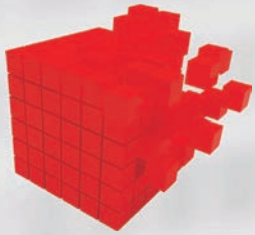
UNIVERZITET SINGIDUNUM

Angelina Njeguš

# POSLOVNI INFORMACIONI SISTEMI



*BUSINESS INFORMATION SYSTEMS*

Beograd, 2021.

UNIVERZITET SINGIDUNUM

Angelina Njeguš

# POSLOVNI INFORMACIONI SISTEMI

Deveto izdanje

Beograd, 2021.

# POSLOVNI INFORMACIONI SISTEMI

*Autor:*
dr Angelina Njeguš

*Recenzenti:*
dr Milan Milosavlejvić
dr Mladen Veinović

*Izdavač:*
UNIVERZITET SINGIDUNUM Beograd, Danijelova 32
www.singidunum.ac.rs

*Za izdavača:*
dr Milovan Stanišić

*Priprema za štampu:*
Jelena Petrović

*Dizajn korica:*
Aleksandar Mihajlović,MA

*Godina izdanja:*
2021.

*Tiraž:*
1200 primeraka

*Štampa:*
Caligraph, Beograd

ISBN: 978-86-7912-582-8

Copyright:
© 2021. Univerzitet Singidunum
Izdavač zadržava sva prava.
Reprodukcija pojedinih delova ili celine ove publikacije nije dozvoljeno.

*Mojim roditeljima, Gospavi i Vuksanu Njeguš*
*za večnu ljubav, strpljenje i podršku.*

# Sadržaj

## DEO I
## UVOD U POSLOVNE INFORMACIONE SISTEME


**UVOD U POSLOVNE INFORMACIONE SISTEME** **1**

**1. OSNOVNI KONCEPTI POSLOVNIH INFORMACIONIH SISTEMA** **2**

1.1. OPŠTA TEORIJA SISTEMA 2

1.2. INFORMACIONI SISTEMI 3

1.3. INFORMACIONA TEHNOLOGIJA 5

1.4. AUTOMATIZOVANI ALATI I TEHNOLOGIJE 6

1.4.1. Evolucija CASE tehnologije 8

1.4.2. Automatizovani alati 8

**2. ARHITEKTURA INFORMACIONIH SISTEMA** **11**

2.1. BLOKOVI ZNANJA 12

2.2. BLOKOVI PROCESA 15

2.3. BLOKOVI KOMUNIKACIJE 17

**3. POSLOVNI INFORMACIONI SISTEMI** **20**

3.1. ERP SISTEMI 21

3.2. KORISTI OD ERP SISTEMA 23

3.3. ERP MODULI 25

3.4. ANALIZA TROŠKOVA I KORISTI ZA ERP 27

3.5. POSLOVNI INFORMACIONI SISTEMI I STRATEŠKA PREDNOST 29


## DEO II
## RAZVOJ POSLOVNIH INFORMACIONIH SISTEMA


**RAZVOJ POSLOVNIH INFORMACIONIH SISTEMA** **35**

**4. METODOLOŠKE OSNOVE RAZVOJA INFORMACIONIH SISTEMA** **36**

4.1. MODELI I MODELIRANJE 37
4.2. STRATEGIJE SISTEMSKE ANALIZE 40
4.3. REINŽENJERING I MODELIRANJE PROCESA 41
4.3.1. Dijagram toka podataka 45
4.3.2. Rečnik podataka 53
4.3.3. IDEF0 – funkcionalno modeliranje 56
4.4. MODELIRANJE PODATAKA I ANALIZA 61
4.4.1. Osnovni pojmovi modela podataka 61
4.4.2. Modeliranje podataka 69
4.4.3 IDEF1X standard za modeliranje podataka 71
4.4.4. Analiza modela podataka 75
Prva normalna forma 77
Druga normalna forma 80
Treća normalna forma 81
Boyce-Coddova normalna forma 82
Četvrta normalna forma 84
Peta normalna forma 85
Normalna forma ključeva i domena (DK/NF) 86
Sinhronizacija sistemskih modela 87
4.5. OBJEKTNO-ORIJENTISANA ANALIZA I MODELOVANJE 88
4.5.1. Jedinstveni jezik modelovanja - UML 92
4.5.2. Proces objektnog modelovanja 105

**5. RAZVOJ POSLOVNIH REŠENJA PREMA MSF OKVIRU 108**
5.1. MSF DISCIPLINE 109
5.1.1. MSF procesni model 112
5.1.2. MSF timski model 113
5.1.3. Upravljanje kompromisima 115
5.2. TEHNIKE PRIKUPLJANJA INFORMACIJA I ANALIZIRANJE POSLOVNIH PROBLEMA I ZAHTEVA 116
5.2.1. Tehnike za prikupljanje informacija 119
5.2.2. Analiza poslovnih problema 125
5.2.3. Analiza zahteva 130
5.3. MODELOVANJE SISTEMSKIH ZAHTEVA SLUČAJEVIMA KORIŠĆENJA 132

5.3.1. Model slučajeva korišćenja 132
5.3.2. Modelovanje zahteva slučajevima korišćenja 134
5.4. ANALIZA IZVODLJIVOSTI REŠENJA I UPRAVLJANJE RIZICIMA 147
5.4.1. Faza definisanja vizije 147
5.4.2. Analiza rizika 159
5.4.3. Kategorije izvodljivosti projekta 162
5.5. KREIRANJE FUNKCIONALNE SPECIFIKACIJE POSLOVNOG REŠENJA 168
5.5.1. Faza planiranja prema MSF-u 170
5.5.2. Konceptni dizajn 172
5.5.3. Logički dizajn 179
Analiza kandidatnih tehnologija 180
Dokumentovanje logičkog dizajna 183
Optimizacija logičkog dizajna 187
5.5.4. Fizički dizajn 187
Projektovanje sloja prezentacije 197
Projektovanje sloja aplikacije 201
Projektovanje sloja podataka 204
Projektovanje bezbednosti 218
Kompletiranje faze planiranja 222
5.6. STABILIZACIJA I UVOĐENJE REŠENJA 225
5.6.1. Vođenje pilot rešenja 230
5.6.2. Faza uvođenja rešenja 231

**6. METODOLOGIJA IMPLEMENTACIJE ERP REŠENJA 233**
6.1. PROCES RAZVOJA ERP SISTEMA 233
6.2. ASAP METODOLOGIJA IMPLEMENTACIJE SAP SISTEMA 237

## DEO III
## ANALITIČKI POSLOVNI INFORMACIONI SISTEMI

**ANALITIČKI POSLOVNI INFORMACIONI SISTEMI 249**
**7. SKLADIŠTE PODATAKA 250**
7.1. RAZVOJ SKLADIŠTA PODATAKA 253

7.2. PRIMENA SKLADIŠTA PODATAKA 258

**8. OLAP SISTEMI 261**

8.1. ARHITEKTURE OLAP SISTEMA 267

8.1.1. Višedimenzioni OLAP (MOLAP) 267

8.1.2. Relacioni OLAP (ROLAP) 268

8.1.3. Hibridni OLAP (HOLAP) 270

**9. OTKRIVANJE ZNANJA I DATA MINING 272**

9.1. TEHNIKE DATA MINING-A 274

9.2. WEB MINING 284

**10. POSLOVNA INTELIGENCIJA 286**

10.1. KORACI RAZVOJA BI PROJEKTA 288

## DEO IV
## MODULI POSLOVNIH INFORMACIONIH SISTEMA

**MODULI POSLOVNIH INFORMACIONIH SISTEMA 317**

**11. ERP SISTEMI: PRODAJA I MARKETING 318**

11.1. MODULI PRODAJE I MARKETINGA U ERP SISTEMIMA 320

11.2. ERP I UPRAVLJANJE ODNOSIMA SA KLIJENTIMA (CRM) 322

**12. ERP SISTEMI: RAČUNOVODSTVO I FINANSIJE 324**

**13. ERP SISTEMI: PROIZVODNJA I UPRAVLJANJE MATERIJALIMA 326**

13.1. SAP R/3 ERP REŠENJE 331

**14 .ERP SISTEMI: LJUDSKI RESURSI 336**

**15. MENADŽMENT LANCA SNABDEVANJA I ELEKTRONSKO TRŽIŠTE 337**

# Predgovor

## Zašto proučavati poslovne informacione sisteme?

Informacione tehnologije (IT) ne spadaju više u resurse organizacije, već postaju poslovno okruženje. Informacioni sistemi (IS) formiraju jedan integralni deo moderne organizacije i poslovanja. Informacioni sistemi se koriste kao podrška svim aspektima organizacionih funkcija i aktivnosti. Međutim, koristi od novih tehnologija se mogu postići jedino ukoliko su usmereni ka organizacionim ciljevima.

Moderno poslovno okruženje zahteva i novi tip menadžera koji kombinuje menadžerske veštine sa ekspertizom u oblasti IT-a. Takav menadžer treba da:

- definiše strategiju IS bilo za radnu grupu, odeljenje ili kompaniju;
- identifikuje potencijalnu potrebu za IS kako bi poboljšao performanse kompanije;
- izabere i nabavi novi IS od odgovarajućih vendora;
- nadgleda razvoj i implementaciju novih sistema;
- upravlja IS kako bi se obezbedila efektivnost u pružanju kvalitetne informacije krajnjim korisnicima.

## Ciljevi

Knjiga „Poslovni informacioni sistemi“ treba da posluži kao vodič u odabiru pravih poslovnih informacionih rešenja za organizaciju, njihovom adekvatnom razvoju i efektivnom upravljanju. Knjiga je zamišljenja da bude jedini izvor za studente dodiplomskih studija Fakulteta za poslovnu informatiku Univerziteta Singidunum, koji proučavaju poslovne informacione sisteme, bez potrebe da nabavljaju različite knjige koje pokrivaju teme kao što su Sistemska analiza i projektovanje, Strategije razvoja IS, *Enterprise Resource Planning* (ERP), Analiza zahteva i definisanje arhitekture rešenja, Modeliranje informacionih sistema, *Data warehousing*, OLAP sistemi, *Data Mining* i dr.

