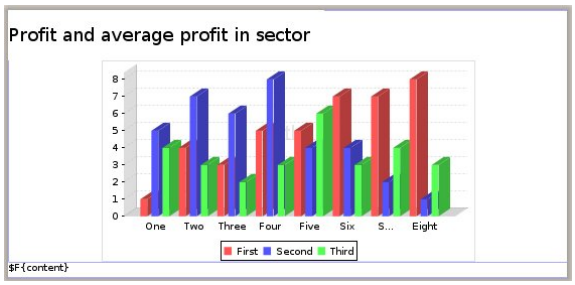


Slika 61: JasperReports šablon za PT izveštaj - podizveštaj 2

Drugi podizveštaj se tiče opštih podataka i informacija o trenutnom profitu (Slika 61). Relativan profit je prikazan uz pomoć obrtomera, pri čemu zone negativnog, neutralnog i pozitivnog profita odgovaraju onima koje su unete kao okvirne granice odgovarajućih fuzzy skupova. Odstupanja profita od zadatih vrednosti su, prema tome, lako uočljiva. Apsolutni profit je prikazan trodimenzionalnim pravougaonim grafikom, a tekstualni deo objašnjenja tj. informacije se dodaju ispod ovih grafika. Na ovaj način, korisnik može odmah da proveri da li informacije odgovaraju podacima. XPath putanja za ovaj deo izveštaja je podešena tako da obuhvate samo oni delovi objašnjenja (i tekstualni i oni sa podacima) koji sadrže samo tag “current profit”.

Slika 62: JasperReports šablon za PT izveštaj - podizveštaj 3

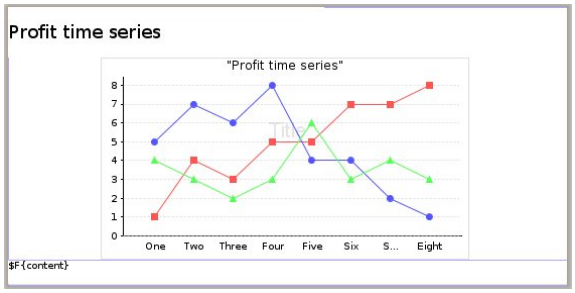
Treći podizveštaj sadrži podatke i informacije o profitu u kontekstu ROI perioda (Slika 62). Informacije su, kao i svuda, u tekstualnom obliku a podaci su dinamički uneti u tekst pa ovaj deo izveštaja ne sadrži grafike.



Slika 63: JasperReports šablon za PT izveštaj - podizveštaj 4

Četvrti podizveštaj (Slika 63) prikazuje trenutni profit upoređen sa prosečnim profitom u grani. Podaci su prikazani na pravougaonom grafiku, a objašnjenja su data ispod njega.

Peti podizveštaj (Slika 64) i šesti podizveštaj (Slika 65) su, u svakom smislu veoma slični, jedino što su podaci prikazani linijskim grafikom (vremenska serija profita) odnosno pravougaonim grafikom sa više nizova podataka (profit, prihodi i troškovi). U oba slučaja, objašnjenja su ispod grafika.



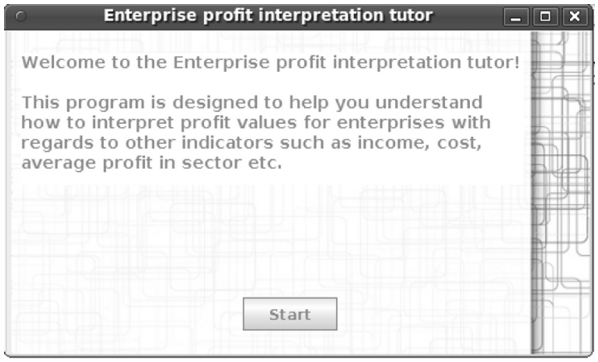
Slika 64: JasperReports šablon za PT izveštaj - podizveštaj 5

Slika 65: Primer JasperReports šablona za izveštaj - podizveštaj 6

Potrebno je napomenuti i to da se JasperReports šablon za PT izveštaj koji ne sadrži informacije već samo podatke razlikuje od prikazanog šablona u dva aspekta. Prvo, ne postoji sekcija sa sižeom (prvi podizveštaj) jer ne postoje informacije koje je moguće sumirati. Drugo, u svakom podizveštaju se koriste isti grafici, ali se umesto dela u kojem je objašnjenje tu nalaze tabele sa podacima sa grafika. Primeri konkretnih izveštaja koji nastaju na osnovu ovih šablona i objašnjenje zašto su napravljena dva izveštaja a ne jedan, se mogu videti u narednom poglavlju.

5.3 Primer korišćenja

Najbolji način da se demonstrira kako PT funkcioniše je kroz jedan jednostavan primer korišćenja. Kada se PT pokrene, prikazuje se početna ekranska forma (Slika 66). Njena osnovna funkcija je da prikaže neke opšte informacije o PT aplikaciji dok se ona ne učita (dugme "Start" postaje aktivno tek kad se učitavanje završi). Razlog zašto je uvedena je taj što je inicijalizacija dela PT baze znanja u Drools-u veoma procesorski zahtevna aktivnost, pa i pored određenih uvedenih optimizacija uvek traje bar nekoliko sekundi. Ovo važi samo za prvo pokretanje aplikacije jer se svi naknadni pozivi bazi znanja upućuju jednoj već učitanoj instanci i traju neuporedivo kraće.

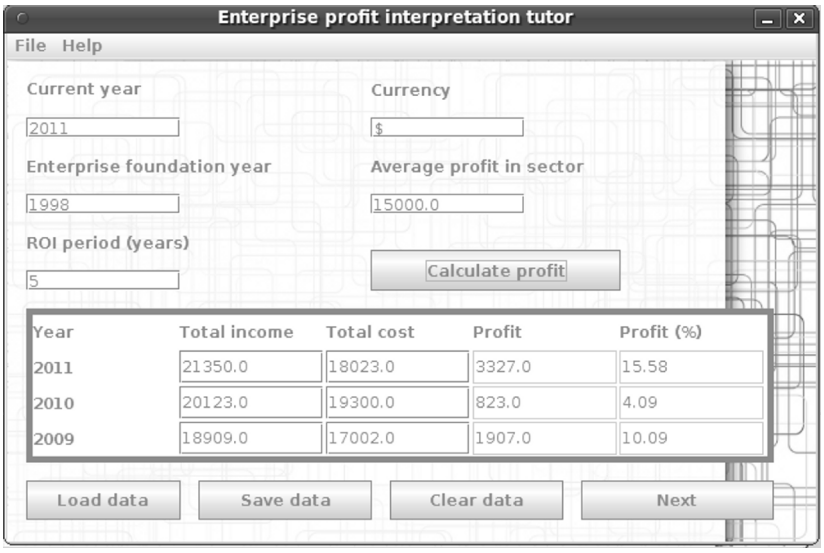


Slika 66: Početna PT ekranska forma

Glavna ekranska forma sa već unetim podacima se može videti na sledećoj slici (Slika 67). Potrebno je primetiti da je, u principu, GUI veoma jednostavan. Ideja je da PT aplikacija bude što jednostavnija za korišćenje i to je osnovni razlog. Zbog toga su sve složene navigacije kroz menije, pomoćni prozori i nepotrebni dijalozi izbačeni iz konačne verzije GUI-ja.

Osnovna svrha glavne ekranske forme je da omogući studentima da unesu podatke i pokrenu proces njihovog tumačenja. Već je navedeno da se ulazni podaci sastoje iz jedanaest vrednosti: trenutna godina, godina osnivanja preduzeća, ROI period (u godinama), valuta, prosečan profit u grani privrede, ukupni prihod za poslednje tri godine (trenutna, prošla i pretprošla godina) i ukupni troškovi za poslednje tri godine. Profit, u apsolutnom i relativnom iznosu, izračunava PT automatski kada se unesu ukupni prihod i ukupni troškovi i kada se pritisne dugme "Calculate profit".

Osim ručnog unosa podataka, oni mogu biti učitani iz fajla ili sačuvani u fajl (dugme "Load data" i dugme "Save data"). Na ovaj način je omogućeno da se skup predefinisanih primera distribuira zajedno sa aplikacijom i služi kao osnova za podučavanje tj. kako varijacije u podacima mogu da utiču na njihovo tumačenje. Međutim, glavna prednost je to što korisnici mogu da prave svoje primere, da ih čuvaju i koriste kasnije. Konačno, ako korisnik želi da izbriše sve podatke sa glavne forme, dovoljno je da pritisne dugme "Clear data". Pokretanje procesa zaključivanja počinje pritiskanjem dugmeta "Next".

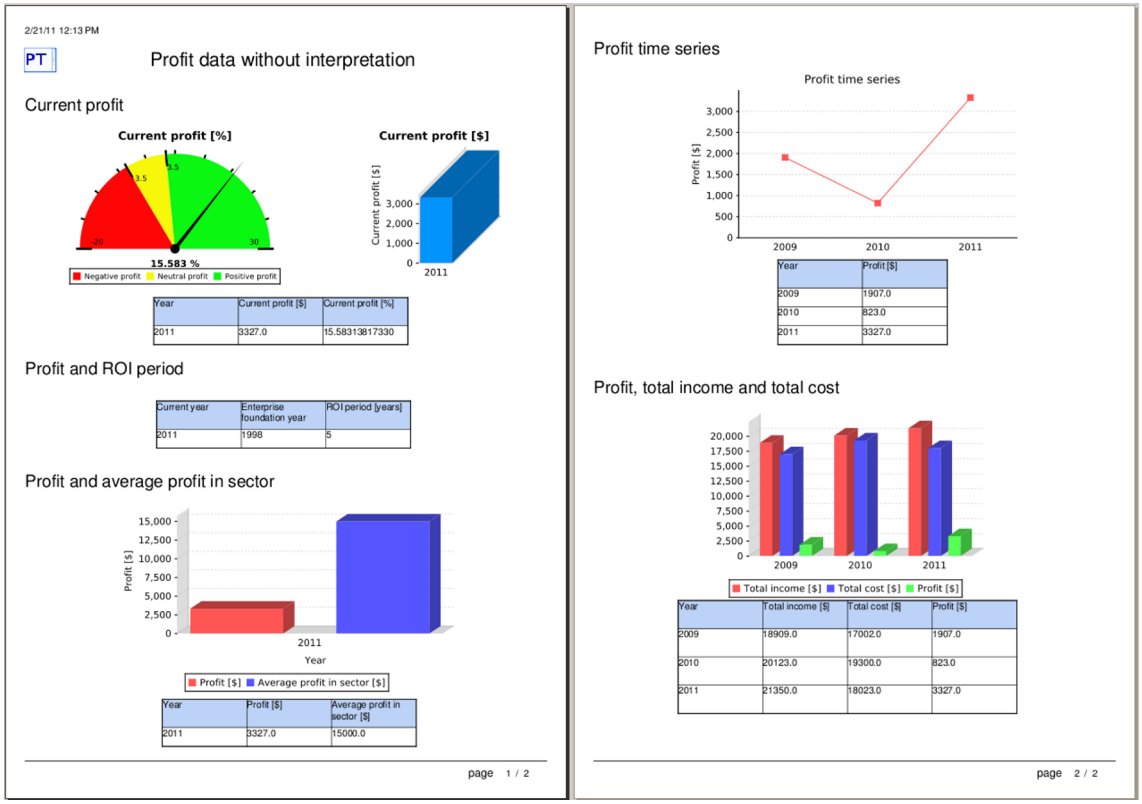


Slika 67: Glavna PT ekranska forma

Sledeće što se korisniku prikazuje su tri PDF fajla. Prvi predstavlja izveštaj koji sadrži podatke, ali ne i njihovo tumačenje, drugi sadrži upitnik, a treći je izveštaj koji sadrži i podatke i njihovo tumačenje tj. informacije. Zamišljeno je da korisnik (u ovom slučaju student) prvo pogleda izveštaj samo sa podacima i pokuša da samostalno odgovori na

pitanja predstavljena upitnikom. Kada to završi, on ili ona može da pogleda izveštaj koji sadrži i tumačenja i da proveri da li su odgovori tačni tj. da nauči kako se dati skup podataka tumači.

Izveštaj koji sadrži samo podatke (Slika 68) je podeljen u pet delova, a svi podaci su predstavljeni tabelarno, ali i putem grafika i obrtomera. Prvi deo ovog izveštaja (sa naslovom "Current profit") sadrži podatke o trenutnom profitu predstavljene obrtomerom, trodimenzionalnim pravougaonim grafikom, ali i tabelom. Već je navedeno da je obrtomer podeljen u tri sekcije (crvenu, žutu i zelenu) koje se odnose na ciljne vrednosti profita. Sa obzirom na to da je strelica u zelenoj sekciji, smatra se da je profit postigao ciljnu vrednost. Pravougaoni grafik prikazuje samo apsolutni iznos profita, a tabela sadrži sve ove podatke zajedno. Drugi deo izveštaja (sa naslovom "Profit and ROI period") sadrži samo jednu tabelu u kojoj se nalaze podaci o trenutnoj godini, godini osnivanja preduzeća i ROI periodu. U trećem delu izveštaja (sa naslovom "Profit and average profit in sector") se trodimenzionalnim pravougaonim grafikom prikazuju trenutni profit i prosečni profit u grani privrede uporedo, dok tabela sadrži oba ova podatka. Iz ovoga se može videti da je trenutni profit manji od prosečnog u grani privrede. Vremenska serija profita je prikazana u četvrtom delu (sa naslovom "Profit time series") i putem linijskog grafika i tabelarno. U ovom slučaju, trenutni profit je veći u odnosu na prošlu godinu i pretprošlu godinu ali je u međuvremenu imao jedan manji pad. Poslednji deo izveštaja (naslov "Profit, total income and total cost") sadrži trodimenzionalni grafik sa više serija podataka i tabelu u kojima su prikazani prihodi, troškovi i profit u poslednje tri godine. Iz ovih podataka se može zaključiti da su prihodi konstantno rasli u poslednje tri godine a da su troškovi, u proseku, porasli (prvo su porasli pa onda malo opali, ali je konačni efekat ipak rast troškova).



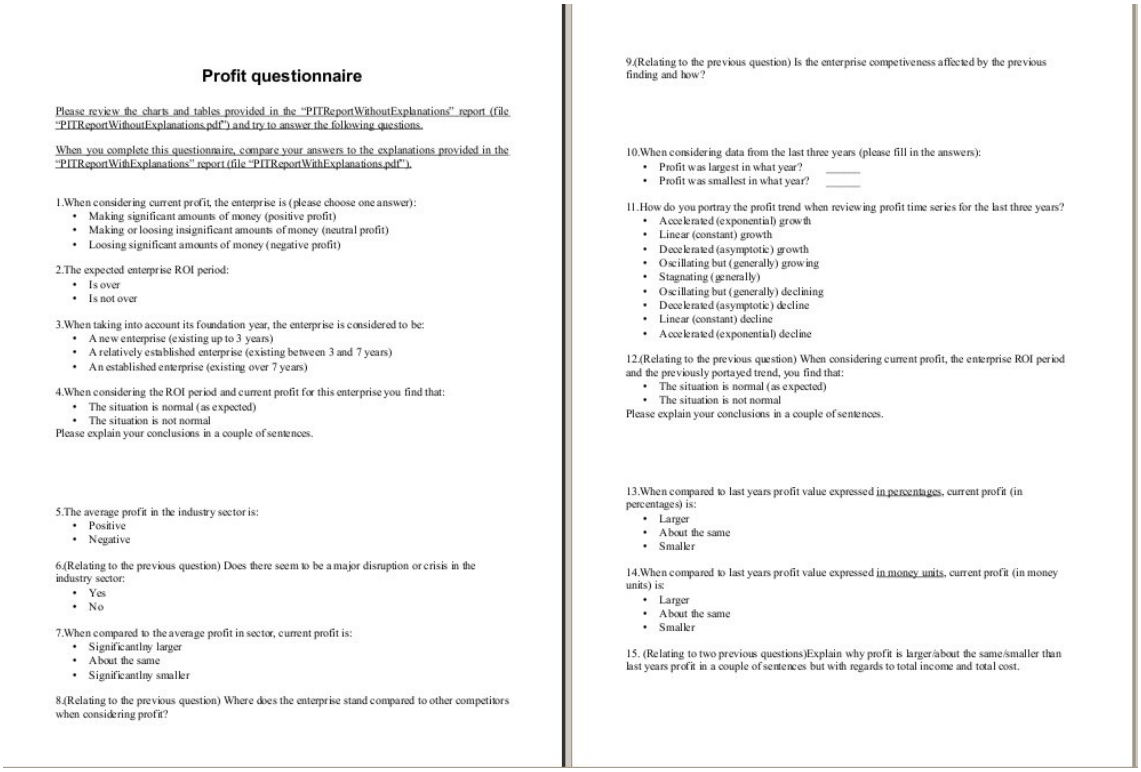
Slika 68: PT izveštaj koji sadrži samo podatke

Drugi PDF fajl (Slika 69) sadrži upitnik čija svrha je da omogući studentima da provere svoje znanje. Da bi mogli da odgovore na petnaest pitanja koja se u njemu nalaze, studenti moraju sami da protumače podatke date u prvom PDF fajlu. Sama pitanja su uvek ista i formulisana su tako da pruže strukturu samom procesu tumačenja, a i da obuhvate sve važne zaključke koji se mogu izvesti iz podataka. U ovom trenutku, upitnik nije interaktivan pa se podrazumeva da studenti beleže svoje odgovore sa strane i onda proveravaju treći PDF fajl radi rešenja.

Treći PDF fajl predstavlja izveštaj koji sadrži i podatke i informacije (Slika 70). Već je napomenuto da ovaj izveštaj ima iste sekcije kao i izveštaj bez informacija, samo je dodat siže na sam početak. Osim toga, podaci su predstavljeni istim graficima, ali se umesto tabela mogu naći informacije tj. tumačenja podataka. Informacije su predstavljene u vidu rečenica kontrolisanog jezika i formatirane su u skladu sa kontekstom: *pozitivan* i *veoma pozitivan* kontekst su prikazani sa pozadinom zelene boje, *upozorenja* sa pozadinom žute, a *negativan* i *veoma negativan* kontekst sa

pozadinom crvene boje. Kontekstualno neutralne informacije su prikazane sa belom pozadinom (to se može videti i u legendi predstavljenoj u gornjem desnom ćošku prve strane izveštaja).