Nakon savladavanja ove knjige, studenti će se osposobiti da:

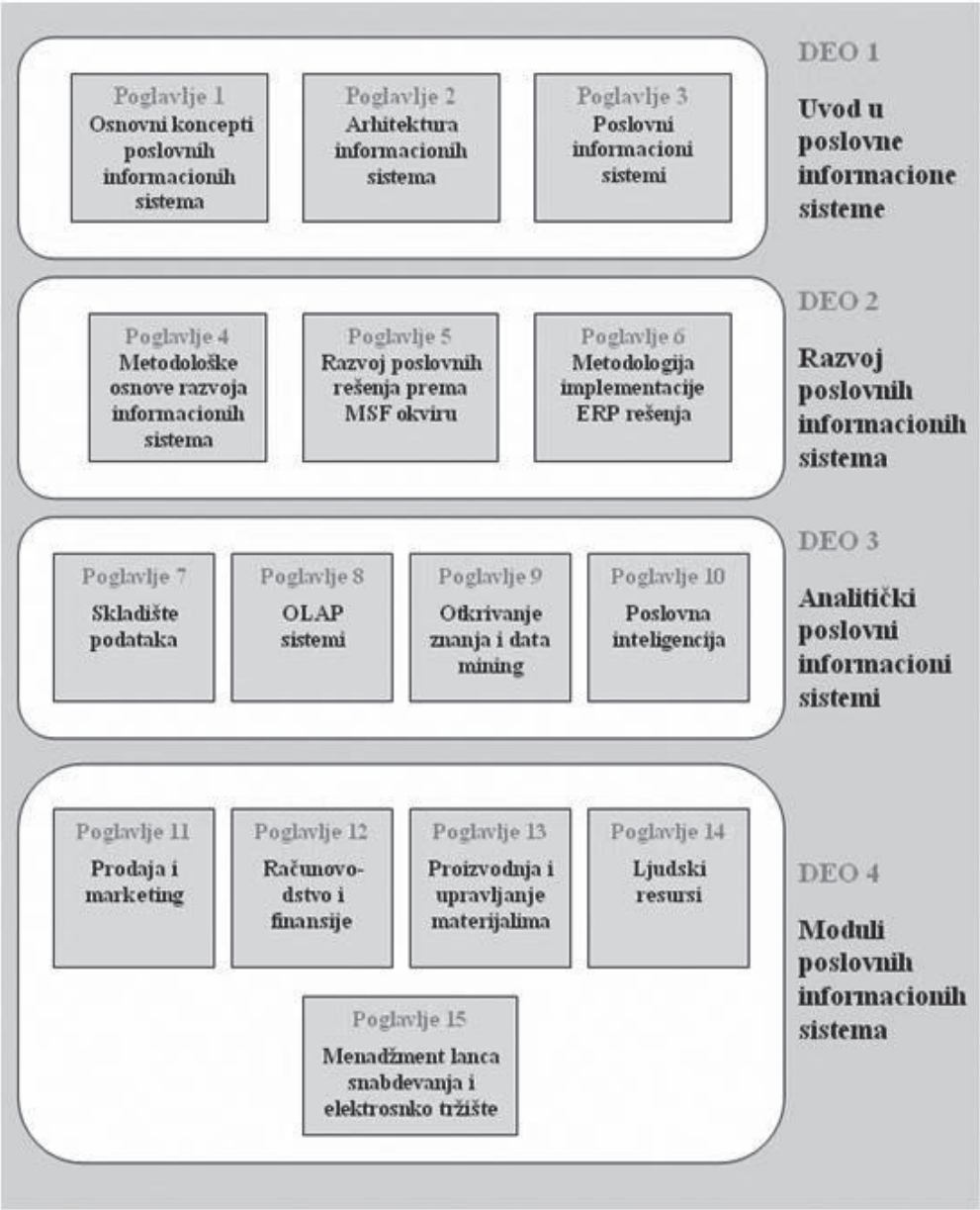
- razvijaju projektni plan razvoja i implementacije poslovnog rešenja;
- analiziraju faktore rizika uvođenja poslovnog rešenja;
- procenjuju i izaberu adekvatno IT rešenje;
- aktivno učestvuju u svim segmentima IT projekta;
- analiziraju module poslovnog rešenja i njihove veze;
- efektivno komuniciraju sa IT stručnjacima sa kojima sarađuju na projektu;
- koriste IT da bi pristupali izvorima informacija.

## Struktura knjige

Knjiga je podeljena u četiri dela, gde svaki deo pokriva različite aspekte upotrebe poslovnih informacionih sistema unutar organizacije:

- **Deo 1: Uvod u poslovne informacione sisteme** – fokusira se na osnovne koncepte i termine koji opisuju komponente poslovnih informacionih sistema; sagledava se arhitektura IS i analiziraju karakteristike, koristi, moduli i strateška prednost poslovnih informacionih sistema.
- **Deo 2: Razvoj poslovnih informacionih sistema** – bavi se proučavanjem različitih metodologija i okvira razvoja IS, polazeći od prikupljanja informacija i analize poslovnih problema i zahteva, preko modelovanja sistemskih zahteva, reinženjeringa poslovnih procesa, analize izvodljivosti rešenja i upravljanja rizicima, kreiranja logičkog i fizičkog dizajna do implementacije, stabilizacije i uvođenja IS u poslovno okruženje. U ovom delu se takođe razmatraju metodologije implementacije gotovih poslovnih rešenja.
- **Deo 3: Analitički poslovni informacioni sistemi** - opisuje analitički deo poslovnih informacionih sistema koji obuhvata *data warehousing*, *data mining*, OLAP sisteme, odnosno jednom rečju poslovnu inteligenciju (*Business Intelligence)* organizacije.
- **Deo 4: Moduli poslovnih informacionih sistema** - prikazuje modularne poslovne informacione sisteme kao što su prodaja i marketing, računovodstvo i finansije, proizvodnja i upravljanje materijalima, ljudski resursi i uopšte menadžment lanca snabdevanja i elektronsko tržište.

# Plan knjige


DEO 1
Uvod u poslovne informacione sisteme
Poglavlje 1 Osnovni koncepti poslovnih informacionih sistema
Poglavlje 2 Arhitektura informacionih sistema
Poglavlje 3 Poslovni informacioni sistemi
DEO 2
Razvoj poslovnih informacionih sistema
Poglavlje 4 Metodološke osnove razvoja informacionih sistema
Poglavlje 5 Razvoj poslovnih rešenja prema MSF okviru
Poglavlje 6 Metodologija implementacije ERP rešenja
DEO 3
Analitički poslovni informacioni sistemi
Poglavlje 7 Skladište podataka
Poglavlje 8 OLAP sistemi
Poglavlje 9 Otkrivanje znanja i data mining
Poglavlje 10 Poslovna inteligencija
DEO 4
Moduli poslovnih informacionih sistema
Poglavlje 11 Prodaja i marketing
Poglavlje 12 Računovo-dstvo i finansije
Poglavlje 13 Proizvodnja i upravljanje materijalima
Poglavlje 14 Ljudski resursi
Poglavlje 15 Menadžment lanca snabdevanja i elektrosnko tržište

# DEO I: UVOD U POSLOVNE INFORMACIONE SISTEME

## Ciljevi poglavlja

Sve organizacije razvijaju informacione sisteme. Bez obzira na vaše zanimanje u bilo kom poslovanju, zasigurno ćete učestvovati u nekom od segmenata razvoja informacionog sistema, bilo kao klijent ili korisnik tih sistema bilo kao sistem analitičar, programer ili sl. Metode koje ćete naučiti, čitajući ovu knjigu, mogu se primeniti na različite domene problema, a ne samo na one koji uključuju računar. Sistemski razvoj nije magija koja rešava sve probleme, ne postoji savršeni alati, tehnike niti metode koje garantuju da će projekat razvoja IS biti uspešan. Međutim, kompletna i konzistentna primena ovih veština vodi ka dugoročnoj strategiji uspešnosti poslovanja.

Pre nego što krenete da učite određene metode, tehnike, alate i tehnologije, neophodno je da razumete osnovne pojmove, koncepte i da uopšte steknete globalnu sliku o razvoju informacionih sistema. Prvi deo ove knjige vas uvodi u fundamentale koncepte, filozofije i trendove koje čine osnovu za dalju nadogradnju i primenu praktičnih alata i tehnika koje ćete naučiti u drugom delu ove knjige.

Nakon proučavanja ovog poglavlja, znaćete da:

- objasnite šta su sistemi, informacioni sistemi i informacione tehnologije;
- objasnite šta su CASE alati;
- ukratko objasnite sve procese (blokove) razvoja celokupnog informacionog sistema prema različitim fokusima na sistem;
- obrazložite pojam poslovnih informacionih sistema i ERP rešenja;
- sagledate module ERP rešenja, analizirate troškove i koristi za ERP sistem;
- sagledate strateške prednosti implementiranjem poslovnog informacionog sistema.

# 1. Osnovni koncepti poslovnih informacionih sistema

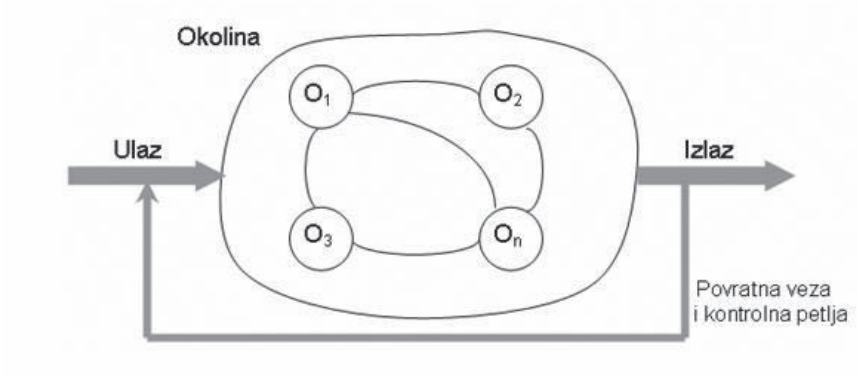
## 1.1. Opšta teorija sistema

Opšta teorija sistema predstavlja naučnu oblast koja se bavi izučavanjem sistema i zakonitosti koje u njima nastaju. Jedna od najvažnijih karakteristika teorije sistema jeste u pristupu, u okviru koga se svaka celina posmatra kao deo neke veće celine. Nastajanje opšte teorije sistema dovelo je do stvaranja sistemskog pristupa, kao i do novih tehnika i metoda analize sistema. Kada se upotrebi termin "sistemski pristup", tada se naglašava da se svi predmeti i pojave posmatraju u njihovoj dinamičnosti i celovitosti u odnosu na okruženje.

Cilj opšte teorije sistema je da služi kao jedinstveni metodološki i pojmovno-kategorijalni okvir sporazumevanja ljudi različitih specijalnosti. Takođe, njen cilj je da obuhvati i objedini fundamentalne pojmove koji važe u svim specifičnim sistemima i teorijama koje se njima bave. Imajući u vidu opštu teoriju sistema, u daljem tekstu definisaće se pojam sistema i navesti njegove karakteristike.

**Sistem se najopštije definiše kao skup objekata (entiteta) i njihovih međusobnih veza usmerenih ka ostvarivanju zajedničkog cilja.** Objekti u sistemu mogu da budu neki fizički objekti, koncepti, događaji i sl. Objekti u sistemu se opisuju preko svojih svojstava koja se nazivaju atributima. Skup objekata koji predstavlja posmatrani sistem definiše granice sistema. Sve izvan granica sistema se naziva okolina ili okruženje sistema. Dejstvo okoline na sistem opisuje se preko ulaza u sistem, a dejstvo sistema na okolinu preko njegovih izlaza, kao što je ilustrovano na slici 1.1.

Dinamičko ponašanje realnog sistema standardno se predstavlja na sledeći način: ulazi u sistem menjaju stanja sistema. Stanje sistema se definiše kao skup informacija o prošlosti i sadašnjosti sistema potrebnih da bi se, pod dejstvom budućih poznatih ulaza, mogli odrediti budući izlazi. U stanju sistema koncentrisana je celokupna istorija realnog sistema. Izlazna transformacija definiše neki način merenja ili posmatranja dinamičkog ponašanja realnog sistema i daje, na osnovu stanja sistema, njegove izlaze.




Slika 1.1. Opšta definicija sistema

Svaki sistem moguće je dekomponovati na podsisteme. Istovremeno, svaki sistem je deo nekog šireg sistema. Hijerarhičnost se mora uzeti u obzir prilikom istraživanja: ponašanja, funkcionisanja, razvoja i upravljanja sistemima. Sistemi ne egzistiraju izolovani, već teže da budu otvoreni sistemi. Otvorenost predstavlja komunikaciju između objekata sistema i objekata iz njegovog okruženja.

## 1.2. Informacioni sistemi

Terminologija pojmova u okviru informacionih sistema varira od autora do autora, međutim većina stručnjaka se slaže u sledećim definicijama:

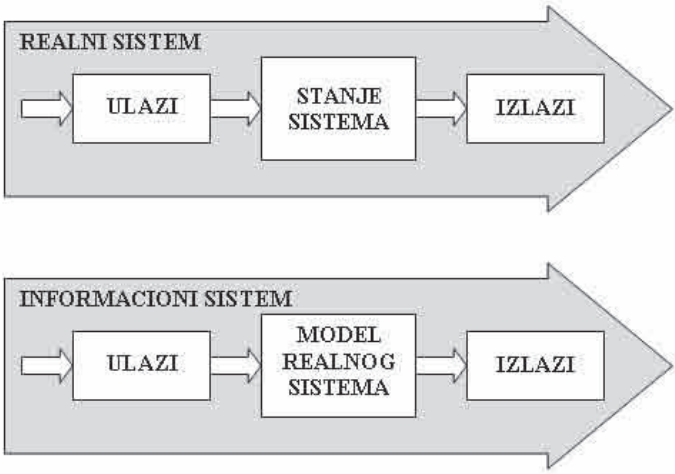
**Podatak** je kodirana predstava o nekoj činjenici iz realnog sveta. On je nosilac informacija i služi za tehničko uobličavanje informacija, kako bi se one mogle sačuvati ili preneti. Pojedinačni podaci sami za sebe nemaju nikakvo značenje ili ga imaju veoma malo.

**Informacija** je protumačeni podatak o pojavi koju podatak prikazuje. Drugim rečima, informacija je prečišćen, organizovan i obrađen podatak u smislenom kontekstu. Informacija je resurs koji je kreiran od podataka, da bi koristio menadžmentu pri donošenju poslovnih odluka. Sposobnost menadžmenta da prikuplja i uopšte upravlja podacima i informacijama postao je kritičan faktor uspešnosti poslovanja.

**Znanje** se gradi na temelju novih informacija koje se nadovezuju na postojeće znanje. Različiti ljudi mogu različito interpretirati informacije u zavisnosti od njihovog znanja.

Polazeći od tumačenja pojma sistema, proizilazi opšta definicija informacionog sistema. **Informacioni sistem (IS)** se može definisati kao sistem u kome se veze između objekata i veze sistema sa okolinom ostvaruju razmenom informacija. Detaljnije obrazloženje pojma informacionog sistema jeste da predstavlja **uređeni i integrisani skup podataka, procesa, interfejsa, mreža, tehnologija i ljudi koji su u međusobnoj korelaciji u cilju podrške i poboljšanja svakodnevnih poslovnih operacija i podrške menadžmentu u rešavanju poslovnih problema, planiranja, upravljanja, predviđanja, koordinisanja i donošenja odluka**.

Informacioni sistem treba da bude model realnog sistema u kome deluje (Slika 1.2). Ulazi u sistem menjaju stanje sistema, a ova promena se reflektuje na izlaz. Preslikavanje realnog sistema u informacioni sistem izvodi se postupkom modeliranja realnog sistema.




Slika 1.2. Položaj informacionog sistema u odnosu na realni sistem

Za većinu ljudi, sistemi su preveliki i kompleksni da bi se shvatili u celosti, na primer, inženjeri koji prave automobile kreiraju modele pre nego što naprave nova kola, arhitekte prave modele zgrada koje projektuju itd. Poslovni informacioni sistemi se ne mogu razumeti bez modela koji predstavljaju sisteme u njihovoj realnosti.

## 1.3. Informaciona tehnologija

Informacione tehnologije su deo razvojne strategije informacionih sistema, te se stoga ovde ukratko obrazlažu.

**Informacione tehnologije (IT) opisuju kombinaciju računarske tehnologije (*hardware* i *software*), telekomunikacione tehnologije, *netware, groupware* i *humanware*:**

- ***Hardware*** – podrazumeva fizičku opremu kao što su mehanički, magnetski, elektronski ili optički uređaji.
- ***Software*** – uključuje predefinisane instrukcije koje kontrolišu rad računarskih sistema ili elektronskih uređaja. Softver koordinira rad hardverskih komponenata u jednom informacionom sistemu. Softver inkorporira standardne softvere kao što su operativni sistemi ili aplikacije, softverske procese, veštačku inteligenciju, inteligentne agente i korisnički interfejs.
- **Telekomunikacije** – podrazumevaju prenos signala duž različitih distanci koji uključuju i prenos podataka, slika, glasova, koristeći radio, televiziju, telefoniju i druge komunikacione tehnologije.
- ***Netware*** – podrazumeva opremu i softver neophodne za razvoj i podršku mreže računara, terminala i komuniokacionih kanala i uređaja.
- ***Groupware*** – predstavlja komunikacione alate kao što su e-mail, videokonferencije i dr., koji podržavaju elektronsku komunikaciju i kolaboraciju između grupa.
- ***Humanware*** – podrazumeva intelektualne kapacitete neophodne za razvoj, programiranje, održavanje i rukovanje tehnologijom. *Humanware* inkorporira znanje i ekspertizu.

Mnoge promene u projektovanju poslovnih procesa, reinženjeringu i deljenju informacionih resursa su znatno olakšane implementacijom informacione tehnologije. Integrisane baze podataka, gde organizacione jedinice dele zajedničke podatke koje se održavaju u centralnoj bazi podataka, predstavljaju osnovu za poslovne informacione sisteme. Mnoge su prednosti integrisanja baza podataka, kao što su poboljšana konzistentnost, integritet, nezavisnost i deljenje podataka i smanjena redundantnost podataka.