Siže izveštaja sa slike (Slika 70) sadrži najvažnije zaključke predstavljene u kratkim crtama i ilustruje prethodne principe. U ovom slučaju, on sadrži pet stavki. Ideja je da korisnik pročita siže da bi stekao uvid u generalno stanje poslovanja, a onda da pređe na ostale delove izveštaja da bi se upoznao sa detaljima. Sve stavke osim treće predstavljaju *pozitivan* kontekst. Poslednja stavka zapravo predstavlja *veoma pozitivan* kontekst i predstavljena je malo većim slovima. Ukratko, prvo je navedeno da je ostvareni profit značajno velik tj. u skladu sa ciljevima (prva stavka), da je u pitanju zrelo preduzeće pa je pozitivan profit jedina opcija koja se može smatrati odgovarajućom (druga stavka), da je profit u poslednjih par godina oscilovao ali da je rastao (četvrta stavka) i da je profit veći u odnosu na prošlogodišnji i u apsolutnom i u relativnom iznosu a da je uzrok tome rast prihoda i pad troškova (peta stavka). Treća stavka predstavlja upozorenje da je trenutni profit značajno manji od prosečnog profita u grani privrede.



Profit questionnaire

Please review the charts and tables provided in the "PITReportWithoutExplanations" report (file "PITReportWithoutExplanations.pdf") and try to answer the following questions.

When you complete this questionnaire, compare your answers to the explanations provided in the "PITReportWithExplanations" report (file "PITReportWithExplanations.pdf").

1.When considering current profit, the enterprise is (please choose one answer):
- Making significant amounts of money (positive profit)
- Making or loosing insignificant amounts of money (neutral profit)
- Loosing significant amounts of money (negative profit)

2.The expected enterprise ROI period:
- Is over
- Is not over

3.When taking into account its foundation year, the enterprise is considered to be:
- A new enterprise (existing up to 3 years)
- A relatively established enterprise (existing between 3 and 7 years)
- An established enterprise (existing over 7 years)

4.When considering the ROI period and current profit for this enterprise you find that:
- The situation is normal (as expected)
- The situation is not normal

Please explain your conclusions in a couple of sentences.

5.The average profit in the industry sector is:
- Positive
- Negative

6.(Relating to the previous question) Does there seem to be a major disruption or crisis in the industry sector:
- Yes
- No

7.When compared to the average profit in sector, current profit is:
- Significantlny larger
- About the same
- Significantlny smaller

8.(Relating to the previous question) Where does the enterprise stand compared to other competitors when considering profit?

9.(Relating to the previous question) Is the enterprise competiveness affected by the previous finding and how?

10.When considering data from the last three years (please fill in the answers):
- Profit was largest in what year? _____
- Profit was smallest in what year? _____

11.How do you portray the profit trend when reviewing profit time series for the last three years?
- Accelerated (exponential) growth
- Linear (constant) growth
- Decelerated (asymptotic) growth
- Oscillating but (generally) growing
- Stagnating (generally)
- Oscillating but (generally) declining
- Decelerated (asymptotic) decline
- Linear (constant) decline
- Accelerated (exponential) decline

12.(Relating to the previous question) When considering current profit, the enterprise ROI period and the previously portayed trend, you find that:
- The situation is normal (as expected)
- The situation is not normal

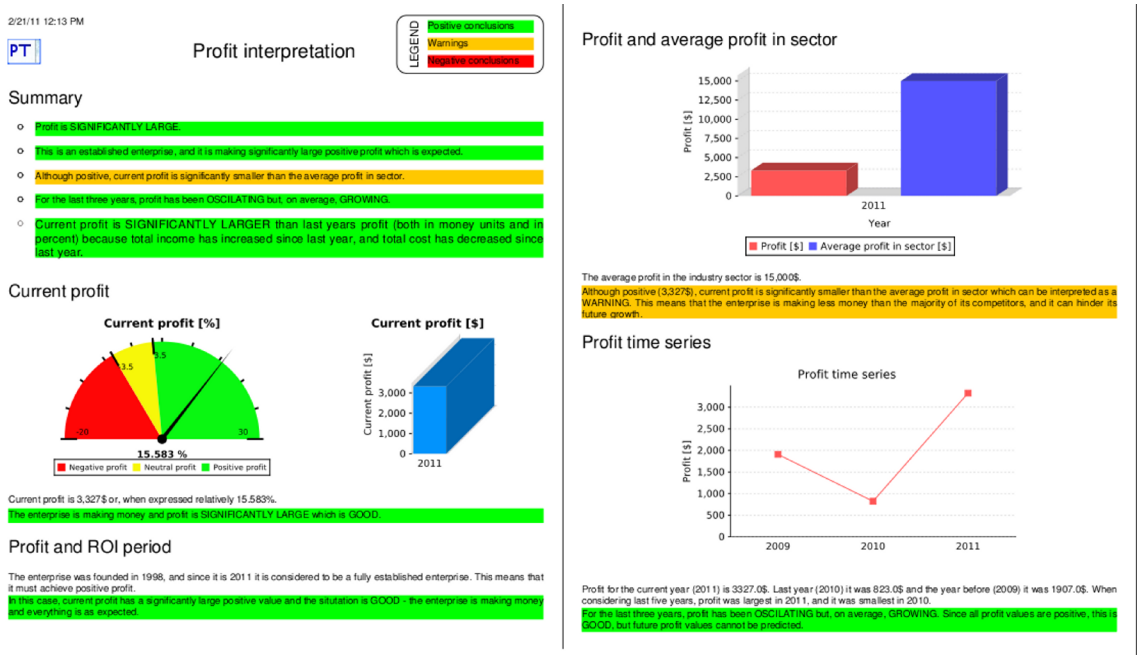
Please explain your conclusions in a couple of sentences.

13.When compared to last years profit value expressed in percentages, current profit (in percentages) is:
- Larger
- About the same
- Smaller

14.When compared to last years profit value expressed in money units, current profit (in money units) is:
- Larger
- About the same
- Smaller

15. (Relating to two previous questions)Explain why profit is larger/about the same/smaller than last years profit in a couple of sentences but with regards to total income and total cost.

Slika 69: PT upitnik o profitu



2/21/11 12:13 PM

PT

Profit interpretation

LEGEND: Positive conclusions; Warnings; Negative conclusions

Summary

- Profit is SIGNIFICANTLY LARGE.
- This is an established enterprise, and it is making significantly large positive profit which is expected.
- Although positive, current profit is significantly smaller than the average profit in sector.
- For the last three years, profit has been OSCILATING but, on average, GROWING.
- Current profit is SIGNIFICANTLY LARGER than last years profit (both in money units and in percent) because total income has increased since last year, and total cost has decreased since last year.

Current profit

Current profit [%]

15.583 %

Negative profit | Neutral profit | Positive profit

Current profit [$]

Current profit [$]: 3,000; 2,000; 1,000; 0; 2011

Current profit is 3,327$ or, when expressed relatively 15.583%.

The enterprise is making money and profit is SIGNIFICANTLY LARGE which is GOOD.

Profit and ROI period

The enterprise was founded in 1998, and since it is 2011 it is considered to be a fully established enterprise. This means that it must achieve positive profit.

In this case, current profit has a significantly large positive value and the situtation is GOOD - the enterprise is making money and everything is as expected.

Profit and average profit in sector

Profit [$]: 15,000; 12,500; 10,000; 7,500; 5,000; 2,500; 0

2011

Year

Profit [$] | Average profit in sector [$]

The average profit in the industry sector is 15,000$.

Although positive (3,327$), current profit is significantly smaller than the average profit in sector which can be interpreted as a WARNING. This means that the enterprise is making less money than the majority of its competitors, and it can hinder its future growth.

Profit time series

Profit time series

Profit [$]: 3,000; 2,500; 2,000; 1,500; 1,000; 500; 0

2009; 2010; 2011

Profit for the current year (2011) is 3327.0$. Last year (2010) it was 823.0$ and the year before (2009) it was 1907.0$. When considering last five years, profit was largest in 2011, and it was smallest in 2010.

For the last three years, profit has been OSCILATING but, on average, GROWING. Since all profit values are positive, this is GOOD, but future profit values cannot be predicted.

Slika 70: Deo PT izveštaja koji sadrži i podatke i informacije - primer 1

Informacije u prvom delu izveštaja posle sižea se tiču trenutnog profita i tu se prvo

navodi koliko profit iznosi u apsolutnom i relativnom iznosu. Ova informacija je kontekstualno neutralna, pa rečenica ima belu pozadinu i konkretan je primer kako izgleda kada se dinamičke vrednosti (u ovom slučaju profit u relativnom i apsolutnom iznosu kao i oznaka valute) unesu u statičan učaureni tekst. Nakon toga, navodi se da je profit protumačen kao *pozitivan* tj. značajno velik u skladu sa zadatim ciljnim vrednostima.

Deo izveštaja koji se odnosi na profit i ROI period sadrži nekoliko zaključaka. Prvo, navedeno je kada je preduzeće osnovano (1998. godine), koja je trenutna godina (2011.) i koliko je dugačak ROI period (5 godina). Na osnovu ovoga je zaključeno da je u pitanju zrelo preduzeće i da mora da ostvaruje pozitivan profit jer je ROI period istekao. Kada je vrednost trenutnog profita upoređena sa ovim činjenicama, zaključeno je da je situacija očekivana i da je sve u redu i to je označeno pozitivnim kontekstom.

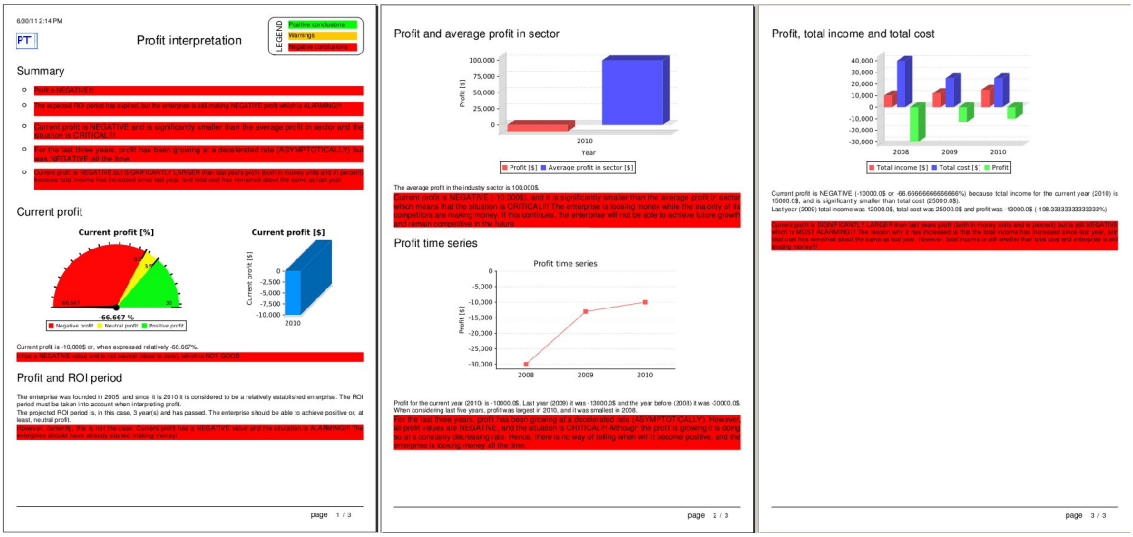
Kada je profit upoređen sa prosečnim profitom u grani (sledeći deo izveštaja), uočeno je da je profit značajno manji od prosečnog u grani i ovaj zaključak je označen kao upozorenje. U produžetku je naznačeno da ovo zapravo znači da preduzeće (u proseku) ostvaruje manji profit u odnosu na konkurenciju, ali i koje su moguće posledice ako se ovo stanje nastavi - potencijalni gubitak konkurentnosti zbog sporijeg razvoja u odnosu na konkurenciju.

Analiza vremenske serije profita počinje navođenjem vrednosti profita za poslednje tri godine u vidu rečenice i naznakom u kojoj godini je bio najviši a u kojoj najniži profit. Nakon toga, zaključeno je da kretanje profita u poslednje tri godine najbolje može da se opiše kao oscilujući rast. Takođe, navedeno je da to znači da je situacija u redu sa obzirom na to da je profit pozitivan, ali da se buduće vrednosti profita ne mogu predvideti.

Poslednji deo izveštaja se odnosi na tumačenje profita u odnosu na troškove i prihod (nije prikazan na slici). U njemu se, kao zaključak, navodi da je uzrok rasta profita porast prihoda i pad troškova što vodi većoj efektivnosti (povećala se prodaja proizvoda tj. usluga) ali i većoj efikasnosti (jeftinija proizvodnja sa niže troškova).

Još jedan primer PT izveštaja sa podacima i informacijama se može videti na slici (Slika

71). Uneti skup podataka je drugačiji, pa ni tumačenje nije isto. Konkretno, može se videti da su zaključci *negativnog* konteksta jer je trenutni profit negativan a preduzeće je zrelo (pa bi moralo da ostvaruje pozitivan profit). Osim toga, profit je manji od prosečnpg u grani privrede a kretanje profita je eksponencijalni rast. Ovo poslednje je okarakterisano kao negativan zaključak zbog toga što je, iako ubrzano raste, profit u poslednje tri godine sve vreme bio negativan. Konačno, uzrok rastu profita je rast prihoda i opadanje troškova, ali su troškovi i dalje veći od prihoda što se (za jedno zrelo preduzeće) smatra nedopustivim.



Slika 71: PT izveštaj koji sadrži podatke i informacije - primer 2

6 Evaluacija

U ovom poglavlju je prikazana diskusija o predloženom rešenju kao i evaluacija postignutih rezultata i mogući pravci budućeg razvoja. Sa obzirom na to da rezultati obuhvataju dve konkretne softverske aplikacije a da se diskusija odnosi na sam teorijski

pristup, poglavlje je podeljeno u četiri celine.

Prvi deo poglavlja sadrži neka opšta zapažanja i razmišljanja autora o potencijalnim pozitivnim i negativnim aspektima predloženog pristupa. Naravno, ona nisu naučno potkrepljena već predstavljaju diskusiju i mogu da otvore put ka pravcima daljih istraživanja.

U drugom delu je prikazana evaluacija JEFF alata. Ona je bila dvojaka i sastojala se od provere od strane krajnjih korisnika, ali i komparativne analize u kojoj se JEFF poredio sa sličnim mehanizmom za objašnjavanje. Od krajnjih korisnika se zahtevalo da naprave ES i da objašnjenja u okviru njega urade uz pomoć JEFF-a a da, nakon toga, popune anketu u vezi sa tim kakva su bila njihova iskustva u radu sa JEFF-om. Komparativna analiza je obuhvatila poređenje sa mehanizmom za objašnjavanje alata Exsys Corvid prema više izabranih kriterijuma.

Treći deo poglavlja obuhvata evaluaciju PT aplikacije. Ona prikazuje utiske krajnjih korisnika u radu sa ovim alatom kao nastavnim sredstvom. Rad sa aplikacijom je uveden kao deo vežbi iz predmeta ekonomika preduzeća diplomskih studija iz menadžmenta, a krajnji korisnici su bili studenti.

Poslednji deo ovog poglavlja obuhvata diskusiju o obe napravljene aplikacije kao i moguće pravce budućeg razvoja.

6.1 Diskusija o potencijalnim pozitivnim i negativnim aspektima predloženog pristupa

U ovom podpoglavlju su opisani potencijalni pozitivni i negativni aspekti predloženog teorijskog pristupa. Naravno, u pitanju je samo diskusija, a provera nekih od ovih pretpostavki je moguća tek posle evaluacije sa grupom krajnjih korisnika. Rezultati jedne ovakve evaluacije su dati u podpoglavlju u kojem je izvršena evaluacija prototipa (pogllavlje 6.3).

Potencijalni pozitivni aspekti predloženog pristupa su:

- Menadžeri ne moraju “ručno” da analiziraju podatke

- Potrebno je veoma malo tehničkog znanja da se u potpunosti iskoristi potencijal SPI
- Velike količine podataka mogu da budu automatski protumačene
- Informacije su “na dohvat ruke”
- Informacije su objektivne
- Informacije su konstantnog kvaliteta

Najočiglednija korist od primene ovakvog pristupa je ta da menadžeri ne moraju da ručno analiziraju podatke. Ovo prvenstveno vodi ka značajnim uštedama u vremenu, a i trudu potrebnom da se stekne uvid u uspešnost poslovanja. Dodatna pogodnost automatskog stvaranja informacija i njihovog predstavljanja u vidu rečenica kontrolisanog jezika je ta da nema potrebe za daljom analizom tj. pravljenjem bilo kakvih upita ka OLAP ili DW skladištima podataka. To znači da čak i korisnici sa veoma niskim nivoom tehničkog znanja mogu da u potpunosti iskoriste potencijale koji su im pruženi.

Još jedna od pozitivnih posledica je ta što se sada mogu obraditi (protumačiti) velike količine podataka odjednom. Time se rešava problem “planina podataka” koje čekaju da ih neko pregleda. Na primer, potpuno je izvodljivo da se portfelj od nekoliko hiljada proizvoda jedne firme analizira i rangira za nekoliko minuta. Prema tome, firme koje posluju sa velikim brojem filijala ili podfirmi mogu da automatski prate njihove performanse. Ovo može da dovede i do povećanja motivacije za korišćenjem SPI jer se dobija sve više korisnih informacija umesto da se gomilaju podaci.

Međutim, najvažnije je to da su sada informacije dostupne bilo kada, bilo gde i “na dohvat ruke”. Isključivanjem ljudskog faktora u transformaciji podataka u informacije, dobija se mogućnost da one budu objektivne i u okviru nekog projektovanog kvaliteta. Drugim rečima, objektivnost informacija nije pod uticajem sklonosti ili interesa osobe koja ih tumači, a njihov kvalitet ne zavisi od znanja, koncentracije ili iskustva te iste osobe.

Pristup može da proizvede i neke negativne aspekte:

- Nezadovoljstvo menadžera zbog gubitka kontrole nad analizom podataka

- Kvalitet ugrađenog znanja direktno utiče na kvalitet informacija
- Kvantitet ugrađenog znanja direktno utiče na kvantitet informacija
- Složenost održavanja baze znanja

Ako se predloženi pristup implementira na odgovarajući način, menadžeri ne moraju da "ručno" analiziraju podatke u izveštajima, već će biti dovoljno da pročitaju informacije. Moguće je da ovo izazove skepticizam kod menadžera i da oni razviju sumnje u kvalitet automatski izvedenih informacija. Zbog toga se informacije prikazuju zajedno sa podacima pa menadžeri mogu da provere da li su u skladu sa podacima. Takođe, ključno je to što su informacije prikazane na način koji odgovara korisnicima tj. prilagođene su njihovom rečniku i nivou znanja.