## 1.4. Automatizovani alati i tehnologije

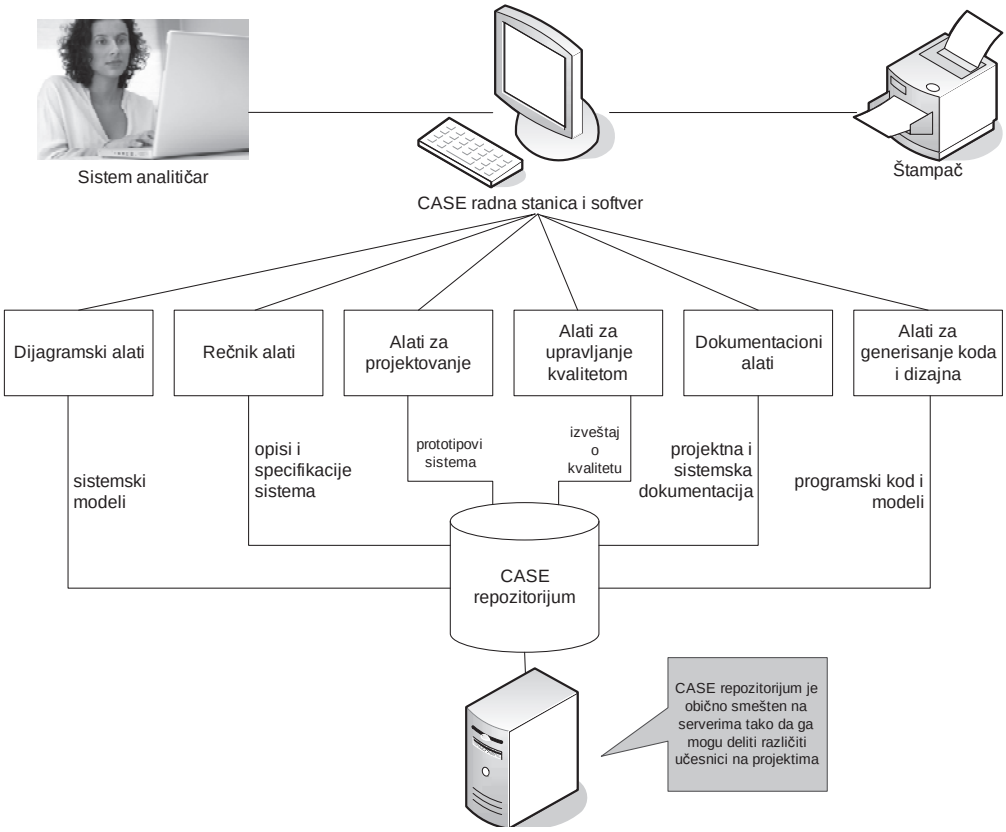
Celokupna današnja informaciona tehnologija je usmerena ka podršci onima koji se bave razvojem informacionih sistema. Ta tehnologija se naziva sistemski inženjering pomoću računara (*Computer-Assisted Systems Engineering* – CASE). **CASE alati su programi (softveri) koji automatizuju i podržavaju jednu ili više faza životnog ciklusa razvoja sistema.** Namera ove tehnologije jeste da ubrza procese razvoja sistema i poboljša njihov kvalitet.

CASE nije metodologija niti bilo kakva njena alternativa. CASE je tehnologija koja podržava metodologije naročito strategije, tehnike i standarde. CASE tehnologija automatizuje celokupnu metodologiju razvoja sistema.

Neki ovu tehnologiju nazivaju softverski inženjering pomoću računara (*computer-aided software engineering*), međutim treba imati u vidu da je softver samo jedna komponenta informacionog sistema, pa se stoga ovde koristi širi pojam - sistem.

U centru bilo koje arhitekture CASE alata se nalazi baza podataka koja se naziva CASE repozitorijum (*repository*) (Slika 1.3). Oko CASE repozitorijuma se nalazi kolekcija alata koji kreiraju sistemske modele i dokumentaciju. Koncept repozitorijuma podrazumeva skladište podataka koji obuhvata specifikacije, dokumentacije i druga znanja, koja se stvaraju tokom sistemskog razvoja projekta. Iskustvo je pokazalo da većinu ljudi, kod CASE alata, prvo privuku njegove grafičke mogućnosti. Međutim, prava snaga CASE alata jeste u njenom repozitorijumu. CASE repozitorijum je baza podataka onih koji razvijaju sistem. To je mesto gde se skladište dijagrami, opisi, specifikacije i drugi elementi koji se pojavljuju u toku razvoja sistema. Mnogi različiti CASE alati mogu da dele informacije putem jedinstvenog repozitorijuma. Da bi se mogao koristiti repozitorijum, CASE alati obezbeđuju kombinaciju sledećih alata (Slika 1.3):

- **Dijagramski alati (*Diagramming tools*)** se koriste za crtanje sistemskih modela.
- **Rečnik alati (*Dictionary tools*)** se koriste za snimanje, brisanje, izmenu i prikazivanje detaljne dokumentacije i specifikacije. Opisi su obično pridruženi elementima sistemskih modela koji su prethodno iscrtani dijagramskim alatima.




Slika 1.3. Arhitektura CASE alata

- **Alati za projektovanje (*Design tools*)** se koriste za projektovanje komponenata sistema uključujući ulaze (*inputs*) i izlaze (*outputs*).
- **Alati za upravljanje kvalitetom (Q*uality management tools*)** analiziraju i utvrđuju konzistentnost i kompletnost modela, opisa i dizajna. Ukoliko dođe do pojave greške, CASE alati ih identifikuju i obaveštavaju korisnike.
- **Dokumentacioni alati (*Documentation tools*)** se koriste za sakupljanje, organizaciju i izveštavanje neophodne dokumentacije iz repozitorijuma.
- **Alati generatora koda i dizajna (*Design and code generator tools)*** automatski generišu dizajn baze podataka iz modela podataka, aplikacione programe ili značajne delove ovih programa.

### 1.4.1. Evolucija CASE tehnologije

CASE alati datiraju još od sredine 1970-tih godina. Pod projektom ISDOS, vođenog od strane dr Daniela Teichrowe sa Mičigen Univerziteta, je razvijen jezik za iskazivanje problema (*Problem Statement Language - PSL)* koji je služio za opisivanje korisničkih problema i zahteva. Drugi jezik *Problem Statement Analyzer (PSA)* je kreiran da analizira ove probleme i zahteve. *PSL/PSA* su se aktivirali sa velikih *mainframe* računara i iziskivali su veoma skupe računarske resurse, tako da je malo koja kompanija tada mogla sebi da priušti adekvatne računarske resurse za *PSL/PSA*.

Preokret je nastao sa pojavom personalnih računara (PC) kada je 1984. godine jedna kompanija, koja se tada zvala Index Technology (danas poznata kao Intersolv), kreirala PC softverski alat koji se zvao *Excelerator,* čiji uspeh je postavio CASE akronim. U današnje vreme, onima koji se bave razvojem sistema, na raspolaganju su hiljade CASE proizvoda.

CASE tehnologija je neverovatno slična drugim inženjerskim tehnologijama, kao što su projektovanje pomoću računara (*Computer-Aided Design* - CAD) ili proizvodnja pomoću računara *(Computer-Aided Manufacturing* - CAM). Savremeni inženjeri koriste CAD alate da projektuju i analiziraju nove proizvode. CAM alati onda automatski generišu računarske programe koje će aktivirati neophodnu mašineriju za proizvodnju datog dizajna. Kao što CAD/CAM radi za inženjere, tako i CASE radi za one koji se bave razvojem informacionih sistema.

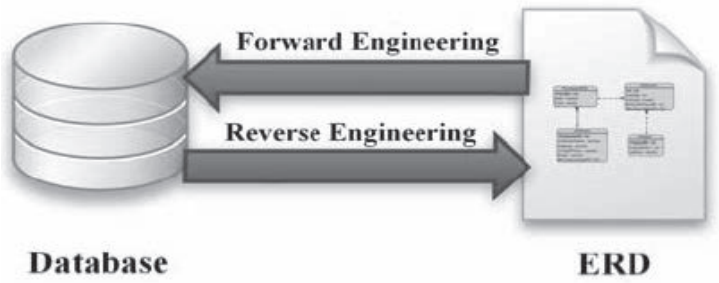
### 1.4.2. Automatizovani alati

Postoje tri klase automatizovanih alata za developere:

- modeliranje sistema pomoću računara (*computer-aided systems modeling*)
- okruženje za razvoj aplikacija (*application development environments*)
- upravljanje projektima i procesima (*project and process management*)

Današnji CASE alati omogućavaju dva različita pristupa za razvoj sistemskih modela i to (Slika 1.4):

- **inženjering unapred (*forward engineering*)** - sposobnost CASE alata da generiše inicijalni softver ili kôd baze podataka direktno iz sistemskih modela.
- **reverzni inženjering (*reverse engineering*)** - sposobnost CASE alata da automatski generiše inicijalne sistemske modele iz softvera ili kôda baze podataka.




Slika 1.4. Inženjering unapred/reverzni

Reprezentativni primeri CASE alata su: BPWin, ERWin, *System Architect, Rational* ROSE UML, *DataArchitect, Oracle Designer*, *SmartDraw*, *Power Designer* i mnogi drugi.

### Okruženje za razvoj aplikacija

Okruženje za razvoj aplikacija (*Application Development Environment* – ADE) je jedan integrisani alat za brzi i kvalitetan razvoj softvera. Sinonim za ADE je *Integrated Development Environment* (IDE). ADE čini programiranje jednostavnijim i efikasnijim. Uglavnom obebezbeđuju sledeće alate:

- **programske jezike i interpretere** – čine srce svakog okruženja za razvoj aplikacija. Moćni alati za ukazivanje na greške u kodu (*debugging)* i pomoć pri njihovom otklanjanju pomažu programerima da brzo identifikuju i reše probleme.
- **alate za dizajniranje interfejsa** – pomažu programerima da, ko-risteći biblioteku komponenti, brzo naprave korisničke interfejse.
- ***middleware*** – softver koji pomaže programerima da integrišu

softver za drugim bazama podataka i računarskim mrežama.

- **alate za testiranje** – koriste se za pravljenje i izvršavanje test skriptova koji mogu konzistentno i potpuno da testiraju softver.
- **alate za kontrolu verzija** – pomažu programerskim timovima da upravljaju višestrukim verzijama programa, kako u toku razvoja tako i nakon implementacije.
- **alate za pomoć (*Help*)** - koriste se za pisanje *on-line* sistema za pomoć, korisničkih priručnika i za *on-line* obučavanje.
- **repozitorijum *link*** – dozvoljava ADE da se integriše sa različitim CASE alatima, kao i sa drugim ADE i razvojnim alatima.

Reprezentativni primeri okruženja za razvoj aplikacija su: *Microsoft Visual Studio.net* (VB.*net*, C#.*net*, J#.*net*), *Oracle Developer*, IBM *Websphere* (Java), *Inprise*[1] J *Builder* (Java), *Macromedia Cold Fusion*, *Sybase Powerbuilder* i mnogi drugi.

### Upravljanje procesima i projektima

Treća klasa automatizovanih alata pomaže kod upravljanja metodologijom razvoja sistema i projektima koji koriste metodologiju. Dok CASE alati i ADE podržavaju analizu, projektovanje i izgradnju novih informacionih sistema i softvera, aplikacije upravljanja procesima i projektima podržavaju životni ciklus aktivnosti.

**Aplikacija upravljanja procesom (*Process manager application*)** je automatizovan alat koji pomaže pri dokumentaciji i upravljanju metodologijama, njihovim isporukama i standardima upravljanja kvalitetom.

**Aplikacija upravljanja projektom (*Project manager application*)** je automatizovan alat koji pomaže u planiranju aktivnosti razvoja sistema, ocenjuje i dodeljuje resurse (uključujući ljude i troškove), raspoređuje aktivnosti i resurse, prati napredovanje prema zadatom rasporedu i budžetu, kontroliše i modifikuje rasporede i resurse i izveštava o napredovanju projekta. Neki od primera automatizovanih alata za upravljanje projektima su *Microsoft Project*, *Applied Business Tecnology - Project Manager Workbench* i dr.

1 Ex Borland.

# 2. Arhitektura informacionih sistema

Arhitektura informacionih sistema obezbeđuje jedinstveni okvir (*framework*) po kome će različiti ljudi sa različitim pogledima organizo-vati fundamentalne blokove razvoja informacionih sistema (Slika 2.1). Pretpostavimo da želite da razvijate jedan informacioni sistem. Različiti ljudi će imati različite poglede na sistem. Menadžeri, korisnici, tehnička lica, svi oni će posmatrati sistem na različit način i sa različitim nivoom detalja. Ove ljude nazivamo nosiocima informacionog sistema, odnosno *stakeholders*-ima. Oni se grubo mogu klasifikovati u četiri grupe [7]:

- Vlasnici sistema *(System Owners)* finansiraju razvoj i održavanje informacionog sistema. Oni poseduju sistem, postavljaju viziju i prioritete u sistemu.
- Korisnici sistema *(System Users)* su ljudi koji za obavljanje svojih poslova, koriste informacioni sistem. Pored internih korisnika sistema, koji rade unutar jedne organizacije, tu spadaju i eksterni korisnici kao što su klijenti, vendori, partneri i oni zaposleni koji rade sa udaljenih lokacija, na primer sa terena ili od kuće. Danas korisnici sistema rade rame uz rame sa projektantima sistema.
- Projektanti sistema *(System Designers)* projektuju sistem kako bi izašli u susret zahtevima korisnika. Oni projektuju baze podataka, ekrane, mreže, programe i dr. Tu sapadaju administratori baza podataka, mreža, Web dizajneri, eksperti za bezbednost i drugi tehnički stručnjaci. U nekim slučajevima, projektanti sistema mogu biti i tzv. graditelji sistema.
- Graditelji sistema *(System Builders)* su tehnička lica koja konstruišu, testiraju, isporučuju, uvode i održavaju informacioni sistem. Neki od njih su programeri aplikacija, sistem programeri, *Webmaster*-i (oni koji kodiraju i održavaju *Web* servere), sistem integratori (integrišu softverske pakete sa hardverom, mrežama i drugim softverskim paketima) i dr.

Kao što je prikazano na slici 2.1., svaka grupa *stakeholders*-a je jedan red u okviru informacionog sistema i može se videti, da generalno postoje četiri pogleda na informacione sisteme, a to su pogledi vlasnika, korisnika, projektanata i graditelja.

Njihovi pogledi se u mnogome razlikuju, dok su jedni fokusirani na opštu sliku, drugi su zainteresovani za detalje, nečiji pogledi su ne-tehnički, dok su drugi vrlo tehnički. Sistem analitičar premošćava komunikacioni jaz između onih kojima trebaju računari i onih koji dobro poznaju tehnologiju. On je odgovoran za prikazivanje krajnjim korisnicima i menadžmentu kako nova tehnologija može povećati efektivnost njihovog poslovanja i njihovih operacija. Sinonimi, u svetu, za sistem analitičara su: arhitekta sistema, sistem inženjer, informacioni inženjer, informacioni analitičar ili sistem integrator.

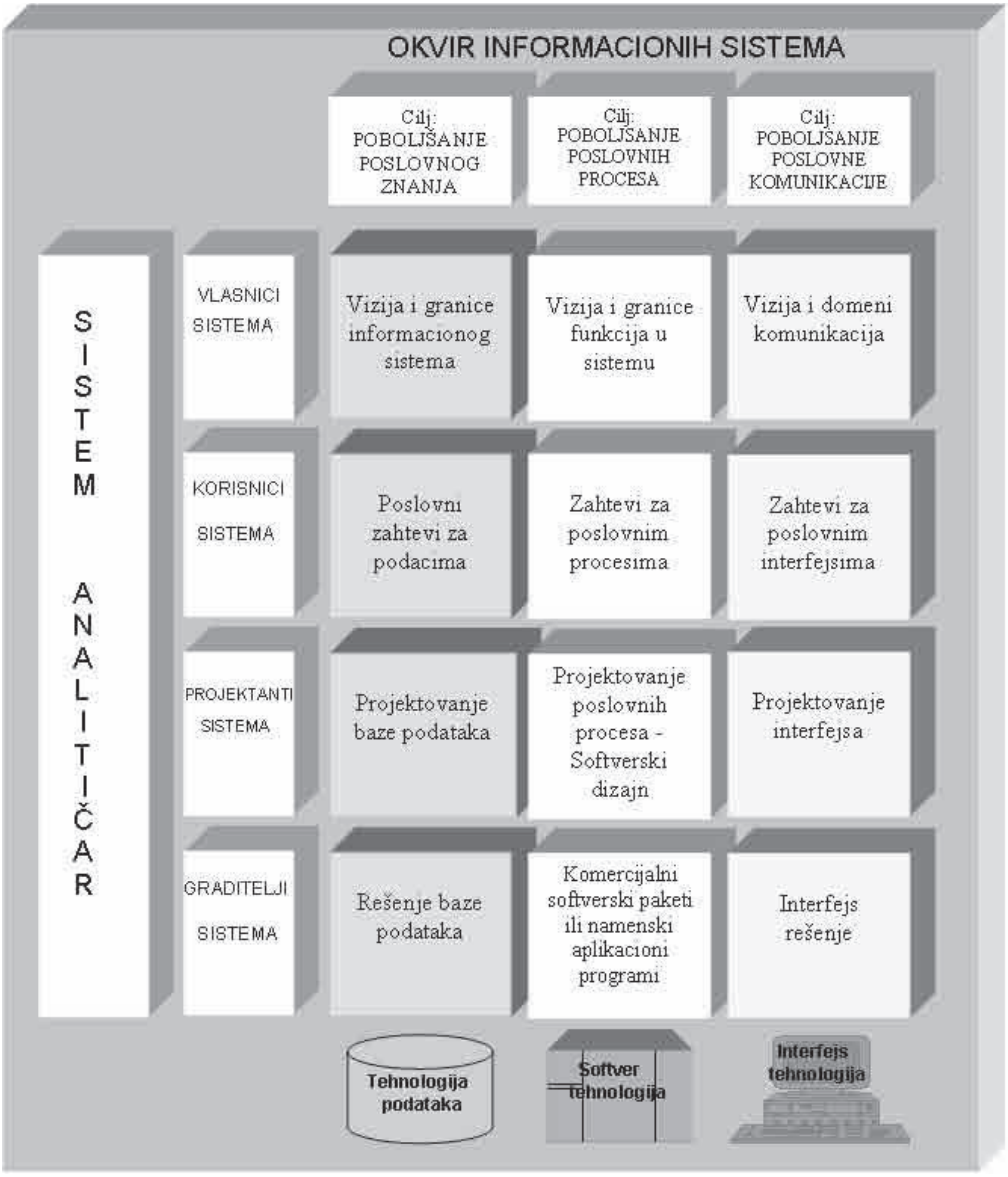
Različiti *stakeholders*-i se mogu usredsrediti na različite aspekte sistema:

- **Znanje** – poslovno znanje treba da pomogne menadžerima u do-nošenju inteligentnih odluka. Cilj je poboljšanje baze znanja u organizaciji.
- **Procesi** – aktivnosti koje izvršavaju misiju poslovanja. Cilj je poboljšanje poslovnih procesa i usluga.
- **Komunikacije** – interfejs sistema sa korisnicima i drugim informacionim sistemima. Cilj je poboljšanje poslovne komunikacije.

Preseci pogleda (redova) i svakog fokusa (kolona) definišu fundamentalne blokove informacionog sistema (Slika 2.1). Blokovi informacionog sistema ne egzistiraju izolovano, već moraju biti sinhronizovani kako bi se izbegle nedoslednosti i nekompatibilnosti unutar sistema. Na primer, projektant baze podataka i programer imaju svoj sopstveni pogled na arhitetkturu sistema, ali ti pogledi moraju biti koordinirani i kompatibilni, da bi sistem bio uspešan.

## 2.1. Blokovi znanja

Kada inženjeri projektuju jedan proizvod, oni moraju da sačine sastavnicu tog proizvoda. Sastavnica ne govori o funkciji tog proizvoda, već o njegovim sastavnim delovima, odnosno pokazuje koje su to sirovine, poluproizvodi i druge komponente koje učestvuju u izgradnji finalnog proizvoda. Ista analogija se koristi i za informacione sisteme. Podaci se mogu posmatrati kao sirovine koje se koriste da bi se “proizvele” informacije.




Slika 2.1. Fundamentalni blokovi informacionih sistema[1]

Podaci se mogu smatrati kao jedan od primarnijih blokova razvoja informacionog sistema. Ovde je cilj prikupiti i uskladištiti podatke korišćenjem adekvatne tehnologije baze podataka, koja će znatno olakšati njihovo čuvanje i upravljanje.