Znanje potrebno za tumačenje KPI mora da bude izuzetno visokog kvaliteta da bi rešenje funkcionisalo kako treba. Ako je uneto znanje nepotpuno, nekonzistentno ili irelevantno, informacije su, prirodno, veoma slabog kvaliteta. Moglo bi da se desi da neki zaključci uopšte ne budu izvedeni, ili da sistem pruža nebitne ili kontradiktorne informacije. Ovaj problem se rešava sa dva aspekta: dobrim prikupljanjem znanja i detaljnim testiranjem sistema. Na sreću, već postoji veliki broj objektivnih i standardizovanih izvora znanja za tumačenje KPI a to su univerzitetske i stručne knjige, časopisi i studije slučaja. Što se tiče testiranja, neophodno je da ono bude urađeno detaljno i da obuhvata procene nekoliko eksperata u smislu tačnosti i potpunosti izvedenih informacija.

I kvantitet znanja značajno utiče na izvedene informacije. Čak iako je znanje najvišeg kvaliteta, ako ga nema dovoljno to dovodi do stvaranja male količine informacija. Ove informacije, iako bitne, ne pružaju dovoljno detaljan uvid u poslovanje. Mala količina znanja se, u ovom slučaju, ne odnosi na nepotpunost znanja jer nije ništa greškom izostavljeno.

Konačno, znanje koje je obuhvaćeno ovakvim sistemom nije nepromenljivo. Održavanje baze znanja tj. njeno ažuriranje znanja koje je u njoj može da predstavlja veoma složenu aktivnost. Delovi znanja koji se, u ovom slučaju, izuzetno često menjaju su ciljne vrednosti KPI. One su podložne uticaju politike preduzeća, strategije ali i

mnogih spoljašnjih faktora, pa njihovo prilagođavanje trenutnoj situaciji može biti vršeno na godišnjem, ali i na mesečnom, nedeljnom, pa čak i dnevnom nivou. Sa druge strane, jednom kada se definišu fuzzy kategorije (skupovi) na osnovu ciljnih vrednosti, dalje zaključivanje se vrši nad tim kategorijama i ne ulazi se u pojedinosti - koje su granice svakog skupa isl. Prema tome, pravo rešenje je da se, pri projektovanju i implementaciji, omogući da deo znanja koji se odnosi na ciljne vrednosti KPI bude što je više moguće odvojen u odnosu na ostatak baze znanja radi lakšeg održavanja. Inače, proceduralno znanje koje se koristi za tumačenje vrednosti samih pokazatelja je uglavnom veoma dobro struktuirano i ustaljeno, tako da se ne mogu očekivati velike promene već samo dodavanje novog znanja u cilju tumačenja nekih dodatnih pokazatelja.

Ključna razlika između predloženog pristupa i pristupa u klasičnim SPI je u njihovoj orijentaciji. SPI su ***orijentisani ka podacima***. Njihov osnovni cilj je, zapravo, ekstrakcija, transformacija, učitavanje, sumiranje i predstavljanje podataka. Sistemi poslovne inteligencije 2.0 i poslovne inteligencije u realnom vremenu iskazuju ovu karakteristiku još više: podaci se procesiraju toliko brzo da izveštaji postaju dostupni skoro trenutno. Predloženi pristup je ***orijentisan ka informacijama***. Iako ponuđeni izveštaji i dalje sadrže podatke, fokus je na informacijama i njihovom odgovarajućem predstavljanju. Posledica toga je da se podaci ne tretiraju kao izlaz sistema već prvenstveno kao ulaz.

6.2 Evaluacija JEFF mehanizma za objašnjavanje

Cilj evaluacije JEFF alata je bio da se utvrdi da li je JEFF postigao zadate ciljeve, kao i kako stoji u poređenju sa drugim mehanizmima za objašnjavanje. Prvi deo evaluacije se sastoji od nekih opštih opažanja i rezultata ankete sa krajnjim korisnicima. Drugi deo je komparativna analiza sa jedinim sličnim mehanizmom za objašnjavanje koji je aktuelan i dostupan za korišćenje - mehanizmom za objašnjavanje Exsys Corvid alata.

6.2.1 Evaluacija sa krajnjim korisnicima

JEFF ispunjava najveći broj ciljeva koji su postavljeni pred njega. On pruža objašnjenja u rečenicama kontrolisanog jezika ali i trag izvršavanja pravila. Objašnjenja mogu biti napisana na različitim jezicima, a dodavanje novog jezika ne utiče na sam alat niti njegove delove ili strukturu ni na koji način. Osim teksta, objašnjenja mogu sadržati i slike, tabele i grafike a mogu se dobiti u vidu PDF, XML ili običnog tekstualnog dokumenta. JEFF je i lak za održavanje i nadogradnju jer su primenjivani principi softverskog inženjerstva u toku njegovog stvaranja. Na primer, ako je potrebno dodati novi tip dela objašnjenja, dovoljno je napraviti samo par novih podklasa - ostatak aplikacijskog okvira se ne menja.

Međutim, jedini način da se utvrdi da li je JEFF jednostavan za korišćenje i da li se lako integriše u postojeće BRE i BRMS je bio da se izvrši evaluacija od strane krajnjih korisnika.

Pošto izvodimo vežbe na predmetu "inteligentni sistemi" na Fakultetu organizacionih nauka, odlučili smo da nam studenti osnovnih studija sa smera informacioni sistemi i studenti master studija sa smera softversko inženjerstvo koji slušaju ovaj predmet budu ispitanici. Svi ispitanici su se prijavili dobrovoljno i nikakav dodatni kriterijum selekcije nije bio primenjen. Njihovo prethodno obrazovanje i položeni predmeti su osigurali da imaju detaljno znanje programskog jezika Java, alata za razvoj Java programa kao i principa softverskog inženjerstva. Za istraživanje se prijavilo ukupno 23 studenta.

U toku predavanja na predmetu Inteligentni sistemi studentima su objašnjeni osnovni principi, metode i tehnike ES, a na vežbama su urađeni konkretni primeri ES u Drools Expert BRE kao i kako se prave objašnjenja uz pomoć JEFF mehanizma za objašnjavanje. Nakon održanih predavanja i vežbi, od prijavljenih studenata se očekivalo da naprave ES u Drools Expert 5.1 BRE, što su oni uspešno i izveli. Ovaj ES je trebalo da sadrži makar trideset pravila i makar osam pitanja koja se postavljaju korisniku i trebalo je da bude implementiran kao desktop Java aplikacija.

Evaluacija se sastojala u tome da u ES koji su već napravili, studenti dodaju i objašnjenja uz pomoć JEFF-a, i da onda da odgovore na niz pitanja (putem upitnika)

koja su se ticala celog iskustva. Osnovni ciljevi evaluacije su bili da se utvrdi da li je JEFF jednostavan za korišćenje i da li ga je lako integrisati u Drools Expert BRE. Bilo je potrebno utvrditi koliko lako se prave tekstualni delovi objašnjenja, koliko lako se dodaju slike u tekst, da li se jednostavno može napraviti izveštaj itd. Svaki osnovni cilj je definisan preko generalne hipoteze i razrađen putem nekoliko sekundarnih hipoteza:

(i) JEFF je jednostavan za korišćenje.

(a) JEFF se jednostavno inicijalizuje i omogućava jednostavan izbor željenog jezika.

(b) Pravljenje (i prevođenje) tekstualnog sadržaja objašnjenja u JEFF-u je jednostavno.

(c) Pravljenje slika kao sadržaja objašnjenja u JEFF-u je jednostavno.

(d) Pravljenje podataka kao sadržaja objašnjenja u JEFF-u je jednostavno.

(e) Pravljenje PDF izveštaja u JEFF-u je jednostavno.

(f) Pravljenje XML izveštaja u JEFF-u je jednostavno.

(g) Pravljenje običnih tekstualnih izveštaja u JEFF-u je jednostavno.

(ii) JEFF se lako integriše u radu sa postojećim alatima (Drools Expert BRE).

(h) JEFF može lako da se poziva direktno iz baze znanja.

(i) Povezivanje tekstualnog sadržaja objašnjenja sa pravilima je jednostavno.

(j) Povezivanje slika kao sadržaja objašnjenja sa pravilima je jednostavno.

(k) Povezivanje podataka kao sadržaja objašnjenja sa pravilima je jednostavno.

Upitnik se sastojao od jedanaest iskaza (svaki od njih se odnosio na po jednu sekundarnu hipotezu), a od studenata se tražilo da izraze svoj stepen slaganja sa datim iskazima na skali od jedan do pet. Pri tome, jedinicom je označeno apsolutno neslaganje, a peticom apsolutno slaganje. U svakom slučaju, studentima je sugerisano i da navedu u par rečenica zašto se slažu ili ne slažu sa iskazom. Rezultati evaluacije od strane prijavljenih dvadeset tri studenta su dati u sledećoj tabeli frekvencija (Tabela 3).

Tabela 3. Rezultati evaluacije JEFF alata sa studentima

Hip.	Kratak opis	Aps. se slažem	Ugl. se slažem	Nisam siguran	Ugl. se ne slažem	Aps. se ne slažem	Diskval.
(a)	Lako se inicijalizuje	20 (86.96%)	2 (8.69%)	1 (4.35%)	0	0	0
(b)	Lako se pravi tekstualni sadržaj	4 (17.39%)	9 (39.14%)	3 (13.04%)	2 (8.69%)	5 (21.74%)	0
(c)	Lako se prave slike kao sadržaj	19 (82.62%)	2 (8.69%)	2 (8.69%)	0	0	0
(d)	Lako se prave podaci kao sadržaj	2 (8.69%)	3 (13.04%)	10 (43.49%)	4 (17.39%)	4 (17.39%)	0
(e)	PDF izveštaj se lako pravi	21 (91.31%)	2 (8.69%)	0	0	0	0
(f)	XML izveštaj se lako pravi	20 (86.96%)	2 (8.69%)	1 (4.35%)	0	0	0
(g)	Tekstualni izveštaj se lako pravi	22 (95.65%)	1 (4.35%)	0	0	0	0
(h)	Lako se poziva iz baze znanja	17 (73.92%)	4 (17.39%)	2 (8.69%)	0	0	0
(i)	Tekstualni sadržaj se lako povezuje sa pravilima	15 (65.22%)	6 (26.09%)	2 (8.69%)	0	0	0

Hip.	Kratak opis	Aps. se slažem	Ugl. se slažem	Nisam siguran	Ugl. se ne slažem	Aps. se ne slažem	Diskval.
(j)	Slike kao sadržaj se lako povezuju sa pravilima	18 (78.26%)	4 (17.39%)	1 (4.35%)	0	0	0
(k)	Podaci kao sadržaj se lako povezuju sa pravilima	5 (21.74%)	15 (65.22%)	3 (13.04%)	0	0	0

Ako se pogledaju predstavljeni rezultati, može se zaključiti da je JEFF, generalno, jednostavan za korišćenje i integraciju sa Drools Expert BRE. JEFF se lako inicijalizuje, izbor željenog jezika je jednostavan, kao i izbor i pravljenje izveštaja u željenom formatu. Povezivanje sadržaja objašnjenja sa pravilima je takođe relativno lako, kao i pravljenje slika kao sadržaja objašnjenja. JEFF se lako poziva iz baze znanja u Drools Expert BRE, što je i očekivano jer se pozivi svode na pozivanje običnih Java metoda. Najkomplikovaniji zadaci su pravljenje (i prevođenje) tekstualnog sadržaja objašnjenja kao i pravljenje podataka kao sadržaja objašnjenja. Komentari studenata objašnjavaju zašto je tako.

Prva ozbiljnija prepreka pri pravljenju tekstualnog sadržaja objašnjenja je, prema navodima, imenovanje property fajlova u kojima ova objašnjenja treba da se nalaze. Studenti smatraju da je konvencija koja se koristi pri formiranju naziva fajla previše složena, a i sama činjenica da za svaki jezik treba uvek napraviti sva četiri fajla (prevodi za tekstualni deo objašnjenja, za naslove slika, za jedinice mera i dimenzije) bez obzira na to da li su svi ti prevodi potrebni ili ne.

Drugi problem pri pravljenju tekstualnog sadržaja objašnjenja je složenost mehanizma za umetanje dinamičkih delova u statički tekst. Zamerke se svode na to da je teško ispratiti koji dinamički deo ide gde u samom tekstu i onda taj redosled zapamtiti za

kasnije.

Konačno, sami property fajlovi podrazumevaju da se tekst objašnjenja piše u jednom redu. Za kratka objašnjenja je to odgovarajuće rešenje. Međutim, fajl postaje nepregledan kada su objašnjenja dugačka po nekoliko rečenica.

Problemi pri pravljenju podataka kao sadržaja objašnjenja se uglavnom svode na nemogućnost da se lako prepozna kada treba koristiti klase za dvodimenzionalne, a kad klase za trodimenzionalne podatke. Navedeno je i to da se predstavljanje reda podataka preko dvojki i tripleta značajno razlikuje od uobičajenog predstavljanja podataka putem kolona i redova u tabelama na koje su studenti navikli.

6.2.2 Komparativna analiza sa drugim mehanizmom za objašnjavanje

U cilju poređenja JEFF mehanizma za objašnjavanje sa nekim drugim sličnim alatom, izvršena je pretraga aktuelnih BRE i BRMS kao i drugih alata. Iako je već navedeno da veliki broj BRE i BRMS ima neke mogućnosti objašnjavanja, problem je u tome što se objašnjenje svodi na trag izvršavanja pravila i namenjeno je isključivo inženjerima baze znanja. Jedini alat koji ima slične mogućnosti objašnjavanja kao JEFF je Exsys Corvid, pa je njegov mehanizam za objašnjavanje upotrebljen za komparativnu analizu.

Kriterijumi koji su korišćeni u analizi su za svrhu imali da uporede ova dva mehanizma sa nekoliko aspekata:

- Slobode korišćenja
- Vrste objašnjenja koja se pružaju
- Korišćenih tehnika i metoda za objašnjavanje
- Vrste korisnika za koje su namenjena objašnjenja
- Tehnički aspekti korišćenja

Sloboda korišćenja alata podrazumeva, u ovom slučaju, utvrđivanje opcija licenciranja (da li se korišćenje plaća i da li je kod otvoren) kao i mogućnosti korišćenja sa drugim alatima. I jedan i drugi kriterijum utiču na to koje su najverovatnije potencijalne grupe

korisnika alata. Besplatni alati su, naravno, dostupni širim krugovima i profilima korisnika i nisu pretežno orijetnisani ka profesionalnim korisnicima kao alati čije korišćenje se plaća. Otvorenost koda tj. njegova dostupnost za izmenu i nadogradnju stvaraju mogućnost slobodnog prilagođavanja alata veoma specifičnim potrebama. Konačno, ako alat ne može da se koristi u kombinaciji sa drugim alatima, to može da ograniči njegovu upotrebljivost za određene situacije.

Vrste objašnjenja koja se pružaju određuju moć samog mehanizma za objašnjavanje, kao i to kada u toku sesije sa ES se može upotrebiti mehanizam za objašnjavanje - da li se može objasniti i proces upitivanja korisnika (objašnjenje ZAŠTO) ili samo proces zaključivanja (objašnjenja KAKO i STRATEGIJA).

Korišćene metode i tehnike za objašnjavanje uveliko određuju kako će krajnji rezultat objašnjavanja izgledati. Na primer, objašnjenja mogu ličiti na rečenice kontrolisanog jezika, ali mogu biti data i kao listing izvršenih pravila. To upućuje na zaključak da su neke metode i tehnike bolje prilagođene za korisnike sa višim nivoom tehničkog znanja, dok su ostale primerenije tehnički slabije obučenim korisnicima.

Već je više puta navedeno da postoje dve **vrste korisnika za koje su namenjena objašnjenja**: one sa visokim nivoom tehničkog znanja i one sa niskim nivoom tehničkog znanja. U prvom slučaju, to su uglavnom inženjeri znanja, dok se druga grupa odnosi na krajnje korisnike ES. U odnosu na to za koje vrste korisnika su podržana objašnjenja, mehanizam za objašnjavanje može da se koristi u različitim fazama razvojnog ciklusa ES: u toku razvoja i debug procesa (“tehnički” korisnici) ili u toku eksploatacije samog ES (“netehnički” korisnici).

Konačno, **tehnički aspekti korišćenja** obuhvataju razne kriterijume koji nisu mogli biti uvršćeni u prethodne grupe kriterijuma, a važni su sa aspekta komparativne analize i odnose se na neke tehničke mogućnosti alata. Na primer, ovde je utvrđeno da li alat ima GUI, koji su izlazni formati podržani, šta sve može da bude sadržaj objašnjenja (tekst, slike...), kao i da li objašnjenja mogu da se zaštite od neovlašćenog pristupa.

Rezultati komparativne analize mehanizma za objašnjavanje Exsys Corvid alata i JEFF mehanizma za objašnjavanje prema navedenim kriterijumima su dati u sledećoj tabeli

(Tabela 4).

Tabela 4. Rezultati komparativne analize mehanizma za objašnjavanje Exsys Corvid alata i JEFF mehanizma za objašnjavanje

Kriterijum	**JEFF**	**Mehanizam za objašnjavanje Exsys Corvid alata**
Licenca	Besplatna, otvoren kod	Plaća se, zatvoren kod
Korišćenje sa raznim BRE i BRMS	DA	NE
Objašnjenje KAKO	DA	DA
Objašnjenje ZAŠTO	NE	DA
Objašnjenje STRATEGIJA	DA	DA
Trag izvršavanja pravila	DA	DA
Učaureni tekst sa dinamičkim umetanjem sadržaja	DA	DA
Domensko znanje	NE	NE
Objašnjenja namenjena tehničkim korisnicima	DA	DA
Objašnjenja namenjena netehničkim korisnicima	DA	DA
Grafički interfejs	NE	DA
Internacionalizacija	DA	DA
Izlazni formati izveštaja	Tekst, XML, PDF	Tekst, RTF, PDF, XML, HTML
Stilsko formatiranje izveštaja	DA*	DA
Izveštaji u vidu fajlova	DA	DA
Izveštaji u vidu izlaza za browser	DA*	DA

Kriterijum	**JEFF**	**Mehanizam za objašnjavanje Exsys Corvid alata**
Vrsta sadržaja u objašnjenjima	Tekst, slike, tabele, grafici	Tekst, slike, tabele, animacije
Zaštita (bezbednost) izveštaja	NE	DA

JEFF i Exsys Corvid imaju potpuno drugačije licencne opcije. Kao što je navedeno, to dovodi do različitih potencijalnih grupa korisnika kao i načina korišćenja. JEFF je besplatan za korišćenje i otvorenog koda (GNU LGPL licenca [105]), pa je možda zbog toga primereniji za naučnu i edukativnu upotrebu - može se koristiti u bilo kojem obimu bez naknade i proširiti tj. prilagoditi na bilo koji način bez ikakve potrebne dozvole. Sa druge strane, to znači da podrška pri korišćenju ne postoji - osim korisničkog uputstva. Exsys Corvid nije besplatan za korišćenje i zatvorenog je koda, pa je ipak primereniji upotrebi u privredi gde se najviše traži da se primene već standardizovana rešenja i gde je podrška pri korišćenju dostupna. Potrebno je napomenuti i to da se Exsys Corvid može licencirati pod posebnim pogodnostima za prosvetne ustanove, ali da i dalje to nije besplatno.