1 Izvor: Whitten, J.L., L.D. Bentley, K.C. Dittman (Profesori sa Purdue Univerziteta, West Lafayette, IN), *Systems analysis and design methods*, šesto izdanje, Irwin McGraw-Hill, New York, 2004, adaptirano.

### Pogled vlasnika sistema na sistem podataka

Prosečan vlasnik sistema, obično nije zainteresovan za sirove podatke. Vlasnik je zainteresovan za resurse poslovanja, koje čine kupci, proizvodi, oprema, zgrade, porudžbine ili plaćanja. Njegov domen jeste da za svaki objekat i relacije između objekata identifikuje eventualne probleme, mogućnosti, ciljeve i ograničenja. Zajedno ti podaci čine kompletan kontekst podataka za informacioni sistem.

Na primer, za jedan sistem prodaje, objekti su Kupci, Proizvodi, Prodajni regioni, Porudžbine i Komercijalisti. Za date objekte treba prikupiti i uskladištiti podatke. Slično tome, relacije se mogu izraziti jednostavnim, deklarativnim rečenicama kao što su:

- Kupci dostavljaju porudžbine.
- Porudžbine prodaju proizvode.
- Kupci su locirani po prodajnim regionima.

Vlasnici sistema su veoma retko zainteresovani za detaljnije podatke u vezi objekata i relacija, sem ukoliko nisu i korisnici sistema.

### Pogled korisnika sistema na sistem podataka

Korisnici informacionog sistema su eksperti za podatke koji opisuju poslovni sistem. Oni kao informacioni radnici svakodnevno prikupljaju, skladište, obrađuju, uređuju i koriste te podatke. Za njih su podaci smešteni po fasciklama, knjigama, organizovani po *spreadsheets* datotekama ili uskladišteni unutar baza podataka. Izazov u sistemskoj analizi jeste da se identifikuju i verifikuju zahtevi za podacima. Zahtevi za podacima su neophodni korisnički podaci koji se predstavljaju u obliku objekata, atributa, relacija i pravila.

Razmotrimo sledeći primer. Vlasnik sistema želi da ima sve podatke o objektu Kupac. Korisnik sistema će nas upozoriti o tome da treba razlikovati Potencijalne kupce, Stvarne kupce i Neaktivne kupce, zbog različitih tipova podataka koji opisuju svaki tip kupca. Takođe, korisnik sistema će nam reći koji su to podaci koji se moraju skladištiti za svaki tip kupca. Na primer, objekat Aktivan kupac će imati sledeća svojstva: šifra kupca, naziv, adresa, kredit i tekući bilans kupca. Za opisivanje zahteva za podacima koristi se model podataka.

#### Pogled projektanta sistema na sistem podataka

Dok korisnici sistema definišu zahteve za podacima, projektanti sistema prevode te zahteve u baze podataka, koje će potom biti dostupne putem informacionog sistema. Pogled projektanta sistema na sistem podataka je u obliku šeme baze podataka. Pored šeme baze podataka, projektant bi trebao da odgovori i na sledeća pitanja: Koje baze podataka će se koristiti (npr. Oracle *10g*, IBM *DB2*, Microsoft *SQL Server*, itd)? Na kojoj platformi će se instalirati (npr., UNIX, *Linux, Windows* itd)? Koje tehnologije će se koristiti za smeštanje podataka u OLAP baze podataka i skladišta podataka (npr., ETL - *Extract Transform and Load*)? itd.

#### Pogled graditelja sistema na sistem podataka

Graditelji sistema su najbliži korisnici tehnologije baze podataka. Oni moraju da predstavljaju podatke u veoma preciznoj jezičkoj formi. Najkorišćeniji standardni upitni jezik koji omogućava komunikaciju sa bazom podataka jeste *SQL* (*Structured Query Language*).

## 2.2. Blokovi procesa

Kada inženjeri projektuju nov proizvod, taj proizvod bi trebao da obezbedi odgovarajući nivo funkcionalnosti ili usluge. Potencijalni kupci definišu željenu funkcionalnost proizvoda, a inženjer kreira dizajn proizvoda kako bi obezbedio datu funkcionalnost. Neke procese obavljaju ljudi, a neke mašine, uključujući i računare. Neki procesi se ponavljaju, dok se drugi odvijaju ne tako često ili čak retko. Cilj je da se automatizuju odgovarajući procesi pomoću softver tehnologije (Slika 2.1 – druga kolona). Kao što se vidi sa slike 2.1, različiti *stakeholders*-i imaju različite poglede na procese sistema.

#### Pogled vlasnika sistema na procese sistema

Kao i obično vlasnici sistema su zainteresovani za grubu sliku, odnosno u ovom slučaju za grupe procesa visokog nivoa nazvanih poslovne funkcije. Tipične poslovne funkcije su proizvodnja, računovodstvo i finansije, prodaja i marketing, ljudski resursi i druge.

Projektni timovi često ove funkcije izražavaju u obliku jednostavnog, hijerarhijski dekomponovanog dijagrama. Vlasnici sistema će pružiti informacije o zapaženim problemima, mogućnostima, ciljevima i ograničenjima funkcija. Takođe će želeti da diskutuju o troškovima i koristima oko projektovanja informacionog sistema.

### Pogled korisnika sistema na procese sistema

Korisnici vide odvojene poslovne procese. Poslovni procesi su odvojene aktivnosti koje imaju svoje ulaze i izlaze, kao i vremena početka i završetka. Zahtevi za procesima su često dokumentovani kao aktivnosti, tokovi podataka ili poslova (*workflow*). Specifične politike i procedure obrazuju osnovu ovih poslovnih procesa. Politike su pravila koje se primenjuju kada se završi poslovni proces. Procedure su instrukcije, logika i precizni koraci koji moraju da se prate kako bi se izvršio poslovni proces.

Mnoge kompanije bi trebalo da reprojektuju poslovne procese kako bi eliminisale redundansu i povećale efikasnost poslovanja. Reprojektovanje poslovnih procesa (*Business Process Redesign - BPR*) podrazumeva proučavanje, analizu i reprojektovanje osnovnih poslovnih procesa u cilju smanjenja troškova i poboljšanja vrednosti poslovanja.

Izazov u sistemskoj analizi jeste da se identifikujuju, izraze i analiziraju zahtevi poslovnih procesa. Jedna od metoda sistemske analize, koja to omogućava je model procesa.

### Pogled projektanta sistema na procese sistema

Pogled projektanta sistema na procese je čisto tehnički. Njegove aktivnosti su ograničene specifiranom tehnologijom razvoja aplikacija kao što su *Java*, C#*.net*, J#, *Visaul Basic.net* i dr. Na osnovu datih poslovnih procesa od strane korisnika sistema, projektant mora prvo da odredi koje procese treba automatizovati i kako ih automatizovati na najbolji mogući način. Projektant u dogovoru sa programerima treba da da odgovor na pitanja: Koja okruženja za razvoj aplikacija ili programski jezici će se koristiti za pisanje softvera (npr., IBM *Websphere* sa Javom, *Microsoft Visual Studio*. NET sa *Visual* C#, *Syebase Powerbuilder*, Oracle-ov *Oracle Forms* i dr.)?

Projektant priprema softversku specifikaciju koja će da (1) ispuni poslovne zahteve korisnika sistema i (2) obezbedi dovoljno detalja i konzistencije za prenošenje projekta graditeljima sistema.

Entropija je termin koji se koristi za opisivanje prirodnog i neizbežnog raspadanja sistema. Međutim, entropijom sistema se može upravljati. Današnji alati i tehnike omogućavaju da se projektuje sistem tako da može da se razvija i menja u skladu sa rastom i promenama zahteva.

#### Pogled graditelja sistema na procese sistema

Graditelji sistema prikazuju procese koristeći programske jezike ili okruženja za razvoj aplikacija koji opisuju ulaze, izlaze, logiku i kontrolu. Neki sistemi za upravljanje bazom podataka obezbeđuju sopstvene već ugrađenje programske jezike. Na primer, *Visual BASIC for Applications* (sadržan je u *Access*-u) i *PL-SQL* (koga sadrži *Oracle*). Svi ovi jezici se koriste za pisanje aplikacionih programa. Aplikacioni programi su jezički-zasnovani, mašinski-čitljivi prikazi o tome šta bi računarski proces trebao da radi ili kako bi računarski proces trebao da ostvari svoje zadatke.

## 2.3. Blokovi komunikacije

Započnimo istim primerom kao i kod blokova znanja i procesa. Kada inženjer projektuje jedan nov proizvod, taj proizvod bi trebao da se lako nauči i koristi. Informacioni sistemi moraju da obezbede efektivan i efikasan interfejs korisnicima sistema i drugim poslovnim informacionim sistemima.

Zajednički cilj kod većine organizacija jeste poboljšanje poslovne komunikacije i kolaboracije između zaposlenih i drugih učesnika u sistemu. Poboljšanje komunikacija u informacionim sistemima su usmerene na dva kritična cilja:

- informacioni sistemi moraju da obezbede efektivne i efikasne komunikacione interfejse korisnicima. Takvi interfejsi treba da podstiču timski rad i koordinaciju aktivnosti.
- informacioni sistemi moraju da obezbede efektivne i efikasne interfejse ka drugim poslovnim informacionim sistemima.

### Pogled vlasnika sistema na komunikacije

Vlasnici sistema su zainteresovani za globalnu sliku sistema i za njegove troškove i koristi. U ranoj fazi sistemskog razvoja projekta, vlasnici sistema treba da odrede:

- Sa kojim poslovnim jedinicama, zaposlenima, klijentima i spoljnim objektima će novi sistem biti u interakciji?
- Gde su locirane poslovne jedinice, zaposleni, klijenti i drugi poslovni objekti?
- Da li će sistem komunicirati sa drugim informacionim sistemima?