Pošto je JEFF implementiran kao aplikacijski okvir u Javi, on može biti korišćen sa bilo kojim BRMS, BRE ili drugim alatom napisanim u Javi. Preciznije, jedini uslov za njegovo korišćenje je da taj drugi alat može da poziva Java metode - bilo u okviru samih pravila ili na neki drugi način. Mehanizam za objašnjavanje Exsys Corvid alata je, u ovom smislu, ograničen (licencom) pa se ne može koristiti sa drugim alatima.

Exsys Corvid podržava sve tri vrste objašnjenja, dok JEFF podržava samo objašnjenja KAKO i STRATEGIJA. JEFF ne podržava objašnjenje ZAŠTO zato što moderni BRE i BRMS alati uglavnom ne postavljaju pitanja krajnjem korisniku jer prikupljaju početne činjenice na drugi način, pa objašnjenje zašto je postavljeno određeno pitanje nije potrebno.

U Exsys Corvid-u, objašnjenje ZAŠTO se predstavlja zajedno sa postavljenim pitanjem.

Ono može biti u vidu linka ka nekoj HTML strani (unetog u sam tekst) ili u vidu teksta koji se može preuzeti po potrebi iz nekog fajla. Takođe, postoji i opcija korišćenja dodatnih HTML okvira koji prikazuju sadržaj objašnjenja svaki put kad se pitanje postavi.

Oba mehanizma za objašnjavanje koriste i trag izvršavanja pravila i učaureni tekst, pa se objašnjenja KAKO i STRATEGIJA mogu napraviti korišćenjem bilo koje od ove dve tehnike. U JEFF-u se ove dve tehnike čak mogu upotrebiti i zajedno u cilju formiranja jednog objašnjenja. Tehnika učaurenog teksta je proširena i dinamičkim unosom sadržaja, pa inženjer znanja može da odredi mesta u tekstu gde sadržaj može da se doda. Takođe, oba mehanizma mogu da pruže sve vrste objašnjenja za sve vrste korisnika - "tehničke" i "netehničke".

Exsys Corvid ima poseban deo (tzv. *Trace applet*) koji omogućava interaktivni prikaz traga izvršavanja pravila, ali se trag može dobiti i u vidu tekstualnog listinga. U oba slučaja, objašnjenje sadrži sve relevantne događaje: aktivacije pravila, promene činjenica itd. Ako je trag predstavljen preko *applet*-a, korisnik ima mogućnost da pretražuje sam trag u cilju pronalaženja željenog događaja.

U Exsys Corvid-u su sama pravila grupisana u *logičke blokove* (*logic block*) ili u *akcione blokove* (*action block* - pravila su predstavljena u vidu tabele). *Komandni blokovi* (*command block*) sadrže proceduralne komande i služe za usmeravanje procesa zaključivanja putem aktivacije i deaktivacije određenih logičkih blokova. Prema tome, objašnjenje STRATEGIJA se može naći u tragu izvršavanja pravila u vidu strategijskih odluka (događaja) donetih usled izvršavanja komandnih blokova. To znači da su objašnjenja KAKO i STRATEGIJA spojena u jedan trag izvršavanja pravila.

Trag izvršavanja pravila koji JEFF pruža je pojednostavljen i sadrži samo identifikator pravila i identifikator grupe kojoj pravilo pripada. Štaviše, trag prikazuje samo ona pravila kojima je pridružen određeni sadržaj objašnjenja. Objašnjenja KAKO i STRATEGIJA su i ovde objedinjena u jedan trag, ali su strategijske odluke označene STRATEGIC kontekstom samog dela objašnjenja.

Exsys Corvid-ov mehanizam za objašnjavanje ima opciju da skladišti učaureni tekst u

fajlu, bazi podataka ali i da ga direktno preuzme iz THEN dela pravila ili da postavi HTML link do njega. Ako je u pitanju objašnjenje KAKO ili STRATEGIJA, *kolekcije* (*collection*) omogućavaju postepeno formiranje objašnjenja tako što se pojedinačni delovi učaurenog teksta (ali i druge vrednosti) unose u njih. Objašnjenje STRATEGIJA nastaje tako što se učaureni tekst doda u kolekciju kada se izvrši komandni blok. Potrebno je napomenuti i to da se za objašnjavanje može koristiti više kolekcija istovremeno, ali i to da se ovako formirano objašnjenje ne može kombinovati sa tragom izvršavanja pravila u cilju dobijanja jedinstvenog objašnjenja.

JEFF omogućava skladištenje učaurenog teksta u property fajlovima, ali i unos iz THEN dela pravila. Pošto objašnjenja mogu da budu prilično dugačka (i po par pasusa), zamišljeno je da se definicije pravila odvoje od sadržaja objašnjenja da se ne bi stvarala gužva.

Exsys Corvid je kompletno okruženje za razvoj ES, pa ima i svoj GUI. Pravljenje objašnjenja se, naravno, vrši putem ovog GUI-ja. JEFF je aplikacijski okvir u Javi i ideja je da se upotrebljava sa više različitih BRE i BRMS alata, pa je izrada jednistvenog GUI-ja koji bi odgovarao svim (ili makar većini) alata, veoma teška ako ne i neizvodljiva. Sa druge strane, najveći broj aktivnosti u JEFF-u se može postići uz pomoć običnih ili Java tekstualnih editora (koje okruženja za razvoj ES već imaju) i ne zahtevaju specijalizovan GUI: pravljenje i izmena property fajlova, pisanje Java komandi za inicijalizaciju JEFF-a i pravljenje izveštaja, pisanje komandi u THEN delu pravila kojima se poziva JEFF itd.

I JEFF i Exsys Corvid mogu da prave objašnjenja na više jezika. U JEFF-u, prevodi se obavezno nalaze u alternativnim property fajlovima, dok Exsys Corvid može da ih skladišti u raznim vrstama fajlova, ali i da ih unese direktno u samu bazu znanja.

Izveštaji koji mogu da se naprave u JEFF-u mogu biti u XML, PDF ili formi običnog teksta. Osim ova tri formata, Exsys Corvid podržava i RTF (Rich Text Format) i HTML a ima i veliki broj opcija za formatiranje izgleda izveštaja (stilovi, fontovi, boje, podešavanja strane itd.). U ovom trenutku, JEFF poseduje neke od opcija za formatiranje, ali se one odnose isključivo na PDF izveštaje.

I jedan i drugi mehanizam za objašnjavanje mogu da sačuvaju objašnjenje kao fajl ili da ga pošalju browser-u, ali u JEFF-u je za ovo poslednje potrebna određena količina dodatnog programiranja.

Objašnjenja iz oba alata mogu da sadrže tekst, slike i podatke ali samo Exsys Corvid podržava i animacije. Takođe, potrebno je navesti i to da JEFF omogućava prikaz podataka i u vidu grafika, dok kod Exsys Corvid-a podaci mogu da se prikažu samo preko tabela.

Konačno, Exsys Corvid može da ograniči pristup nekom objašnjenju na osnovu IP adrese. Ovo, naravno, funkcioniše samo kada je aplikacija distribuirana. JEFF, u ovom trenutku, nema nikakve opcije za ograničavanje pristupa objašnjenjima.

6.3 Evaluacija prototipa za automatizovano tumačenje podataka o profitu

Jedan od glavnih ciljeva PT alata je bio da se pokuša sa njegovom upotrebom u nekoj realnoj situaciji i to radi sticanja povratne informacije o njegovoj funkcionalnosti. Sa obzirom na to da je u pitanju relativno jednostavna aplikacija, odlučeno je da se isproba kao pomoćno nastavno sredstvo za studente menadžmenta. Na Fakultetu organizacionih nauka se, na smeru za menadžment, izvode predavanja i vežbe iz predmeta “ekonomika preduzeća”, pa je odlučeno da se PT upotrebi u nastavi iz ovog predmeta. Osmišljeno je da studenti budu ispitanici, i da prijavljivanje za evaluaciju bude dobrovoljno.

Osnovni ciljevi ove evaluacije su bili da se utvrdi da li je PT:

- Funkcionalan
- Jednostavan za upotrebu
- Da li su objašnjenja u izveštajima kvalitetna
- Da li obezbeđuje zanimljiviji način za učenje materije (nego putem knjiga, zbirki, itd)
- Da li obezbeđuje brži način za učenje materije (nego putem knjiga, zbirki, itd)

Osim toga, prikupljeni su podaci i o tome i kakva su prethodna iskustva studenti imali sa sličnim edukativnim softverima, kao i to gde su odlučili da izvrše evaluaciju - kod

kuće, u toku vežbi ili na obe lokacije. Ispitan je i stav studenta prema tome da li bi voleli da se edukativni softver češće koristi u redovnom obrazovanju, kao i to da li bi ovakav edukativni softver (sličan PT aplikaciji) koristili ponovo.

Svaki od osnovnih ciljeva je dodatno razrađen preko nekoliko podciljeva, a oni su pretvoreni u iskaze čija istinitost (tj. stepen slaganja ispitanika) se proveravala u toku evaluacije:

- Program se jednostavno pokreće.
- GUI je lako razumljiv.
- GUI ima prijatan dizajn.
- Navigacija kroz menije je jednostavna i intuitivna.
- Svaka ekranska forma se brzo učitava.
- Zadovoljan sam povratnim porukama u slučaju greške.
- Program je koristan.
- Program je jednostavan za korišćenje.
- Objašnjenja u izveštajima su lako razumljiva.
- Objašnjenja u izveštajima su obuhvatna.
- Objašnjenja u izveštajima su korisna.
- Grafici iz izveštaja su korisni u toku učenja.
- PT doprinosi povećanju nivoa mog znanja.
- Korišćenje PT aplikacije u toku učenja mi je bilo zanimljivo.
- Smatram da je učenje preko PT aplikacije brže nego korišćenjem klasičnih materijala (knjiga i zbirki)

Predmet “ekonomika preduzeća” je odslušalo 336 studenata, od kojih se 148 dobrovoljno prijavilo da učestvuje u evaluaciji. Svih 148 prijavljenih je zaista i završilo evaluaciju. Samo istraživanje je sprovedeno na sledeći način.

Pre same evaluacije, studenti su slušali uvodne lekcije o profitu, prihodima, troškovima i opštim merama za ocenu investicija. Cilj ovih lekcija je bio da steknu osnovno razumevanje navedenih pokazatelja i da nauče kako se izračunavaju, ali bez ulaženja u detalje u vezi sa tim kako se njihove vrednosti tumače i kako utiču jedni na druge.

Efekat je taj da su studenti u evaluaciju ušli sa približno istim nivoom znanja tj. spremni da ga nadograde uz pomoć PT aplikacije.

Evaluacija je izvedena u nekoliko koraka. Prvo, u toku nastave, studenti su obučeni da koriste PT aplikaciju i dati su im svi potrebni materijali: kopija PT aplikacije, predefinisani primeri poslovnih podataka za PT aplikaciju i korisničko uputstvo. Ovih osamnaest predefinisanih primera koji su isporučeni uz PT nisu bili obrađeni u toku obuke, već su bili rangirani po složenosti i služili su kao osnova za kasnije (samostalno) učenje. Obuka je vršena u salama računarskog centra i svaki student je sedeo za svojim računarom.

Nakon toga, od studenata se tražilo da prođu kroz sve predefinisane PT primere, da probaju da protumače podatke na osnovu izveštaja sa podacima i da vide da li su im tumačenja tačna. Sve se, zapravo, svodilo na to da studenti odgovore na pitanja iz PT upitnika, i provere da li se njihovi odgovori poklapaju sa tumačenjem iz izveštaja koji sadrži i podatke i informacije. Cilj ovog dela zadatka je da, na osnovu konkretnih primera poslovnih situacija, uz pomoć PT aplikacije nauče kako se izabrani pokazatelji tumače i kako utiču jedni na druge. Svaki od osamnaest predefinisanih primera je služio da simulira po jednu specifičnu poslovnu situaciju.

Kada su to završili, od studenata se tražilo da naprave sopstvene primere poslovnih podataka u PT-u i to tako da ti primeri odslikavaju neke konkretne poslovne situacije. Na primer, bilo je potrebno da naprave skup podataka koji odslikava zrelo preduzeće koje ne posluje pozitivno jer je kriza na tržištu u toj grani privrede. Cilj ovog drugog dela zadatka je bio da studenti pokušaju da upotrebe PT aplikaciju i svoje tek stečeno znanje za simulaciju poslovnih situacija - i onih kojii su se od njih tražili, ali i nekih po sopstvenom izboru. Smatralo se da, na ovaj način, stečeno znanje može biti utvrđeno jer se upotrebljava na drugačiji način - ne za tumačenje konkretne situacije, već da se izmisle podaci koji bi poslovnu situaciju činili onakvom kakva se traži.

Kao što je napomenuto, studenti su mogli da biraju da li će ovaj zadatak završiti kod kuće, na laboratorijskim vežbama (organizovanim u računarskom centru) ili na oba mesta. Svaki student je morao samostalno da završi celokupan zadatak počev od instalacije samog alata do čuvanja napravljenih primera. Potrebno je napomenuti i to da

su studenti dobili uputstvo da ne koriste klasične materijale za učenje (knjige i zbirke) pri izradi zadatka.

Po završetku zadatka, znanje svakog studenta je ocenjeno putem kratkog testa, a pregledani su i svi primeri koje su napravili i to u cilju utvrđivanja da je zadatak zaista i izvršen. Rezultati evaluacije stečenog znanja studenata nisu prikazani u ovom poglavlju o evaluaciji jer su deo posebnog istraživanja.

Konačno, studenti su morali da popune upitnik koji se ticao njihovih utisaka i iskustava u toku korišćenja PT aplikacije. Sam upitnik se sastojao od dvadeset pitanja od kojih su se prva četiri odnosila na prethodna iskustva sa sličnim edukativnim softverima a sva ostala na već navedene ciljeve evaluacije. Od studenata se takođe tražilo da, svojim rečima, napišu sve prednosti i mane PT koje su uočili. Razlog zašto je ostavljeno mesto za navođenje prednosti i mana je taj što se struktuiranim pitanjima (pitanjima sa ponuđenim odgovorima) može potvrditi samo ono što se može predvideti, a nastojalo se da se otkriju i druga opažanja ispitanika u radu sa PT aplikacijom.

Veoma je važno napomenuti i to da je PT aplikacija koju su studenti koristili takva da je na engleskom jeziku. To znači da su GUI, svi izveštaji i PT upitnik na engleskom jeziku. Pošto studentima to nije maternji jezik, diskutabilno je da li je to uticalo na rezultate evaluacije i koliko. Međutim, sa obzirom na to da se ovaj jezik uči u okviru obavezne nastave u osnovnim i srednjim školama, kao i da su svi ispitani studenti u prethodnom semestru položili ispit iz poslovnog engleskog jezika, smatralo se da to ipak neće uticati na rezultate testiranja. To je bilo potvrđeno nakon evaluacije jer je samo jedanaest studenata navelo da su imali probleme da razumeju izveštaje zbog toga što su napisani na engleskom jeziku (naveli su to kao manu PT).

Rezultati evaluacije koji se tiču mesta izvršenja zadatka, kao i prethodnih iskustava sa edukatovnim softverima se mogu videti u tabeli (Tabela 5). Najveći broj studenata je odlučio da zadatak izvrši u kompjuterskom centru, dok je manji broj njih odlučio da to uradi kod kuće ili na oba mesta. Ovo se može objasniti time što je, u toku rada u računarskom centru, uvek bio neko prisutan od tehničkog osoblja pa je pomoć bila na raspolaganju u svakom trenutku.

Što se tiče prethodnih iskustava, najveći broj studenata nikad nije imalo prilike da se susretne sa sličnim softverom u toku redovnog obrazovanja. Pod redovnim obrazovanjem se ovde smatralo obrazovanje stečeno u toku osnovne škole, srednje škole i fakulteta. Oni koji jesu probali edukativni softver (u bilo kojoj vrsti obrazovanja a ne samo redovnom), su pretežno imali neutralna ili pozitivna iskustva. U komentarima, studenti su uglavnom navodili da je softver koji su prethodno koristili bio softver za obuku vožnje, ili za učenje jezika. Konačno, većina ispitanih studenata bi volela da se edukativni softver koristi češće u toku školskog obrazovanja.

Tabela 5. Tabela frekvencija sa podacima o mestu izvršenja zadatka i prethodnim iskustvima

Iskaz (skraćen)	**Odgovori**					
Završio/la sam zadatak:	Kod kuće 21 (14.2%)	U rač. centru 98 (66.2%)	Na oba mesta 29 (19.6%)	Nisam završio/la 0 (0.0%)		
Već sam koristio/la edukativni softver u toku redovnog obrazovanja.	Da 49 (33.1%)	Ne 99 (66.9%)				
Moja prethodna iskustva sa edukativnim softverom iz bilo koje vrste obrazovanja su:	Veoma pozitivna 4 (2.7%)	Pozitivna 31 (20.9%)	Neutralna 35 (23.6%)	Negativna 2 (1.4%)	Veoma negativna 0 (0.0%)	Nemam prethodno iskustvo 76 (51.4%)

Iskaz (skraćen)	Odgovori		
Voleo/la bih da se edukativni softver više upotrebljava u okviru školskog obrazovanja.	Da 122 (82.4%)	Ne 6 (4.1%)	Ne znam 20 (13.5%)

U narednoj tabeli (Tabela 6) se mogu videti rezultati dela evaluacije koji se tiče funkcionalnosti i jednostavnosti korišćenja PT aplikacije. Ova dva aspekta su ispitana sa stanovišta jednostavnosti samog pokretanja programa, dizajna i razumljivosti GUI-ja, brzine rada programa, mogućnosti da se jednostavno postigne željena akcija kao i opšteg utiska o jednostavnosti u radu.

Tabela 6. Tabela frekvencija sa podacima o jednostavnosti korišćenja

Iskaz (skraćen)	Odgovori					
Program se jednostavno pokreće.	Potpuno se slažem 31 (20.9%)	Slažem se 85 (57.4%)	Nisam siguran/na 6 (4.1%)	Ne slažem se 21 (14.2%)	Potpuno se ne slažem 5 (3.4%)	Nedostaje 0 (0.0%)
GUI je lako razumljiv.	Potpuno se slažem 37 (25.0%)	Slažem se 93 (62.8%)	Nisam siguran/na 10 (6.8%)	Ne slažem se 8 (5.4%)	Potpuno se ne slažem 0 (0.0%)	Nedostaje 0 (0.0%)
GUI ima prijatan dizajn.	Potpuno se slažem 20 (13.5%)	Slažem se 85 (57.4%)	Nisam siguran/na 9 (6.1%)	Ne slažem se 32 (21.6%)	Potpuno se ne slažem 2 (1.4%)	Nedostaje 0 (0.0%)

Iskaz (skraćen)	**Odgovori**					
Navigacija kroz menije je jednostavna i intuitivna.	Potpuno se slažem 33 (22.3%)	Slažem se 93 (62.8%)	Nisam siguran/na 18 (12.2%)	Ne slažem se 4 (2.7%)	Potpuno se ne slažem 0 (0.0%)	Nedostaje 0 (0.0%)
Svaka ekranska forma se brzo učitava.	Da 107 (72.3%)	Ne 40 (27.0%)				Nedostaje 1 (0.7%)
Zadovoljan sam povratnim porukama u slučaju greške.	Da 60 (40.5%)	Delimično 82 (55.4%)	Ne 4 (2.7%)			Nedostaje 2 (1.4%)
Program je koristan.	Potpuno se slažem 40 (27.0%)	Slažem se 101 (68.2%)	Nisam siguran/na 7 (4.7%)	Ne slažem se 0 (0.0%)	Potpuno se ne slažem 0 (0.0%)	Nedostaje 0 (0.0%)
Program je jednostavan za korišćenje.	Potpuno se slažem 36 (24.3%)	Slažem se 100 (67.6%)	Nisam siguran/na 8 (5.4%)	Ne slažem se 4 (2.7%)	Potpuno se ne slažem 0 (0.0%)	Nedostaje 0 (0.0%)

Kao što se može videti, najveći broj studenata se slaže da se program jednostavno pokreće. Međutim, smatralo se da će odgovori ukazivati na veći stepen slaganja sa ovim iskazom, ali to nije bio slučaj. Razlog je taj što, iako je PT jednostavna Java desktop aplikacija (koja se ne instalira i može da se pokrene klikom miša), jedan broj studenata nije imao instaliranu Java virtuelnu mašinu na svojim računarima pa nije mogao da je pokrene. Zbog toga su morali da prođu kroz proces pronalaženja problema i pre prvog pokretanja programa. Sa obzirom na to da su u pitanju studenti menadžmenta, nije se uzelo u obzir da oni znanje o programskom jeziku Java nisu dobili u toku redovnih studija, pa nisu ni znali za ovaj uslov. Uzrok problema je, zapravo, otkriven kada su studenti napisali ovo kao jednu od mana PT.

Najveći broj ispitanih studenata smatra da je GUI PT aplikacije relativno razumljiv i da je navigacija kroz menije jednostavna i intuitivna. Međutim, iako je dizajn GUI-ja ocenjen uglavnom kao prijatan za rad, najčešće spominjani negativni aspekt PT je upravo zastarelost i prevelika jednostavnost GUI dizajna. Problem je, prema tvrdnjama, u tome što dizajn nije estetski privlačan, i veoma je ograničen u smislu broja načina na koji se može obaviti neka aktivnost, pa podseća na amaterski pokušaj pre nego na profesionalni rad. Još jedna stvar koja je navedena je bilo očekivanje da se programu može pristupiti preko browser-a, što je uobičajen i očekivan način rada za veliki broj poslovnih aplikacija. To, nažalost, ovde nije bio slučaj.

PT ekranske forme se, prema rezultatima evaluacije, učitavaju brzo. Ipak, i tu postoji mesta za poboljšanje. Profilisanjem brzine rada aplikacije je utvrđeno da su dva procesorski najzahtevnija procesa inicijalizacija Drools baze znanja i Jasper Reports alata za izveštavanje. Baza znanja se sporo inicijalizuje zato što se prevodi iz tekstualne forme (Drools DRL fajl), sintaksno proverava i tek onda kompajlira u izvršivu bazu znanja (Java objekti). Alat za izveštavanje se takođe sporije inicijalizuje zbog toga što se predefinisani šabloni za izveštaje kompajliraju iz fajla sa šablonom (XML fajl) u Java objekte koji su izvršivi. Sve to znači da se prvo pokretanje PT aplikacije uvek čini sporo (jer se tad izvršavaju ova dva procesa inicijalizacije), dok je svako naredno pokretanje (ako aplikacija nije u međuvremenu gašena) mnogo brže. Već je navedeno da je početna PT ekranska forma napravljena zbog toga da informiše korisnike, ali i da služi kao “paravan” dok se Drools baza znanja inicijalizuje da korisnici ne bi u kasnijem radu morali da čekaju. Inicijalizacija Jasper Reports alata za izveštavanje nije ovako rešena, već se vrši tek kad se izabere PT komanda za pravljenje izveštaja.

Jedan od aspekata PT aplikacije koji mora biti adresiran u najskorije vreme je obaveštavanje korisnika povratnim porukama u slučaju pojavljivanja greške. Mišljenje je da su studenti samo delimično zadovoljni povratnim porukama u slučaju kad naprave grešku zato što se najveći broj takvih poruka odnosi na proveru ograničenja ulaznih podataka (npr. da li je neko uneo negativan broj za godinu) i ne obuhvataju ostale vrste grešaka. To znači da ako se, iz nekog razloga, ne može inicijalizovati baza znanja, korisnik neće dobiti adekvatnu poruku o tome već će program samo prestati da radi.

Greška koju su ispitanici često ponavljali je bila ta da su pokušali da naprave nove PDF izveštaje uz pomoć PT aplikacije, a da su prethodne izveštaje i dalje držali otvorene za pregledanje. To je dovodilo do nemogućnosti da se novi fajlovi kreiraju (jer istoimeni fajlovi već postoje, a zaključani su za izmene jer su trenutno otvoreni), a PT aplikacija nije obaveštavala o uzroku problema ni na koji način.

Na kraju, studenti smatraju da je PT koristan i, generalno, jednostavan za upotrebu. Najčešće pominjana prednost PT aplikacije je upravo ta da je jako korisna jer omogućava manje ili nedovoljno obrazovanim korisnicima da steknu detaljan uvid u to kako preduzeće posluje.

Procena kvaliteta objašnjenja i grafika u izveštajima se može videti u sledećoj tabeli (Tabela 7).

Tabela 7. Tabela frekvencija sa podacima o kvalitetu objašnjenja i grafika u izveštajima

Iskaz (skraćen)	**Odgovori**			
Objašnjenja u izveštajima su lako razumljiva.	Da 104 (70.3%)	Delimično 43 (29.1%)	Ne 0 (0.0%)	Nedostaje 1 (0.7%)
Objašnjenja u izveštajima su obuhvatna.	Da 92 (62.2%)	Delimično 54 (36.5%)	Ne 1 (0.7%)	Nedostaje 1 (0.7%)
Objašnjenja u izveštajima su korisna.	Da 108 (73.0%)	Delimično 36 (24.3%)	Ne 3 (2.0%)	Nedostaje 1 (0.7%)
Grafici iz izveštaja su korisni u toku učenja.	Da 117 (79.1%)	Delimično 28 (18.9%)	Ne 1 (0.7%)	Nedostaje 2 (1.4%)

Jasno je da se smatra da su objašnjenja lako razumljiva i korisna. Grafici koji se nalaze u oba izveštaja (sa i bez informacija) se takođe smatraju korisnim, kad je učenje u pitanju. Ali, stvari ne stoje tako dobro kada je u pitanju obuhvatnost objašnjenja. Veliki broj studenata smatra objašnjenja obuhvatnim, ali jedan veliki broj smatra da su samo delimično obuhvatna. Mišljenje je da bi objašnjenja, za studente na ovom nivou znanja, ipak trebalo da sadrže još detaljnije informacije o tome kako su zaključci doneti, da obuhvataju neke definicije, teoriju, pa čak i neke kratke primere koji objašnjavaju ceo

proces zaključivanja. Još jedna mana PT koja je navedena nekoliko puta, a u vezi je sa objašnjenjima, je i ta da objašnjenja nisu interaktivna tj. nije moguće dobiti dodatna objašnjenja ukoliko to korisnik zahteva.

Ukupni utisci stečeni korišćenjem PT aplikacije se mogu videti u narednoj tabeli (Tabela 8). Najveći broj ispitanih studenata veruje da PT donekle doprinosi njihovom nivou znanja. Osnovna ideja je bila da se PT upotrebi kao dodatno nastavno sredstvo, pa je ovaj rezultat očekivan. Čitavo iskustvo učenja na ovaj način je ocenjeno kao delimično interesantno, a studenti smatraju i da se materija brže savladava u odnosu na učenje iz knjiga i zbirki. Kada su upitani da li bi ponovo korisitili ovakav edukativni softver, najveći broj studenata je odgovorio pozitivno.

Naravno, ovde je samo izmeren subjektivan osećaj ispitanika u vezi sa tim da li se učenje putem PT aplikacije izvodi brže i da li doprinosi njihovom nivou znanja. Da bi se zaista ispitalo da li PT doprinosi nivou znanja studenata i da li zaista mogu da brže savladaju materiju i u kojoj meri, potrebno je dodatno istraživanje u vidu eksperimenta, ali će o ovome biti više reči u narednom poglavlju.

Tabela 8. Tabela frekvencija sa podacima o opštim utiscima o korišćenju PT aplikacije

Iskaz (skraćen)	**Odgovori**			
PT doprinosi povećanju nivoa mog znanja.	U potpunosti 33 (22.3%)	Delimično se slažem 112 (75.7%)	Ne doprinosi 2 (1.4%)	Nedostaje 1 (0.7%)
Korišćenje PT aplikacije u toku učenja mi je bilo:	Veoma zanimljivo 39 (26.4%)	Donekle zanimljivo 98 (66.2%)	Nije zanimljivo 10 (6.8%)	Nedostaje 1 (0.7%)

Iskaz (skraćen)	Odgovori					
Smatram da je učenje preko PT aplikacije brže nego korišćenjem klasičnih materijala (knjiga i zbirki)	Potpuno se slažem 34 (23.0%)	Slažem se 68 (45.9%)	Nisam siguran/na 34 (23.0%)	Ne slažem se 11 (7.4%)	Potpuno se ne slažem 0 (0.0%)	Nedostaje 1 (0.7%)
Koristio/la bih ovakav edukativni softver ponovo.	Da 114 (77.0%)	Ne 5 (3.4%)	Ne znam 28 (18.9%)			Nedostaje 1 (0.7%)

6.4 Diskusija o razvijenim aplikacijama i budući pravci razvoja

U ovom poglavlju je prikazana diskusija i razmotreni su budući pravci razvoja JEFF mehanizma za objašnjavanje i PT aplikacije.

JEFF

Što se tiče JEFF mehanizma za objašnjavanje, trenutni napori su usmereni ka tome da se otklone neki njegovi nedostaci koji su uočeni u toku evaluacije. Na dugoročnom planu je zamišljeno da se JEFF što više koristi u realnim situacijama i to u cilju sticanja praktičnog iskustva, ali i daljeg razvoja do nivoa u kojem će moći da se koristi kao profesionalni alat.

Konkretni ciljevi koji su postavljeni pred JEFF mehanizam za objašnjavanje, a tiču se budućeg razvoja su sledeći:

- Pojednostavljenje načina na koji se pravi tekstualni sadržaj objašnjenja
- Pojednostavljenje načina na koji se prave podaci kao sadržaj objašnjenja
- Izrada GUI-ja
- Proširenje mogućnosti formatiranja izveštaja
- Uvođenje ograničenja pristupa objašnjenjima
- Dodavanje novih vrsta izlaznih formata izveštaja

- Dodavanje novih vrsta objašnjenja
- Dodavanje novih vrsta sadržaja u objašnjenja
- Dodavanje mogućnosti pravljenja interaktivnih objašnjenja

Prvo, potrebno je da se **pojednostavi način na koji se pravi sadržaj u vidu teksta i sadržaj u vidu podataka**. To može, u ovom trenutku, da se uradi na dva načina. Jedan od njih je da se sam proces pojednostavi u smislu pravljenja jednostavnijeg modela. To bi podrazumevalo, na primer, pravljenje određenih novih klasa koje bi služile kao fasade (pattern Facade) i olakšavale upotrebu već postojećih klasa. Drugi način je kreiranje GUI-ja.

Izrada GUI-ja za JEFF bi zasigurno olakšala sve zadatke koje inženjer znanja radi u ovom alatu. Već je napomenuto da je teško izvesti tako nešto a da se taj GUI kasnije koristi jednostavno sa svim mogućim BRE i BRMS alatima. U tom smislu, trenutno se razmatraju dve opcije:

- Pravljenje posebnog GUI-ja korišćenjem standardnih Java biblioteka kao samostalne aplikacije.
- Pravljenje GUI-ja koji bi se uklopio u neko od već postojećih okruženja i bio njegov deo.

U prvom slučaju, prednost je ta što bi se GUI koristio takav kakav je nezavisno od BRE ili BRMS sa kojim se trenutno radi. Mana je ta što bi GUI bio potpuno odvojen od GUI-ja BRE ili BRMS, pa se osećaj integrisanosti i povezanosti dva alata ne bi mogao ostvariti. Inženjer znanja bi morao da prvo napravi ES u alatu po izboru, pa tek onda da pokrene JEFF GUI da napravo objašnjenja. Kasnije, u toku održavanja, opet bi moralo da se radi sa dva nepovezana interfejsa i da se vodi računa o nekim aspektima koji ne bi bili važni da su alati integrisani (npr. o tome koji JEFF fajlovi se gde nalaze u okviru softverskog projekta ES).

U drugom slučaju, težilo bi se ka integrisanosti sa već postojećim GUI-jem, i to je glavna prednost. Inženjer znanja bi, prema tome, imao/la osećaj da mu je samo dodata nova funkcionalnost u BRE/BRMS alat koji već koristi a specijalizovani editori za JEFF

fajlove i dodatni meniji bi mu/joj pomogli da ne mora da brine o stvarima kao što su nazivi property fajlova isl. Glavna mana je to što se jedinstveni GUI koji može da se uklopi u sve alate ne bi mogao napraviti. Racionalizacijom ovog uslova se dolazi do zaključka da je nešto na tom planu ipak moguće uraditi ako se skup podržanih alata ograniči na Drools, JESS i JRules. Naime, sva tri navedena alata imaju GUI urađen u Eclipse okruženju, pa bi JEFF GUI mogao biti implementiran kao Eclipse plugin koji bi radio na isti način u sva tri slučaja.

Proširenje mogućnosti formatiranja izveštaja bi se, zapravo, odnosilo na izlazne formate za koje to ima smisla. Cilj je da se u JEFF alatu ima sve što je potrebno da inženjer znanja može da napravi izveštaj koji bi mu zadovoljio najveći broj potreba u smislu izgleda, stila itd. a da ne mora da upotrebljava neke specijalizovane alate za izveštavanje. U ovom trenutku, JEFF podržava samo neke opcije formatiranja za PDF izveštaje koji se tiču veličine strane, margina, stila paragrafa isl. koje bi bilo potrebno proširiti. Uočeno je da se opcije koje fnedostaju (a bile su potrebne u mnogim slučajevima) tiču: podele teksta na kolone, podešavanja veličine i pozicije slika, izbora veličine i pozicije grafika, unosa logotipova i žigova na strane i unosa broja strane i datuma. Na sreću, iText aplikacijski okvir koji JEFF koristi da pravi PDF izveštaje već ima ove opcije, pa ih je potrebno samo iskoristiti.

Uvođenje ograničenja pristupa objašnjenjima je veoma važan aspekt korišćenja koji mora biti adekvatno obrađen pre nego što JEFF može da počne da se upotrebljava kao profesionalni alat. JEFF model specificira da svako objašnjenje može da ima vlasnika (atribut *owner*), ali nikakvi bezbednostni mehanizmi koji bi onemogućavali nedozvoljen pristup nisu u ovom trenutku implementirani. Sam programski jezik Java poseduje i standardizovane biblioteke za enkripciju podataka, pa će najverovatnije one biti upotrebljene u ovu svrhu.

Dodavanje novih vrsta izlaznih formata izveštaja je takođe jedan od daljih pravaca razvoja JEFF alata. U ovom trenutku se vrše napori da se podrži i HTML kao izlazni format i to u cilju boljeg iskorišćavanja opcije JEFF-a da objašnjenje prosledi direktno browser-u u vidu izlaznog toka. Sa obzirom na to da su softverski patterni upotrebljeni u toku projektovanja JEFF-a, dodavanje podrške za novi izlazni format nije previše

komplikovano i ne remeti već postojeće delove aplikacijskog okvira. U budućnosti, namera je da se podrže i neki od otvorenih standarda za dokumenta kao: Open Office skup standarda, LateX itd.

Dodavanje novih vrsta objašnjenja je bitno da bi JEFF mogao da se koristi kao profesionalni alat. Ovde se prvenstveno misli na objašnjenje ZAŠTO. Ovo objašnjenje se dobija u trenutku kada ES postavlja pitanja korisniku (kao kratak odgovor), dok se objašnjenja KAKO i STRATEGIJA dobijaju po završetku procesa odlučivanja (kao detaljni izveštaji). Takođe, objašnjenje ZAŠTO se najčešće ne sastoji iz više delova koji objašnjavaju ceo proces rezonovanja već je jednokratno - odnosi se na jedno pitanje. Sa druge strane, tehnika učaurenog teksta se takođe može upotrebiti da bi se dobilo ovo objašnjenje. Kada se sve uzme u obzir, deo JEFF-a koji bi podržavao objašnjenje ZAŠTO bi se umnogome razlikovao od postojećeg domenskog modela, ali bi, najverovatnije, koristio neke od već postojećih mehanizama i klasa u JEFF-u.

Dodavanje novih vrsta sadržaja u objašnjenja je opcija koja će najverovatnije biti omogućena u budućnosti. Jednu grupu čine dinamički multimedijalni sadržaji kao zvuk, animacije, filmovi i slično. Exsys Corvid već ima ove opcije, a sadržaji su zapravo Adobe Flash prezentacije. Druga vrsta sadržaja koja bi bila veoma interesantna i potencijalni kandidat za dodavanje u objašnjenja su hiper linkovi. Na ovaj način bi se mogla ostvariti povezanost objašnjenja sa nekim udaljenim sadržajima na Internetu kao i, eventualno, povezanost sa resursima u okviru nekog internog informacionog sistema (ukoliko je distribuiran).