Odgovori na ova pitanja će pomoći u definisanju domena komunikacija u projektu razvoja informacionog sistema. Vlasnici sistema će identifikovati i analizirati relevantne probleme, mogućnosti i ograničenja.

### Pogled korisnika sistema na komunikacije

Korisnici sistema su zainteresovani za korisnički interfejs informacionog sistema. Korisnički interfejs definiše kako korisnici sistema pristupaju informacionom sistemu da bi uneli podatke, pravili upite, dobili izveštaje i koristili *help* (pomoć). Jedan od standarda korisničkog interfejsa jeste grafičko korisnički interfejs (*GUI – Graphical User Interface*). Tu se razlikuju Windows forme od Web formi u zavisnosti od toga da li aplikaciju treba da podržavaju i *browser*-i.

### Pogled projektanta sistema na komunikacije

Donekle su pogledi projektanta i korisnika sistema slični, jer su i jedni i drugi uključeni u dizajniranje ekrana, ulaznih i izlaznih podataka. Međutim, dok su korisnici sistema zainteresovani za oblik i sadržaj, projektant sistema se bavi konzistencijom, kompletnošću, korisničkim dijalogom i interfejs tehnologijama. Projektant će se pozabaviti i sledećim pitanjima: Kako će se razvijati korisnički interfejsi, da li sa MS Windows ili Web komponentama (npr., xHTML editor kao što je *Macromedia Dreamweaver*, *portal* IBM *Websphere* i dr)? Kako će se razmenjivati podaci između različitih informacionih sistema (npr., *data broker* IBM MQ *Messaging* ili XML)?

Korisnički dijalog u interakciji sa aplikacionim programom, opisuje kako se korisnik pomera sa ekrana na ekran kako bi obavio zadatak. Projektant sistema crta interfejs šemu, koja definiše osobine interfejsa, stanja sistema, događaje koji menjaju stanje sistema i odzive na događaje.

### Pogled graditelja sistema na komunikacije

Graditelji sistema izgrađuju, instaliraju, testiraju i implementiraju korisničke i sistemske interfejse. Interfejs tehnologije su mahom ugrađene u okruženje razvoja aplikacija (*Application Development Environment* – ADE) kao što je na primer Visual Studio .NET.

Jedna od interfejs tehnologija je *middleware*. *Middleware* je koristan softverski sloj koji se nalazi između aplikacionog i sistemskog softvera, a služi da transparentno integriše različite tehnologije kako bi one mogle da funkcionišu. Jedan primer *middleware* jeste povezanost otvorenih baza podataka (*Open Database Connectivity* – ODBC*).* ODBC alati dozvoljavaju aplikacionim programima da rade sa različitim sistemima za upravljanje bazama podataka (*Database Management Systems* – DBMS) bez potrebe da budu prerađeni usled nijansi i različitosti sistema za upravljanje bazama podataka. XAML (*eXtensible Application Markup Language*)[1] postaje standard za komunikaciju između sistema.

1 XAML (izgovara: zammel) je deklarativni XML zasnovan jezik koji definiše objekte, njihova svojstva i međusobne veze u XML-u. XAML dinamički kreira UI (user interface) za Windows Presentation Foundation (WPF). WPF je nova prezentacija API (Application Programming Interface) u .NET Frameworku 3.0 (bivši WinFX). WPF API ima širok domen funkcionalnosti, od Windows kontrola kao što su 3D grafika, specijalni efekti i multimedija.

# 3. Poslovni informacioni sistemi

Imajući u vidu prethodno obrazložen pojam informacionih sistema dolazi se do opšte definicije poslovnih informacionih sistema. **Poslovni informacioni sistemi su informacioni sistemi koji podržavaju poslovne funkcije i obezbeđuju poslovnu inteligenciju i analitiku.** Na primer, organizacija može koristiti određeni informacioni sistem kako bi upravljala aktivnostima prodaje, marketinga i ljudskim resursima. Poslovni informacioni sistemi se oslanjaju na pet osnovih resursa:

- **Ljudski resursi** – uključuju korisnike IS i one koji razvijaju, održavaju i rukuju sistemom.
- **Hardverski resursi** – uključuju sve tipove mašina, kao što su telefoni, ruteri, DVD-jevi, PDA (*Personal Digital Assistant*), računari i dr.
- **Softverski resursi** – uključuju računarske programe, priručnike, politiku kompanije i dr.
- **Komunikacioni resursi** – uključuju mreže i neophodan hardver i softver koji ih podržava.
- **Resursi podataka** – opisuju sve podatke kojima organizacija ima pristup, bez obzira na njihovu formu. Uključuje baze podataka, fajlove, dosijee, fascikle i dr.

Tipične poslovne funkcije u nekom preduzeću su računovodstvo, pro-izvodnja, logistika, marketing, ljudski resursi i dr. U većini odeljenja unutar organizacije postoje sopstveni informacioni sistemi, kao na primer računovodstveni informacioni sistem, marketing IS, kadrovski IS itd. Kada sistem nije integrisan, donosioci odluka će vrlo teško izvući informacije iz višestrukih sistema. Ukoliko odluke nisu zasnovane na integrisanoj informaciji, onda preduzeće ne koristi sve prednosti koje takva informacija može da obezbedi.

## 3.1. ERP sistemi

Za većinu ljudi termin poslovni informacioni sistem podrazumeva ERP softver, mada on inkorporira i CRM sisteme, SCM, PLM, SRM, alate za optimizaciju poslovanja i dr.

**ERP (*Enterprise Resource Planning*) sistem je poslovni softver koji omogućava organizacijama da automatizuju i integrišu glavne poslovne procese, dele opšte podatke i pristupaju informacijama u realnom okruženju.**

Preduzeća moraju efikasno da kontrolišu osnovne funkcije u svom poslovanju, istovremeno, da uočavaju potrebe i planiraju strateške inicijative primenjujući nove tehnologije. ERP pomaže organizacijama koje se bave lancima nabavke, upravljanjem zalihama, upravljanjem porudžbinama klijenta, planiranjem proizvodnje, računovodstvom, upravljanjem ljudskim resursima i drugim poslovnim funkcijama.

ERP tržište je trenutno jedno od najbrže rastućih tržišta u softverskoj industriji. ERP rešenja predstavljaju značajnu poslovnu investiciju koji osiguravaju konkurentnost, osetljivost na zahteve klijenata i fleksibilnost u vođenju poslovanja u globalnoj ekonomiji. ERP sistemi uvode „najbolje poslovne prakse“ koje se jednostavno definišu kao „najbolji način izvođenja procesa“. Implementiranje ERP rešenja omogućava kompanijama da izvrše reinženjering poslovne prakse ka „najboljoj praksi“ i integrišu informacione resurse.

S obzirom na brzi porast integrisanih usluga, zahtevi projektovanja postaju sve složeniji i zahtevniji. Uspeh primene ERP rešenja će zavisiti od efektivnog menadžmenta, organizacionih promena, primene naprednih tehnologija i neophodnog nivoa znanja i veština. Međutim, mnogi projekti su propali usled toga što su ili vođeni tehnološkim motivacijama umesto poslovnim opravdanjem ili nisu iskorišćavali sve potencijale implementiranog ERP rešenja.

ERP softverski paketi su predstava poslovnih procesa. Ukoliko softver ne odslikava postojeće poslovne procese, onda bi trebalo promeniti ili poslovne procese ili sam softver. Promene softvera zahtevaju dodatno vreme i svaki put kada prodavac nadograđuje (*upgrade*) softver, neophodno je ponovo uraditi kastomizaciju (prilagođavanje) softvera. S druge strane, ukoliko preduzeće menja svoje poslovne procese onda i ljudi moraju da se prilagođavaju, a ljudi ne vole promene. Ukoliko se promene ne prihvate dobro, zasigurno je da će implementacija softvera biti osuđena na propast.

Adekvatno uvođenje poslovnog informacionog sistema omogućava vođenje poslovanja efektivnije i efikasnije, kao i brže reagovanje na promene u okruženju čime se ostvaruje konkurentska prednost na tržištu. **Poslovni IS imaju za cilj da skladište poslovne informacije samo jedanput, u formi u kojoj se omogućava pristup od strane više različitih korisnika, a u cilju donošenja različitih tipova odluka.**

Na domaćem tržištu, u poslednjih nekoliko godina, najzastupljeniji ERP sistemi su: SAP R/3, *Oracle Applications*, *Microsoft Dynamics NAV*, *HansaWorld*, *Datalab Pantheon*™ i mnogi drugi.

### Evolucija ERP sistema

Šezdesetih godina većina softverskih paketa je omogućavala samo kontrolu zaliha, da bi se 70-tih godina uveli MRP (*Material Requirements Planning*) sistemi koji su se koristili za raspoređivanje proizvodnje i uključivali su sastavnicu materijala sa listom materijala neophodnih za proizvodnju svakog proizvoda. Kasnije, MRP sistemi su poboljšani uvođenjem alata za planiranje prodaje, obradu porudžbina klijenata i planiranje kapaciteta što je obezbeđivalo input u redosled proizvodnje poznatijeg kao zatvoreni (*closed-loop*) MRP. Osamdesetih godina, MRPII sistemi uključuju sisteme za vođenje finansija i računovodstva integrisanih sa sistemima za upravljanje materijalima i proizvodnjom. MRPII vodi ka integrisanim poslovnim sistemima koji prevode zahteve za materijalima i kapacitetima u proizvodnji u finansijske informacije. Devedesetih godina, ERP sistemi obezbeđuju integraciju svih informacionih tokova u kompaniji – računovodstvo, ljudski resursi, upravljanje lancem snabdevanja, informacije o klijentima i dr.

ERP sistemi nameću pristup „integrisanih sistema“ uspostavljanjem opšteg skupa aplikacija koje podržavaju poslovne operacije. Kao što se može videti iz tabele 3.2., ERP sistemi prevazilaze neefikasnosti izolovanih sistema i neintegrisanih podataka, omogućavajući integrisane podatke koje podržavaju višestruke poslovne funkcije.