Dodavanje mogućnosti pravljenja interaktivnih objašnjenja je veoma važno jer su, u ovom trenutku, objašnjenja potpuno statička. Trenutno se razmatra nekoliko načina na koje se može postići interaktivnost i to da li će se primeniti neki od njih ili više u kombinaciji:

- Uvođenje hiper linkova.
- Uvođenje internih linkova
- Omogućavanje pretraživanja

Prvi način je uvođenje hiper linkova kao sadržaja objašnjenja, i odgovarajućih upita

(bolje reći naziva upita npr. “Detaljnije”) na koje bi se otvorili sadržaji tih linkova. Još uvek nije sasvim jasno da li je bolje da ovi linkovi budu uneti u sadržaj dela objašnjenja (direktno u sam tekst objašnjenja tako da neka reč zapravo predstavlja link), da predstavljaju poseban deo objašnjenja (npr. posle paragrafa teksta objašnjenja sledi zaseban paragraf sa linkom na kojem piše “Detaljnije”) ili da se omoguće obe opcije. Naravno, naslov na samom linku bi bio proizvoljan čime bi se mogla stvoriti iluzija da se može dodati odgovor za bilo koji upit ukoliko je unapred zamišljen i implementiran. U svakom slučaju, objašnjenja bi postala interaktivnija jer bi se moglo, po želji, ulaziti u druge (povezane) sadržaje i dobiti detaljnija objašnjenja određenih pojmova isl.

Uvođenje internih linkova je u bliskoj vezi sa prethodnom opcijom, samo destinacije tih linkova ne bi bile van objašnjenja već unutar objašnjenja. Drugim rečima, ostvarila bi se mogućnost da se delovi objašnjenja međusobno povežu, pa bi korisnik mogao da preko nekog linka ostvari uvid tj. pređe na neki drugi deo objašnjenja. Na primer, u slučaju PT aplikacije, bilo bi moguće da se svaka stavka sižea (na početku izveštaja sa podacima i informacijama) poveže sa odgovarajućim delom izveštaja koji sadrži detalje u vezi sa tom stavkom.

I uvođenje mogućnosti pretraživanja objašnjenja bi moglo da bude implementirno u budućnosti kao sredstvo za povećanje interaktivnosti. Pri tome, ne misli se na jednostavno pretraživanje preko ključnih reči jer je to već moguće - svaki PDF ili tekstualni dokument se može pretraživati na ovaj način veoma lako. Pretraživanje bi obuhvatalo stvaranje jednostavnih upita i pregled rezultata na osnovu nekih metapodataka u okviru samih objašnjenja: vrste dela objašnjenja, konteksta, pridruženih tagova isl. U tom smislu, bilo bi moguće, na primer, izlistati samo one delove objašnjenja koji predstavljaju upozorenja, sadrže sliku i imaju pridružen tag “proizvod”.

Važno je napomenuti to da nije moguće implementirati sve opcije za interaktivnost u svim izlaznim formatima izveštaja. Na primer, PDF fajlovi mogu da sadrže i interne i hiper linkove, ali ne podržavaju predloženi oblik pretraživanja. Tekstualni fajlovi ne podržavaju nikakvu vrstu interaktivnosti u ovom smislu, a XML fajlovi omogućavaju sve, samo je potrebno da postoji odgovarajući alat koji može da izvršava XSLT transformacije iil XPath upite. Na osnovu svega, može se zaključiti da će možda biti

implementiran neki interni format ili neki interni JEFF alat koji će omogućavati sve vrste pretraživanja, a da će se u postojećim vrstama izveštaja implementirati one vrste interaktivnosti koje su moguće.

Što se tiče korišćenja JEFF alata u realnim situacijama, treba navesti da je JEFF odskora predstavlja pomoćno nastavno sredstvo na predmetu "inteligenti sistemi" na Fakultetu organizacionih nauka. Na ovaj način, studenti pružaju konstantan tok povratnih informacija o njegovim karakteristikama, prednostima i manama, a stiče se i neprocenjivo iskustvo o njegovoj primeni u sferi obrazovanja. JEFF se već koristi i na nekim od naših projekata, pa je njegova upotreba kao profesionalnog alata već puštena u probu.

Enterprise profit interpretation tutor (PT)

Budući pravci razvoja za PT alat su usmereni ka njegovom usavršavanju kao nastavnog sredstva, ali i ka vršenju dodatnih evaluacionih studija. Što se tiče prve stvari, prvenstveno je neophodno otklanjanje nedostataka uočenih u toku evaluacije, ali i drugih nedostataka koji su uočeni u toku njegovog razvoja.

To znači da bi u cilju daljeg razvoja PT trebalo omogućiti:

- Bolje izveštavanje o greškama u toku rada
- Detaljnija objašnjenja
- Brže izvršavanje
- Jednostavnije pokretanje
- Bolji i moderniji GUI
- Bolji model podataka
- Veću (širu) bazu znanja

Bolje izveštavanje o greškama u toku rada se može postići prilagođavanjem teksta već postojećih poruka o greškama, ali i pravljenjem novih poruka za one greške koje nisu obrađene u ovom trenutku. Tu se prvenstveno misli na greške izazvane nekim sistemskim nedostacima (fali određena biblioteka ili fajl) ali i korisničkim akcijama koje nemaju veze sa unosom podataka (korisnik je ostavio otvoren izveštaj i pokušava

da napravi njegovu novu verziju).

Detaljnija objašnjenja su neophodna jer su korisnici naveli da im ipak fali još detalja da bi im sve bilo jasno. Proširenja ovih objašnjenja bi onda obuhvatala i neki deo teorije iz oblasti ekonomike preduzeća kao na primer definicije ili formule, ali i dodatne (kratke) primere na kojima se objašnjavaju neki osnovni principi i pravila.

Brže izvršavanje PT aplikacije je moguće postići na nekoliko načina. Sa obzirom na to da su procesorski najzahtevniji procesi inicijalizacije Drools baze znanja i Jasper Reports alata za izveštavanje, najveći dobici u brzini bi se dobili ako bi se makar jedan od ova dva procesa ubrzao. Dobri rezultati su već postignuti na taj način što se baza znanja ne kompajlira iz DRL fajla, već se koristi već iskompajlirana baza. Ona se samo učita iz fajla mehanizmom Java deserijalizacije u vidu gotovih Java objekata. To znači da se jednom kompajlirana baza sačuva (serijalizuje) u fajl, a da se svakim pokretanjem PT aplikacije koristi ovaj primerak baze. Pošto je mehanizam deserijalizacije mnogo brži od procesa kompajliranja, ušteda procesorskog vremena je značajna. Naravno, svaki put kada se pravila u bazi znanja promene, potrebno je ponovo izvršiti kompajliranje i serijalizaciju baze znanja da bi nova verzija bila dostupna PT aplikaciji. U dosadašnjem razvoju nije uočen način na koji bi se mogla ubrzati inicijalizacija alata za izveštavanje.

Jednostavnost pokretanja i **bolji i moderniji GUI** se mogu postići time što bi se PT pretvorio u web aplikaciju. Ovako, dizajn GUI-ja ne bi bio ograničen ni na koji način, a mogle bi da se koriste sve napredne tehnologije u cilju njegove modernizacije i povećanja interaktivnosti. Što se tiče jednostavnosti pokretanja, studenti ne bi morali da imaju kopiju aplikacije na svom računaru, već bi se njoj pristupalo preko browser-a. Time je izbegnuta i potreba da se na računar instalira Java, a za pristup bi bila potrebna samo web adresa. Jedini problem koji se može pojaviti je da je PT procesorski zahtevan, pa da bi istovremeni pristup velikog broja korisnika (studenata) zahtevao jake servere na kojima bi se aplikacija izvršavala. Naravno, i ovde su moguća određena poboljšanja sa obzirom na to da se, u suštini, ne pamte podaci jedne sesije sa korisnikom. To znači da bi svi pozivi bili bez “stanja” i da aplikacija ne bi morala da bude projektovana tako da prati šta svaki korisnik radi tj. mogla bi da svaki poziv na neku akciju tretira kao

“anoniman”.

Neophodno je napraviti **bolji model podataka** ako se PT namerava dalje razvijati i koristiti u radu. Osnovno ograničenje PT aplikacije (kao prototipa) je to što ne preuzima podatke iz DW ili OLAP, već se oni samo unose preko GUI-ja. Iako preslikavanje DW tj. OLAP u objektni model nije teško ostvariti preko aplikacijskih okvira za perzistenciju u Javi (Hibernate [106], Java Persistence API [107]), pitanje je kako bi objektni model podataka morao biti projektovan. Trenutni objektni model podataka se sastoji iz jedne Java klase sa velikim brojem atributa i metoda što, sa inženjerske tačke gledišta, niti je poželjno niti pravilno. Potencijalni načini za poboljšanje modela podataka koji se trenutno razmatraju su sledeći:

- Pravljenje malog DW ili OLAP iz kojeg bi se preuzimali podaci
- Pravljenje odgovarajućeg objektnog modela podataka i njegovo apstrahovanje u aplikacijski okvir

Kad bi se napravio odgovarajući DW ili OLAP i iz njega preuzimali podaci, to bi bilo najsličnije realnoj situaciji upotrebe. Model podataka u DW tj. OLAP bi bio denormalizovan u cilju jednostavnijeg izveštavanja. Slučajevi korišćenja alata bi se onda podelili u dve nezavisne grupe: one koji se tiču unosa podataka u DW tj. OLAP i one koji se tiču preuzimanja tih podataka i njihovog tumačenja. Sama realizacija bi donela dragoceno iskustvo u vezi sa tim kako bi povezivanje bilo ostvareno, koji bi se upitni jezik koristio (SQL ili MDX) itd.

Već je navedeno da je postojeći objektni model podataka potrebno ponovo projektovati. U toku izrade PT aplikacije je uočeno da se veliki broj elemenata iz modela podataka ponavljaju tj. da se mogu generalizovati u neke opštije koncepte. Na primer, skoro svaki poslovni pokazatelj ima neke osnovne atribute od kojih neki čine metapodatke: naziv, jedinicu mere, vremensku dimenziju, maksimalnu vrednost, minimalnu vrednost. KPI imaju i neke dodatne atribute kao ciljne vrednosti i kategorije ciljnih vrednosti. Opet, dešava se i to da se ciljne vrednosti KPI menjaju u toku vremena, pa bi bilo poželjno zapamtiti i prethodne vrednosti radi poređenja, kao i vreme kad su promenjene. Kada se ovi koncepti generalizuju u smislu aplikacijskog okvira, dodavanje novih pokazatelja i

njihovo preuzimanje iz skladišta podataka biće mnogo jednostavnije. Tako bi, na primer, dodavanje novog pokazatelja u bazu znanja podrazumevalo instanciranje već postojećih klasa i ne bi zahtevalo dodatno programiranje.

U cilju poboljšanja njegove funkcionalnosti, PT bi trebalo da ima i **veću (širu) bazu znanja**. Proširenje bi, u prvom trenutku, obuhvatalo dodavanje znanja za tumačenje vrednosti ukupnih prihoda i troškova po pojedinačnim faktorima. To znači da bi se, kada su troškovi u pitanju, u obzir uzimali fiksni troškovi i varijabilni troškovi i njihovi faktori:

- Materijalni troškovi
- Troškovi energije
- Troškovi amortizacije
- Troškovi radne snage itd.

Na ovaj način bi se mogla postići mnogo detaljnija analiza uzroka promena troškova. Na primer, moglo bi da se utvrdi da li je rast troškova primarno uzrokovan rastom cene radne snage ili ne.

Kada su prihodi u pitanju, analiza bi se proširila i na pokazatelje kao:

- Prihod po proizvodu (usluzi)
- Prihod po grupi proizvoda (usluga)
- Prihod po "prodajnom mestu" itd.

To bi omogućilo automatizovanu identifikaciju najboljih i najlošijih proizvoda i prodajnih mesta sa stanovišta prihoda, ali i njihovo međusobno poređenje i rangiranje. Naknadna proširenja baze znanja bo mogla da se tiču analize profita ali po proizvodima, prodajnim mestima itd.

Već je navedeno da bi bilo potrebno sprovesti i neke **dodatne evaluacione studije** u cilju procene efikasnosti i efektivnosti PT kao nastavnog sredstva. U ovom trenutku su u vidu dve potencijalne studije.

Prva studija bi obuhvatala procenu kvaliteta naučenog gradiva uz pomoć PT aplikacije.

Bila bi sprovedena u obliku eksperimenta, a studenti izabrani slučajnim izvlačenjem bi bili subjekti. Kontrolna grupa bi dobila klasične materijale za učenje (knjige i zbirke), dok bi eksperimentala grupa dobila i njih ali i PT aplikaciju. Obe grupe bi dobile za zadatak da savladaju određeno gradivo o tumačenju profita, a onda bi se putem testa proverio kvalitet naučenog gradiva jedne i druge grupe i uporedio. Kvalitet bi se poredio4 sa više aspekata, pa bi najverovatnije podrazumevao proveru znanja teorijskog gradiva ali i proveru primene tog znanja na konkretnim primerima. Osnovna hipoteza bi bila da korišćenje PT doprinosi boljem kvalitetu naučenog gradiva, a dokazivala bi se statističkim poređenjem pokazatelja iz samog testa.

Druga evaluaciona studija bi imala predavače ekonomije ili ekonomike kao ispitanike a bila bi realizovana u vidu ankete ili polustruktuiranog intervjua. Osnovni cilj bi bio da se utvrdi koji su utisci o korišćenju PT aplikacije sa aspekta predavača, kao i koje su prednosti i mane. Pre evaluacije, predavači bi bili obučeni kako da koriste PT a od njih bi se zahtevalo i da organizuju posebne časove na kojima bi koristili PT kao dodatno nastavno sredstvo. Nakon toga, anketom ili intervjuom bi se prikupilo njihovo mišljenje o celom iskustvu.

7 Zaključak

Problem istraživanja koji je prikazan u ovom radu se odnosi na veliku količinu podataka i nedostatak informacija u izveštajima SPI. Iako uglavnom prepuni grafika i tabela, izveštaji sadrže samo podatke tj. njihove grafičke predstave koje bi trebalo da omoguće jednostavniji uvid u poslovanje. To dovodi do posledice da korisnici ovih izveštaja moraju ručno da tumače ove podatke (pregledanjem više izveštaja), pa mogu da propuste da uoče neke bitne informacije zbog: prevelike količine podataka, premorenosti, nedostatka koncentracije, nedostatka znanja ili iskustva, zbog subjektivnosti ili zlonamernosti.

Predloženim pristupom se nalaže automatizacija procesa tumačenja podataka tj. njihovog pretvaranja u informacije. Ceo pristup se zasniva na jednačini koja opisuje da podaci postaju informacije kada im se doda značenje. Informacije su utoliko korisnije što za njihovo shvatanje nije potrebno dodatno tumačenje i direktno doprinose znanju primaoca. Najvažnije je to da informacije omogućavaju trenutno reagovanje tj. akciju. U ovom slučaju, to znači da korisnici čim pročitaju izveštaj koji sadrži i informacije, mogu da razmisle o tome koje akcije su neophodne jer odmah dobijaju detaljan uvid u stanje poslovanja.

Prikazano je da se, u slučaju izveštaja SPI, tumačenje dobija ukoliko se upotrebi domensko znanje potrebno za tumačenje vrednosti ovih poslovnih pokazatelja. Drugim rečima, automatizacija se postiže sakupljanjem i formalizovanjem znanja potrebnog za tumačenje i njegovom primenom nad podacima.

Iz svega navedenog, dalje je zaključeno da bi realizacija ovog pristupa obuhvatala primenu metoda, tehnika i tehnologija ES u cilju automatizacije procesa tumačenja vrednosti poslovnih pokazatelja. To je, ujedno, bio predmet istraživanja.

7.1 Provera tačnosti hipoteza

Sa stanovišta ostvarenih rezultata istraživanja, može se reći da su sve hipoteze dokazane: generalna, posebne i pojedinačne. Podsećanja radi, generalna hipoteza

istraživanja je:

> "Tehnike i tehnologije ekspertnih sistema mogu uspešno da se primene u okviru sistema za izveštavanje kao sredstvo za automatizovano formiranje informacija u izveštajima."

PT aplikacija metodom modelovanja na konkretnom softverskom prototipu upravo dokazuje da se tehnike i tehnologije ES mogu uspešno primeniti u SPI za automatizaciju pravljenja informacija u izveštajima. Konkretan izlaz iz ove aplikacije je i izveštaj koji sadrži podatke o profitu, ali i njihovo tumačenje u vidu informacija. Informacije su automatski izvedene korišćenjem tehnika i tehnologija ES.

Posebne hipoteze istraživanja sadrže tvrdnje da tehnike za predstavljanje znanja, tehnike zaključivanja i tehnike za objašnjavanje iz oblasti ES mogu uspešno da se koriste za predstavljanje znanja potrebnog za tumačenje vrednosti poslovnih pokazatelja, automatizovano zaključivanje (izvođenje informacija) i pretvaranje donetih zaključaka tj. informacija u rečenice koje liče na govorni jezik, respektivno.

Pojedinačnim hipotezama se specificiraju konkretne tehnike, pa se navodi da se deklarativni deo znanja (za tumačenje vrednosti pokazatelja) može predstaviti putem klasa, a proceduralni deo znanja putem pravila i fuzzy teorije. Zaključivanje se može izvršiti ulančavanjem unapred i fuzzy zaključivanjem, a informacije (zaključci) mogu da se pretvore u rečenice koje liče na govorni jezik korišćenjem tehnike učaurenog teksta.

U PT aplikaciji se upravo primenjuju sve navedene tehnike, pa se time na primeru dokazuje da su pojedinačne (a samim tim i posebne) hipoteze istinite. Deklarativno znanje potrebno za predstavljanje samih pokazatelja i njihovih vrednosti jeste predstavljeno putem Java klasa. Proceduralno znanje koje se odnosi tumačenje ciljnih vrednosti KPI ali i predstavljanje svih nejasnih izraza u poslovanju je predstavljeno putem fuzzy pravila i skupova i implementirano u jFuzzyLogic alatu. Na ovaj način su implementirana i pravila potrebna za određivanje trendova u vremenskim serijama. Svo preostalo proceduralno znanje koje se ne odnosi na nejasne izraze ili određivanje trendova vremenskih serija je predstavljeno preko proceduralnih pravila i

implementirano u Drools Expert alatu.

Zaključivanje u PT aplikaciji se vrši dvojako u datom redosledu: kao fuzzy zaključivanje (u jFuzzyLogic alatu) i kao ulančavanje unapred (u Drools Expert alatu). U prvom slučaju, podaci se svrstavaju u fuzzy kategorije i izvode se fuzzy zaključci. U drugom slučaju, gotovi fuzzy zaključci se koriste u okviru običnih pravila i ulančavanjem unapred se izvode konačni zaključci.

Konačno, svi izvedeni konačni zaključci se pretvaraju u informacije u vidu rečenica kontrolisanog jezika putem JEFF mehanizma za objašnjavanje. JEFF koristi tehnike učaurenog teksta (ali po potrebi i traga izvršavanja pravila) koja omogućava dodavanje delova objašnjenja kako se koji važan zaključak uz pomoć Drools Expert alata donese.

7.2 Ostvareni doprinosi

Ako se sve sumira, glavni doprinosi ostvareni istraživanjem su sledeći:

- Pružen je pregled postojećeg stanja iz oblasti SPI.
- Pružen je pregled postojećeg stanja iz oblasti ES.
- Formiran je novi pristup problemu pravljenja izveštaja SPI.
- Demonstriran je pristup na delu putem izrade prototipa (PT aplikacije).
- Pružena je ocena pristupa na osnovu evaluacije prototipa.

Pregled postojećeg stanja iz oblasti SPI (poglavlje 2.1) koji je dat u radu obuhvata prvenstveno definicije svih važnih pojmova iz ove oblasti kao i razgraničenja sa sličnim pojmovima kao što je poslovna inteligencija. Nakon toga, prikazane su sve metode, tehnike i tehnologije koje se koriste u SPI u cilju prikupljanja i skladištenja podataka, njihove obrade, kao i odgovarajuće analize i prikaza. Arhitektura savremenih SPI aplikacija je takođe obrađena, a napravljen je i pregled načina prostiranja poziva u ovakvim sistemima. Konačno, dat je i pregled aktuelnih trendova i kretanja u ovoj oblasti.

Pregled postojećeg stanja iz oblasti ES (poglavlje 2.2) takođe obuhvata odgovarajuće definicije i klasifikacije, nakon čega sledi pregled aktuelnih metoda, tehnika i

tehnologija. Ove metode, tehnike i tehnologije su grupisane u četiri oblasti koje se odnose na: predstavljanje znanja, predstavljanje neizvesnog znanja, zaključivanje i formiranje objašnjenja. Arhitektura ES je takođe objašnjena, nakon čega su dati savremeni primeri primene ES, kao i trendovi u ovoj oblasti.

Novi pristup problemu formiranja izveštaja SPI (poglavlje 3) je definisan i obrazložen sa teorijskog stanovišta. Prvo je jasno razgraničen problem koji se želi rešiti (i sa stanovišta teorije i kroz par primera), a to je da korisnici moraju ručno da tumače podatke u izveštajima SPI. Zaključeno je da je uzrok tog problema upravo to što u tim izveštajima nema informacija već sadrže samo podatke, a onda je prikazano idejno rešenje koje se sastoji u automatskom izvođenju informacija na osnovu poslovnih podataka. Razrada idejnog rešenja kroz izbor odgovarajućih metoda i tehnika ES i SPI, kao i određivanje arhitekture integracije ovog rešenja u SPI su usledili nakon toga. Diskusijom se ukazalo na (potencijalne) pozitivne i negativne aspekte rešenja, a zatim je dat i plan realizacije rešenja.

Demonstracija pristupa putem izrade prototipa - PT aplikacije (poglavlje 5) je omogućila proveru i dokazivanje postavljenih hipoteza. Ova jednostavna Java aplikacija omogućava tumačenje godišnjih vrednosti profita za preduzeća, ali i drugih veoma usko povezanih pokazatelja. PDF izveštaji koje PT aplikacija pravi sadrže i podatke (u vidu grafika i tabela) ali i informacije u vidu rečenica kontrolisanog jezika. Tekst kojim su predstavljene informacije je označen u zavisnosti od konteksta i omogućava brže uočavanje važnih zaključaka. Aplikacija je isprobana kao dodatno nastavno sredstvo na studijama menadžmenta.

Ocena pristupa na osnovu evaluacije prototipa (poglavlje 6.3) je obuhvatila rezultate evaluacione studije u kojoj su studenti bili ispitanici, a koristili su PT aplikaciju. Dokazano je da studenti smatraju da je PT aplikacija korisna i relativno jednostavna za korišćenje. Njen GUI je ocenjen kao jednostavan za navigaciju i prijatnog dizajna, ali istovremeno zastareo i sa veoma lošim izveštavanjem u slučaju pojavljivanja greške u radu. Brzina rada je takođe ocenjena kao veoma dobra, ali i tu ima mesta za poboljšanje. Objašnjenja u izveštajima se smatraju za korisna i lako razumljiva, ali bi bilo potrebno da budu obuhvatnija. Najveći broj ispitanih studenata veruje da rad sa PT aplikacijom

donekle doprinosi povećanju njihovog znanja ali i da pruža brži i donekle zanimljiviji način za savladavanje gradiva o tumačenju profita.

Budući pravci razvoja prototipa podrazumevaju poboljšavanje aplikacije putem otklanjanja nekih njenih nedostataka uočenih u toku evaluacije i izrade: bolji sistem izveštavanja o greškama, bolji i moderniji GUI, brže izvršavanje, detaljnija objašnjenja, ali ujedno i bolji model podataka i proširenje baze znanja tako da obuhvati još neke pokazatelje. Takođe, predviđeno je da se izvrše i dodatne evaluacione studije koje bi se odnosile na merenje efikasnosti PT kao nastavnog sredstva, ali i beleženje utisaka i merenje zadovoljstva predavača koji koriste PT u toku nastave.

Osim glavnih doprinosa ostvareni su i drugi dodatni doprinosi:

- Prikazana je komparativna analiza mehanizama za objašnjavanje u modernim BRE i BRMS alatima
- Napravljen je JEFF mehanizam za objašnjavanje za postojeće BRE i BRMS napisane u Javi
- Izvršena je evaluacija JEFF mehanizma za objašnjavanje

Komparativna analiza mehanizama za objašnjavanje u modernim BRE i BRMS alatima (Poglavlje 4.1) je poslužila kao osnova za pronalaženje mehanizma za objašnjavanje koji bi se koristio u istraživanju. Međutim, dokazano je da savremeni BRE i BRMS u Javi koji su obuhvaćeni analizom nemaju mehanizme za objašnjavanje namenjene pružanju objašnjenja krajnjim korisnicima. Objašnjenja koja su podržana su isključivo namenjena inženjerima znanja i drugom tehničkom osoblju koje razvija sistem i to u svrhu testiranja i debugging aktivnosti.

JEFF mehanizam za objašnjavanje (poglavlje 4) je aplikacijski okvir u Javi koji je besplatan i otvorenog koda i koji je nastao kao odgovor na rezultate prethodno pomenute komparativne analize. Njegov osnovni cilj je bio da omogući da BRE i BRMS napisani u Javi mogu da prave objašnjenja KAKO i STRATEGIJA namenjena krajnjim korisnicima koji nisu tehničko osoblje. U aspektu ovog istraživanja, njegova svrha je bila ta da se zaključci doneti tumačenjem podataka pretvore u rečenice kontrolisanog jezika.

Objašnjenja koja JEFF pruža mogu da se predstave u obliku rečenica kontrolisanog jezika, uz mogućnost dodavanja slika, tabela i grafika, ali mogu da uključe i trag izvršavanje pravila. Prvi oblik više odgovara netehničkom osoblju jer se objašnjenja mogu predstaviti u više paragrafa pa i strana teksta, dok drugi oblik pogoduje tehničkom osoblju i može se koristiti pri testiranju i debugging aktivnostima. Od tehnika za objašnjavanje, JEFF koristi učaureni tekst sa mogućnošću dinamičkog umetanja vrednosti na određenim mestima, ali i trag izvršavanja pravila.

Objašnjenja mogu da budu na bilo kojem jeziku. Dodavanje podrške za novi jezik je jednostavno i ne zahteva izmenu samog mehanizma. Izlaz može da bude u XML, PDF ili formi koja sadrži samo običan tekst.

Evaluacija JEFF mehanizma za objašnjavanje (poglavlje 6.2) je bila dvojaka a izvršena je radi procene da li su zadati ciljevi postignuti i gde se JEFF nalazi u odnosu na slične alate. Dokazano je da su skoro svi ciljevi postignuti i da je skoro podjednako dobar kao profesionalni alat u odnosu na koji je urađena komparativna analiza.

Mane JEFF mehanizma za objašnjavanje koje su uočene od strane krajnjih korisnika se odnose na to da je složen za korišćenje u situacijama kada je potrebno napraviti tekstualni sadržaj ili podatke kao sadržaj objašnjenja. U poređenju sa drugim alatom, zaključeno je da mu nedostaju neke opcije kao: objašnjenje ZAŠTO, napredno formatiranje izveštaja i GUI. Još jedan veliki nedostatak je nepostojanje mehanizma za zaštitu izveštaja od neovlašćenog pristupa.

Budući pravci razvoja JEFF alata se odnose na otklanjanje mana uočenih u toku evaluacije, ali i aktivnog nastojanja da se sve više koristi u realnim situacijama. Prvo podrazumeva pojednostavljenje načina na koji se pravi sadržaj za objašnjenja, izradu GUI-ja, proširenje mogućnosti formatiranja izveštaja i dodavanja sigurnosnih mehanizama. Da bi se JEFF koristio sve više u realnim situacijama, odlučeno je da se dodaju nove vrste izlaznih formata (za početak HTML), nove vrste objašnjenja (prvenstveno ZAŠTO) i nove vrste sadržaja objašnjeja (animacije, filmovi ili zvuk). Konačno, smatra se da bi veoma važan pravac razvoja bio i dodavanje interaktivnosti u objašnjenja i to tako da korisnik može da dobije dodatna pojašnjenja ako su mu/joj potrebna. Već je napomenuto da je prvi korak ka upotrebi u realnim situacijama

napravljen i da se JEFF trenutno koristi u okviru nekoliko istraživačkih projekata i ujedno kao nastavno sredstvo.

Kada se svi rezultati i doprinosi istraživanja sumiraju, može se reći da su hipoteze dokazane (zaista se predložene metode i tehnike mogu uspešno upotrebiti radi pravljenja informacija), ali da predloženi pristup nije isproban u situacijama za koje je namenjen. Prototip koji je napravljen dokazuje da je pristup moguć i izvodljiv za primenu, ali je isproban i razmotrena je samo njegova uspešnost u smislu nastavnog sredstva za studente, a ne kao sredstva za pomoć menadžerima u cilju tumačenja podataka.

7.3 Mogućnost primene rešenja

Da bi predloženi pristup zaista bio primenjen u praksi, potrebno je da se napravi aplikacija koja će da bude prilagođena menadžerima i koja će zaista da funkcioniše u nekom privrednom sistemu. To može da podrazumeva poboljšanje prototipa (PT aplikacije) ali ipak, najverovatnije, razvoj potpuno nove aplikacije koja će da se oslanja na iskustva i principe uočene u toku razvoja i evaluacije prototipa. Pri tome, mora se obratiti pažnja da se ostvari:

- Veoma precizan i promišljen izbor domena (koji pokazatelji će da se tumače).
- Implementacija sa određenim ključnim poboljšanjima u odnosu na prototip.
- Uspešna integracija sa postojećim SPI.

Veoma precizan i promišljen izbor domena iz kojeg će se tumačiti poslovni pokazatelji je ključni korak u cilju uspešne realizacije. Skup pokazatelja koji bude izabran mora da bude dovoljno složen jer se, u suprotnom, neće jasno videti prednost u odnosu na ručno tumačenje podataka. Iz istog razloga je potrebno i da podaci koji se tumače budu značajno velikog obima. Naravno, znanje koje se koristi za tumačenje pokazatelja iz izabranog domena mora da bude jasno definisano i da nema previše neodređenosti. Konačno, trebalo bi i utvrditi da li postoje značajni i verodostojni izvori znanja koji su dostupni u cilju njegovog prikupljanja i formalizovanja.

Implementacija sa određenim ključnim poboljšanjima u odnosu na prototip mora da bude ostvarena iz nekoliko razloga. Prvo, postojeći objektni model podataka nije dobro projektovan i postaće neodrživ u situacijama gde postoji veliki broj pokazatelja koje je potrebno često dodavati, brisati i menjati. Drugo, izveštaji koji se budu prosleđivali menadžerima će morati da imaju odgovarajuću zaštitu od neovlašćenog pristupa. Treće, okruženje u kojem će aplikacija raditi će biti distribuirano, što podrazumeva da se implementiraju i neki drugi mehanizmi kao: udaljen pristup izvorima podataka, distribucija i pristup izveštajima preko web-a (ili na neki drugi način), sinhronizacija sa nekim drugim procesima u postojećem informacionom sistemu itd. Četvrto, objašnjenja će morati da budu prilagođena menadžerima što podrazumeva da će se koristiti određeni skup stručnih pojmova i da velikog detaljisanja u vezi sa teorijskom osnovom na kojoj se donose zaključci neće biti (sistem neće prvenstveno služiti za podučavanje). Peto, moraće da se sprovede stroga kontrola kvaliteta ugrađenog znanja i odgovarajućih objašnjenja. Sve greške u smislu pogrešnih tumačenja se mogu veoma negativno odraziti na poslovanje, pa i izazvati velike gubitke. Konačno, najverovatnije će biti potrebno da se omogući i da menadžeri donekle mogu da izmene bazu znanja. To se prvenstveno odnosi na ciljne vrednosti KPI, jer nije poželjno da se za takve izmene svaki put angažuje tehničko osoblje.

Uspešna integracija sa postojećim SPI je neophodna da korisnici ne bi stekli utisak da rade sa odvojenom aplikacijom. To podrazumeva neke tehničke aspekte koji su već napomenuti, a tiču se rada u distribuiranom okruženju. To takođe može da podrazumeva i korišćenje nekog dela postojećeg SPI za generisanje izveštaja, u cilju postizanja uniformnosti izgleda isl.

8 Literatura

[1] Tomić B., Ekspertni sistemi i sistemi za izveštavanje, pristupni rad, Fakultet organizacionih nauka, Univerzitet u Beogradu, Beograd, 2008.

[2] Wikipedia the free encyclopedia, Enterprise reporting, 2011, dostupno u elektronskoj formi na adresi: http://en.wikipedia.org/wiki/Report

[3] Ćirić B., Poslovna inteligencija, Data status, Beograd, 2006.

[4] Biere M., Business intelligence for the enterprise, Prentice Hall, 2003.

[5] Vercellis C., Business Intelligence: Data Mining and Optimization for Decision Making, John Wiley and Sons, 2009

[6] Turban E., Sharda R. i Delen D., *Decision Support and Business Intelligence Systems*, Prentice Hall, 2010.

[7] California State University – Monterey Bay, *Data warehouse glossary*, 2007, dostupno u elektronskoj formi na adresi: http://it.csumb.edu/site/x7101.xml

[8] Songini M., *QuickStudy: Extract, Transform and Load (ETL)*, Computerworld, 2004, dostupno u elektronskoj formi na adresi: http://www.computerworld.com/s/article/89534/QuickStudy_ETL?taxonomyId=9&pageNumber=1

[9] Inmon W., *Building the Data Warehouse*, QED Technical Publishing Group, 1992.

[10] Nagabhushana S., *Data Warehousing - Olap And Data Mining*, New Age International, 2006.

[11] Georgetown University, *Data warehouse: glossary*, 2010, dostupno u elektronskoj formi na adresi: http://uis.georgetown.edu/departments/eets/dw/GLOSSARY0816.html

[12] E.F. Codd, S.B. Codd i C.T. Salley, *Providing OLAP to user analysts: an IT mandate*, Hyperion Solutions, 1993, dostupno u elektronskoj formi na adresi: http://www.minet.uni-jena.de/dbis/lehre/ss2005/sem_dwh/lit/Cod93.pdf

[13] Pendse N., *What is OLAP? An analysis of what the often misused OLAP term is supposed to mean*, Olap report, dostupno u elektronskoj formi na adresi: http://www.olapreport.com/fasmi.htm

[14] Adamson C., *Star Schema - The Complete Reference*, McGraw-Hill, 2010.

[15] Microsoft SQL development center, SQL Server 2008 Books Online: Multidimensional Expressions (MDX) Reference, 2019, dostupno u elektronskoj formi na adresi: http://msdn2.microsoft.com/en-us/library/ms145506.aspx

[16] Paredes J., *The Multidimensional Data Modeling Toolkit*, OLAP World Press, 2009.

[17] Brase C. H. i Brase C. P., *Understanding Basic Statistics*, Houghton Mifflin Company, 4th ed., 2007.

[18] Rumsey D., *Statistics for dummies*, Wiley Publishing, 2003.

[19] Rumsey D., *Statistics II for dummies*, Wiley Publishing, 2009.

[20] Frawley W., Piatetsky-Shapiro G. and Matheus C., Knowledge Discovery in Databases: An Overview, *AI Magazine*, pp. 213-228, 1992.

[21] MIT Open Course Ware, *Data Mining course*, 2003, doustupno u elektronskoj formi na adresi: http://ocw.mit.edu/OcwWeb/Sloan-School-of-Management/15-062Data-MiningSpring2003/CourseHome/

[22] Devedžić V., *Inteligentni informacioni sistemi*, Fakultet organizacionih nauka, Digit, Beograd, 2000.

[23] Devedžić V. i saradnici, *Tehnologije inteligentnih sistema*, Fakultet organizacionih nauka, Beograd, 2004.

[24] Microsoft SQL development center, *SQL Server 2005 Books Online: Data Mining Extensions (DMX) Reference*, 2007, dostupno u elektronskoj formi na adresi: http://msdn2.microsoft.com/en-us/library/ms132058.aspx

[25] Kaplan R. i Norton D., Using the Balanced Scorecard as a Strategic Management System, *Harward Business Review*, No. 74, 1996, pp. 75-87.

[26] Balance Scorecard Institute, *What is the Balanced Scorecard?*, 1998, dostupno u elektronskoj formi na adresi: http://www.balancedscorecard.org/BSCResources/AbouttheBalancedScorecard/tabid/55/Default.aspx

[27] Few S., *Information dashboard design: the effective visual communication of data*, O'Reiley Media, 2006.

[28] Sorenson M., *Glossary: What is executive dashboard*, Tech Target CIO Decisions Media, 2006, dostupno u elektronskoj formi na adresi: http://searchcio.techtarget.com/definition/executive-dashboard

[29] Hackathorn R., Minimizing action distance, *The Data Administration Newsletter*, 2003, dostupno u elektronskoj formi na adresi: http://www.tdan.com/view-articles/5132/

[30] White C., Building the Smart Business: In-Line Real-Time BI, *DM Review Magazine*, December 2004, dostupno u elektronskoj formi na adresi: http://www.information-management.com/issues/20041201/1014671-1.html

[31] Nicholls C., BI 2.0: The Next Generation, *DM Review Magazine*, November 2006, dostupno u elektronskoj formi na adresi: http://www.information-management.com/issues/20061101/1066763-1.html

[32] White C., Now is the Right Time for Real-Time BI, *DM Review Magazine*, September 2004,dostupno u elektronskoj formi na adresi: http://www.information-management.com/issues/20040901/1009281-1.html

[33] Negash S. i Gray P., Business Intelligence, in *Handbook on Decision Support Systems 2*, Eds. Burstein F. and Holsapple C., Springer - Verlag Berlin Heidelberg, pp. 175-195, 2008.

[34] Azvine B., Cui Z. i Nauck D., Towards real-time business intelligence, *BT Technology Journal*, Volume 23, No. 3, pp 214-225, 2005.

[35] Castellanos M., Dayal U. i Miller R. (Eds.), Enabling Real-Time Business Intelligence, Proceedings from the *Third International Workshop (BIRTE 2009)* held

at the 35th International Conference on Very Large Databases (VLDB 2009), Springer - Verlag Berlin Heidelberg, 2010.

[36] Michalewicz Z., *Adaptive business intelligence*, Springer - Verlag Berlin, 2007.

[37] Graco W., Semenova T. i Dubossarsky E., Toward knowledge-driven data mining, in *Proc. of the 2007 International Workshop on Domain Driven Data Mining: Conference on Knowledge Discovery in Data*, 2007.

[38] Charest M., Delisle S., Cervantes O. i Shen Y., Bridging the gap between data mining and decision support: A case-based reasoning and ontology approach, *Intelligent Data Analysis*, No. 12, 2008, pp. 211–236.

[39] Rennolls K. i AL-Shawabkeh A., Formal structures for data mining, knowledge discovery and communication in a knowledge management environment, *Intelligent Data Analysis*, No. 12, 2008, pp. 147–163.

[40] Xidonas P., Ergazakis E., Ergazakis K., Metaxiotis K., Askounis D., Mavrotas G. i Psarras J., On the selection of equity securities: An expert systems methodology and an application on the Athens Stock Exchange, *Expert Systems with Applications*, No. 36, 2009, pp. 11966–11980.

[41] Chakraborty C. i Chakraborty D., Fuzzy rule base for consumer trustworthiness in Internet marketing: An interactive fuzzy rule classification approach, *Intelligent Data Analysis*, No. 11, 2007, pp. 339–353.

[42] Bobillo F., Delgado M., Gomez-Romero J. i Lopez E., A semantic fuzzy expert system for a fuzzy balanced scorecard, *Expert Systems with Applications*, No. 36, 2009, pp. 423–433.

[43] Glossary of terms, PCAI magazine, dostupno u elektronskoj formi na adresi: http://www.pcai.com/web/glossary/pcai_d_f_glossary.html

[44] Durkin J., *Expert Systems - Design and Development*, Macmillan publishing company, 1994.

[45] Jackson P., *Introduction to Expert Systems*, Addison Wesley, 1999.

[46] Hopgood A., *Knowledge-Based Systems for Engineers and Scientists*, CRC Press,

1993.

[47] Kumar E., *Artificial Intelligence*, I. K. International Pvt. Ltd., 2009.

[48] Vermesan A., Foundation and Application of Expert System Verification and Validation, in *The Handbook of Applied Expert Systems*, Liebowitz J., Ed. ,CRC Press, 1993.

[49] Sowa J., Semantic Networks, in *Encyclopedia of Artificial Intelligence*, Shapiro S., Ed., Wiley, 1992.

[50] Goebel R. i Cantu-Ortiz F., Logic, in *The Handbook of Applied Expert Systems*, Liebowitz J., Ed. ,CRC Press, 1993.

[51] Tomić B., *Principi programiranja*, Fakultet Organizacionih Nauka, 2009.

[52] Zadeh L., Fuzzy sets, *Information and Control,* No. 8, Vol. 3, pp. 338–353, 1965.

[53] Zadeh L., Fuzzy algorithms, *Information and Control,* No. 12, Vol. 2, pp. 94–102, 1968.

[54] Donini F., Lenzerini M., Nardi D., Pirri F., i Schaerf M., Nonmonotonic reasoning, *Artificial Intelligence Review*, No. 4, pp. 163-210, 1990.

[55] Krishnamoorthy C. i Rajeev S., *Artificial Intelligence and Expert Systems for Engineers*, CRC Press, 1996.

[56] Forgy C., Rete: a fast algorithm for the many-pattern/many-object-pattern match problem, *Artificial Intelligence*, No. 19, Issue 1, pp. 17-37, 1982.

[57] Berry D., Explanation: The Way Forward, Expert Systems With Applications, No. 8, Issue 4, pp. 399-401, 1995.

[58] Ye R. i Johnson P.E., The impact of explanation facilities on user acceptance of expert system's advice, MIS Quarterly, No. 19, Issue 2, pp.157-172, 1995.

[59] Cawsey A., User-modelling in interactive explanations, *Journal of User Modelling and User Adapted Interaction*, No. 3, Issue 3, pp. 221–247, 1993.

[60] Dhaliwal J., Design and use of Explanation Facilities, in *The Handbook of Applied Expert Systems*, Liebowitz J. Ed., CRC Press, 1993.

[61] Wooley B., Explanation Component Of Software Systems, *ACM Crossroads*, No. 5, Issue 1, pp. 24-28, 1998.

[62] Tomić B., Devedžić V. i Jovanović J., Expert Systems Revisited: A Practical Approach, In *Progress in Expert Systems Research***,** Lipshitz, A. Ed., Nova Science Publishers Inc., 2007.

[63] Blickenstorfer C., *The Computer in Your Car - What is OBD-II (OnBoard Diagnostics)?*, 2006, dostupno u elektronskoj formi na adresi: http://pencomputing.com/

[64] *Engine Check*, dostupno u elektronskoj formi na adresi: http://www.enginecheck.co.uk/index.php

[65] *EASE Diagnostics*, dostupno u elektronskoj formi na adresi: http://www.obd2.com/autowatch/obd2/autowatch.htm

[66] Meta cog, *Auto tech*, dostupno u elektronskoj formi na adresi: http://www.metacog.com/index.htm

[67] Cisco inc., *Self Defending Network*, dostupno u elektronskoj formi na adresi: http://www.cisco.com/go/sdn

[68] Oracle corporation, *Self Managing Database*, dostupno u elektronskoj formi na adresi: http://www.oracle.com/

[69] Jannach D., Zanker M., Felfernig A. i Friedrich G., *Recommender Systems: An Introduction*, Cambridge University Press, 2010.

[70] Tintarev N. i Masthoff J., A Survey of Explanations in Recommender Systems, in *Proc. of the 2007 IEEE 23rd International Conference on Data Engineering Workshop*, Istanbul, Turkey, 2007, pp. 801-810.

[71] *JESS - The Rule Engine for the Java Platform*, Sandia National Laboratories, dostupno u elektronskoj formi na adresi: http://www.jessrules.com/

[72] *Exsys Corvid*, Exsys Inc., dostupno u elektronskoj formi na adresi: http://www.exsys.com/

[73] *Drools Expert Rule Engine*, JBoss Community, dostupno u elektronskoj formi na adresi: http://jboss.org/drools/drools-expert.html

[74] *Drools Guvnor BRMS*, JBoss Community, dostupno u elektronskoj formi na adresi: http://jboss.org/drools/drools-guvnor.html

[75] *Blaze Advisor - Business Rule Management System*, FICO (Fair Isaac Corporation), dostupno u elektronskoj formi na adresi: http://www.fico.com/en/Products/DMTools/Pages/FICO-Blaze-Advisor-System.aspx

[76] *IBM Websphere ILOG JRules*, IBM Corporation, dostupno u elektronskoj formi na adresi: http://www-01.ibm.com/software/ integration/business-rule-management/jrules/

[77] The Business Rules Group, *Defining Business Rules – What Are They Really?*, 2000, dostupno u elektronskoj formi na adresi: http://www.businessrulesgroup.org/first_paper/br01c0.htm.

[78] Chisholm M., *How to build a business rules engine*, Morgan Kaufmann, 2004.

[79] Von Halle B., *Business rules applied: business better systems using the business rules approach*, John Wiley and Sons, 2002.

[80] Ross R., *Principles of the Business Rules Approach*, Addison Wesley, 2003.

[81] Milanović M., Gašević D., Wagner G. i Hatala M., *Rule-enhanced Business Process Modeling Language for Service Choreographies*, in Andy Schuerr, Bran Selic (Eds): Model Driven Engineering Languages and Systems, 12th International Conference, MODELS 2009, Denver, Colorado, USA, October 4-9, 2009, Proceedings. Lecture Notes in Computer Science 5795.

[82] Davis B. i Olson H., *Management Information Systems: Conceptual Foundations, Structure, and Development*, McGraw-Hill, 1985.

[83] Silver A. i Silver L., *Systems Analysis and Design*, Addison Wesley, 1989.

[84] Checkland B. i Scholes J., *Soft Systems Methodology in Action*, John Wiley & Sons, 1990.

[85] Floridi L., Is Semantic Information Meaningful Data?, *Philosophy and Phenomenological Research*, No. 70, Issue 2, pp. 351-370, 2005.

[86] Tomić B., Automated Interpretation of Key Performance Indicators by using Rules, In Giurca, A., Gasevic, D., Taveter, K., Eds,, *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches*, IGI Publishing, 2009.

[87] Tomić B., Jovanović N. i Horvat B., JEFF - Java Explanation Facility Framework, dostupno u elektronskoj formi na adresi: http://sourceforge.net/projects/jeff/

[88] Tomić B., Jovanović N. i Horvat B., JEFF: Java Explanation Facility Framework, in P*roc. 16th YU INFO Conference on Computer Science an Information Systems (CD Edition)*, ed. D. Korunović (Kopaonik, Serbia, 2010), pp. 58-63.

[89] *Java Internationalization*, Oracle Corporation, dostupno u elektronskoj formi na adresi: http://java.sun.com/javase/technologies/ core/basic/intl/

[90] *Dom4j*, dostupno u elektronskoj formi na adresi: http://dom4j.sourceforge.net/

[91] *iText PDF: your Java-PDF library*, dostupno u elektronskoj formi na adresi: http://www.itextpdf.com/

[92] *JUnit*, dostupno u elektronskoj formi na adresi: http://www.junit.org/

[93] Tomić B. i Vlajić S., Functional Testing for Students: a Practical Approach, *Inroads - ACM SIGCSE Bulletin*, 40(4) (2008), pp. 58-62.

[94] Wine Advisor Expert System Knowledge Base, *Expertise 2 Go*, dostupno u elektronskoj formi na adresi: http://www.expertise2go.com/webesie/e2gdoc/wine.kb

[95] Jovanović N., *Evaluacija mehanizama za objašnjavanje sistema za upravljanje poslovnim pravilima*, diplomski rad, Fakultet Organizacionih Nauka, Univerzitet u Beogradu, 2010.

[96] Gamma E., Helm R., Johnson R. i Vlissides J., *Design Patterns: Elements of Reusable Software*, Addison-Wesley, 1995.

[97] Metsker S.J., *Design Patterns Java Workbook*, Addison Wesley, 2002.

[98] Cooper J., *The Design Patterns Java Companion*, Addison Wesley, 1998.

[99] Horvat B., *Alat za formatiranje izveštaja mehanizama za objašnjavanje ekspertnog sistema*, diplomski rad, Fakultet Organizacionih Nauka, Univerzitet u Beogradu, 2010.

[100] *jFuzzyLogic*, dostupno u elektronskoj formi na adresi: http://jfuzzylogic.sourceforge.net/html/index.html

[101] *Jasper Reports*, dostupno u elektronskoj formi na adresi: http://jasperforge.org/projects/jasperreports

[102] Milićević V. i Ilić B., *Ekonomika preduzeća: fokus na savremeno poslovanje*, Fakultet Organizacionih Nauka, 2005.

[103] Jovanović P., *Upravljanje investicijama*, Fakultet Organizacionih Nauka, 2005.

[104] Žarkić Joksimović N., *Upravljanje finansijama: osnove i principi*, Fakultet Organizacionih Nauka, 2005.

[105] *GNU Lesser General Public Licence (LGPL)*, dostupno u elektronskoj formi na adresi: http://www.gnu.org/licenses/lgpl.html

[106] *Hibernate*, dostupno u elektronskoj formi na adresi: http://www.hibernate.org/

[107] *Java Persistence API*, dostupno u elektronskoj formi na adresi: http://www.oracle.com/technetwork/articles/javaee/jpa-137156.html

9 Slike

10 Tabele

11 Biografija autora

Bojan Tomić je rođen 28.12.1980. godine u Beogradu. Pohađao je osnovnu školu "Petar Petrović Njegoš", i nosilac je Vukove diplome. Upisao je Matematičku gimnaziju u Beogradu 1995. godine, da bi je završio 1999. godine sa odličnim uspehom. Nosilac je nagrade „Najplemenitiji Jugosloven" za 1998. godinu koju dodeljuje list „Večernje novosti".

Fakultet organizacionih nauka je upisao 1999. godine - smer informacioni sistemi. Završio je osnovne studije u februaru 2005. godine sa prosekom 9,02. Vođen od strane mentora Prof. dr Vladana Devedžića, napisao je diplomski rad na temu "Softverski alat JavaDON za pravljenje ekspertnih sistema".

U novembru 2005. godine je upisao magistarske studije na Fakultetu organizacionih nauka, smer informacioni sistemi. U martu 2006. godine, dobio nagradu Beogradskog univerziteta za najbolji naučno-istraživački rad na nivou univerziteta za rad "JavaDON: An Open Source Expert System Shell". Posle pohađanja četiri semestra magistarskih studija, prebačen je na četvrti semestar doktorskih studija na Fakultetu organizacionih nauka, smer informacioni sistemi. Od početka doktorskih studija je objavio 10 radova u domaćim i stranim časopisima, na konferencijama kao i u monografijama od kojih su dva rada u časopisima sa SCI liste.

U oktobru 2007 godine, Bojan Tomić je počeo sa radom na Fakultetu organizacionih nauka u zvanju demonstratora, a u februaru 2008. godine stiče zvanje saradnika u nastavi na odseku za Softversko inženjerstvo i učestvuje u izvođenju nastave na više predmeta od kojih se izdvajaju „Principi programiranja" i „Inteligentni sistemi". U martu 2009. godine stiče zvanje asistenta na odseku za Softversko inženjerstvo i učestvuje i u nastavi na diplomskim akademskim (master) studijama, kao i na studijama na engleskom jeziku.

Prilog 1.

Izjava o autorstvu

Potpisani Bojan B. Tomić

broj upisa 08/2006

Izjavljujem

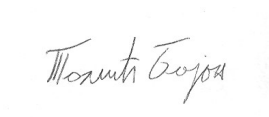
da je doktorska disertacija pod naslovom

Ekspertni sistemi i sistemi za izveštavanje

- rezultat sopstvenog istraživačkog rada,
- da predložena disertacija u celini ni u delovima nije bila predložena za dobijanje bilo koje diplome prema studijskim programima drugih visokoškolskih ustanova,
- da su rezultati korektno navedeni i
- da nisam kršio/la autorska prava i koristio intelektualnu svojinu drugih lica.

Potpis doktoranda

U Beogradu, 16.05.2012.



Prilog 2.

Izjava o istovetnosti štampane i elektronske verzije doktorskog rada

Ime i prezime autora	Bojan B. Tomić
Broj upisa	08/2006
Studijski program	Informacioni sistemi
Naslov rada	Ekspertni sistemi i sistemi za izveštavanje
Mentor	Prof. dr Vladan Devedžić

Potpisani Bojan B. Tomić

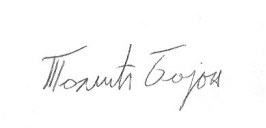
Izjavljujem da je štampana verzija mog doktorskog rada istovetna elektronskoj verziji koju sam predao/la za objavljivanje na portalu **Digitalnog repozitorijuma Univerziteta u Beogradu**.

Dozvoljavam da se objave moji lični podaci vezani za dobijanje akademskog zvanja doktora nauka, kao što su ime i prezime, godina i mesto rođenja i datum odbrane rada.

Ovi lični podaci mogu se objaviti na mrežnim stranicama digitalne biblioteke, u elektronskom katalogu i u publikacijama Univerziteta u Beogradu.

Potpis doktoranda

U Beogradu, 16.05.2012.



_____

Prilog 3.

Izjava o korišćenju

Ovlašćujem Univerzitetsku biblioteku „Svetozar Marković“ da u Digitalni repozitorijum Univerziteta u Beogradu unese moju doktorsku disertaciju pod naslovom:

Ekspertni sistemi i sistemi za izveštavanje

koja je moje autorsko delo.

Disertaciju sa svim prilozima predao/la sam u elektronskom formatu pogodnom za trajno arhiviranje.

Moju doktorsku disertaciju pohranjenu u Digitalni repozitorijum Univerziteta u Beogradu mogu da koriste svi koji poštuju odredbe sadržane u odabranom tipu licence Kreativne zajednice (Creative Commons) za koju sam se odlučio/la.

1. Autorstvo
2. Autorstvo - nekomercijalno
3. **Autorstvo – nekomercijalno – bez prerade**
4. Autorstvo – nekomercijalno – deliti pod istim uslovima
5. Autorstvo – bez prerade
6. Autorstvo – deliti pod istim uslovima

(Molimo da zaokružite samo jednu od šest ponuđenih licenci, kratak opis licenci dat je na poleđini lista)

Potpis doktoranda

U Beogradu, 16.05.2012.

Kratak opis licenci

1. Autorstvo - Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence, čak i u komercijalne svrhe. Ovo je najslobodnija od svih licenci.

2. Autorstvo – nekomercijalno. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela.

3. Autorstvo - nekomercijalno – bez prerade. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca ne dozvoljava komercijalnu upotrebu dela. U odnosu na sve ostale licence, ovom licencom se ograničava najveći obim prava korišćenja dela.

4. Autorstvo - nekomercijalno – deliti pod istim uslovima. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca ne dozvoljava komercijalnu upotrebu dela i prerada.

5. Autorstvo – bez prerade. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, bez promena, preoblikovanja ili upotrebe dela u svom delu, ako se navede ime autora na način određen od strane autora ili davaoca licence. Ova licenca dozvoljava komercijalnu upotrebu dela.

6. Autorstvo - deliti pod istim uslovima. Dozvoljavate umnožavanje, distribuciju i javno saopštavanje dela, i prerade, ako se navede ime autora na način određen od strane autora ili davaoca licence i ako se prerada distribuira pod istom ili sličnom licencom. Ova licenca dozvoljava komercijalnu upotrebu dela i prerada. Slična je softverskim licencama, odnosno licencama otvorenog koda.