## 3.2. Koristi od ERP sistema

Sa sveukupnog poslovnog gledišta, ERP sistemi dostižu brojne važne ciljeve, kao što su ubrzani protok informacija, minimalno vreme odziva na zahteve klijenata i dobavljača, odlučivanje na nižim nivoima i jedinstvenu, pouzdanu i blagovremenu informaciju donosiocima odluka. Sve ovo dovodi do smanjivanja troškova, zaliha i boljih performansi rada.

Tabela 3.1. Evolucija ERP sistema

| Tipovi sistema | Godine | Svrha | Sistemi |
|---|---|---|---|
| Sistemi za praćenje porudžbina | 1960-te | Koriste istorijske podatke kako bi prognorzirali buduće zalihe robe. Kada zalihe padnu ispod predodređenog nivoa, poručuju se dodatne zalihe. | Projektovani da upravljaju obimnom proizvodnjom nekolicine proizvoda sa konstantnom tražnjom i fokusom na troškove. |
| Sistemi za planiranje zahteva za materijalima (*Materials requirement planning* - MRP) | 1970-te | Planiranje proizvodnje proizvoda i poručivanje zaliha prema tražnji. | Fokus na marketingu, naglašava potrebu za planiranjem i integracijom proizvodnje |
| Planiranje resursa proizvodnje (*Manufacturing resource planning* - MRP-II) sistemi | 1980-te | Dopunjuju se sa planiranjem kapaciteta, pravljenjem rasporeda i nadgledanjem izvršavanja proizvodnih planova. | Fokus je na kvalitetu; proizvodna strategija se fokusira na kontrolu procesa, smanjenje troškova i detaljne izveštaje o troškovima. |
| MRP-II sa proizvodnim izvršnim sistemima (*Manufacturing execution systems* – MES) | 1990-te | Obezbeđuju mogućnost prilagođavanja rasporeda proizvodnje prema potrebama klijenta. | Fokus na sposobnosti kreiranja i prilagođavanja novih proizvoda i usluga prema specifičnim potrebama klijenata. |
| ERP (*Enterprise resource planning*) sistemi | kasnije 90-te i dalje | Integrisana proizvodnja sa ostalim poslovnim funkcijama u preduzeću. | Integriše dobavljače, poslovne funkcije i klijente. |

ERP sistemi su projektovani tako da obezbede poslovnu korist u prodaji i distribuciji, proizvodnji, računovodstvu, na polju usluga i troškova. Istraživanja švedskih proizvodnih kompanija pokazuju da ERP sistemi obezbeđuju blagovremene informacije, povećane interakcije kroz preduzeće, brže projektovanje i proizvodnju proizvoda i poboljšani menadžment poručivanja.

ERP sistemi mogu da ubrzaju poslovne procese, smanje vreme ciklusa i troškove poslovnih procesa. Oni služe kao osnova za elektronsko poslovanje (*eBusiness*) jer obezbeđuju funkcije *front-office*-a koji omogućavaju klijentima da postave i prate svoje porudžbine putem Web-a.

Tabela 3.2. Pogled na sisteme pre i posle ERP sistema

| | **Pre ERP sistema** | **Posle ERP sistema** |
|---|---|---|
| Informacioni sistemi | Izolovani sistemi. | Integrisani sistemi. |
| Koordinacija | Nedostatak koordinacije između poslovnih funkcija (npr., proizvodnja i prodaja) | Podržava koordinaciju između poslovnih funkcija. |
| Baze podataka | Neintegrisani podaci; podaci imaju različita značenja; nekonzistentna definicija podataka. | Integrisani podaci; podaci imaju isto značenje. |
| Održavanje | Sistemi se pojedinačno održavaju; nekonzistentni rezultati; visoki troškovi održavanja pojedinačnih sistema. | Uniformno održavanje; promene utiču na sve sisteme. |
| Interfejsi | Otežano upravljanje interfejsima između sistema. | Zajednički interfejsi. |
| Informacija | Redundantna, nekonzistentna informacija. | Konzistentna, blagovremena i tačna informacija. |
| Arhitektura sistema | Monolitna arhitektura sistema. | Višeslojna arhitektura. |
| Procesi | Nekompatibilni (neusaglašeni) procesi. | Konzistentni poslovni procesi koji se zasnivaju na modelu informacija. |
| Aplikacije | Različite aplikacije (npr., mnogi različiti sistemi za nabavku). | Jedna aplikacija (npr., zajednički sistem nabavke). |

Aplikacije kao što su Upravljanje odnosima sa klijentima (*Customer Relationship Management* – CRM) oslanjaju se osnove ERP sistema. Najčešći razlozi zašto se organizacije opredeljuju za ERP sisteme je zamena izolovanih sistema, potreba za standardizacijom, važnost sticanja konkurentne prednosti, potreba za standardima kvaliteta i potreba za poboljšanjem interakcija sa poslovnim partnerima i klijentima.

## 3.3. ERP moduli

ERP sistemi treba da omoguće odabir odgovarajuće oblasti primene, kao što su: upravljanje finansijama, proizvodnja, distribucija, upravljanje odnosima, upravljanje servisom, e-trgovina, analitika itd.

Trenutno, vodeći ERP vendori su SAP, Oracle i Microsoft koji podržavaju glavne funkcionalne oblasti poslovanja, uključujući obradu porudžbina, nabavke, planiranje proizvodnje, računovodstva, ljudske resurse i dr. Prema statistici, najčešće implementirani moduli su moduli za finansije i računovodstvo.

Tabela 3.3. ERP moduli koje podržavaju glavni vendori

| Funkcije | SAP | Oracle | Microsoft |
|---|---|---|---|
| Obrada porudžbina | *Sales and Distribution* (SD) | *Marketing Sales Supply Chain* | *Sales and Marketing* |
| Nabavka | *Materials Management* (MM) | *Procurement* | *Purchase* |
| Planiranje proizvodnje | *Production Planning* (PP) | *Manufacturing* | *Manufacturing* |
| Finansijsko računovodstvo | *Financial Accounting* (FA) | *Financials* | *Financial Management* |
| Ljudski resursi | *Human Resources* (HR) | *Human Resources* | *Human Resources* |

U proceni ERP rešenja, mnoge organizacije razmatraju alternative projektovanja gde svaka od njih ima svoju cenu kao i prednosti i nedostatke. Kompletna implementacija gotovog ERP rešenja koje nudi vendor, tzv. „vanila“ implementacija, je skupa i zahteva dosta vremena, ali nudi prednosti totalne integracije i reinženjering poslovnih procesa. Implementacija pojedinih ERP modula, na primer modul finansija i računovodstva, manje košta i zahteva manje vremena, ali postoji nedostatak totalne integracije podataka kroz višestruke funkcionalne oblasti poslovanja. Alternativa izgradnje ERP sistema u organizaciji (*in-house*), donosi brojne rizike i najduže traje. Prednost ovog pristupa je izgradnja softvera koji odslikava jedinstvene procese organizacije. Poslednja alternativa je održavanje postojećih izolovanih sistema unutar organizacije. Ova alternativa znatno umanjuje konkurentnu prednost organizacije na tržištu, jer jedan ERP sistem obezbeđuje integraciju podataka i standardizaciju procesa koji odslikavaju „najbolje prakse“.

Na osnovu brojnih faktora kao što su integrisanost podataka, troškovna efektivnost, konkurentnost na tržištu, poslovni uticaj i vreme, većina organizacija zaključuje da je „vanila“ ili parcijalna ERP implementacija bolja nego razvoj u kući ili održavanje postojećih izolovanih sistema. Tabela 3.4 prikazuje mogući scenarijo ERP implementacije prema „Fortune 500“.

Tabela 3.4. Meni ERP Alternativa

| Opcije | Troškovi i vreme | Prednosti | Nedostaci |
|---|---|---|---|
| Vanila ERP implementacija | 150$ miliona preko 5 godina | Kompletna standardizacija poslovnih procesa zasnovanog na najboljoj praksi. | Konkurenti mogu da nabave isto ERP rešenje. |
| Parcijalna ERP implementacija | 108$ miliona preko 2-3 godine | Parcijalne promene u poslovnim procesima. | Nedostatak totalne integracije |
| Razvoj u kući | 240 miliona $ preko 7-10 godina | Jedinstveni ERP sistem. | Dugoročna analiza i projektovanje procesa; visoki troškovi. |
| *Status quo* | Nema troškova, ali ni dobiti | Nema promena u poslovnim procesima. | Mogu da dovedu do slabljenja konkurentne pozicije na tržištu. |

Jedan od glavnih zahteva u opravdanju investiranja u ERP sisteme je ocena merljivih (opipljivih) i nemerljivih koristi. Prema istraživanjima *Benchmarking Partners*, Inc, za *Deloitte Consulting*, najvažnija merljiva korist nakon uvođenja ERP sistema je smanjenje zaliha. Rezultati istraživanja su prikazani u Tabeli 3.5.

Tabela 3.5. Merljive koristi sa ERP sistemom

| Merljive koristi | % kompanija |
|---|---|
| Smanjenje zaliha | 32 |
| Redukcija kadrova | 27 |
| Poboljšana produktivnost | 26 |
| Poboljšano upravljanje porudžbinama | 20 |
| Smanjenje finansijskog ciklusa | 19 |
| Smanjenje troškova IT-a | 14 |
| Smanjenje troškova nabavke | 12 |
| Poboljšano upravljanje gotovinskim novcem | 11 |
| Povećanje dobiti/profita | 11 |
| Smanjenje troškova transporta/logistike | 9 |
| Smanjeno održavanje | 7 |
| Poboljšanje *online* isporuke | 6 |

*Izvor*: Fryer, Bronwyn, „The ROI Challenge“, CFO, Septembar, 1999, str. 90.

Najveći faktor nemerljivih koristi je raspoloživost informacija (Tabela 3.6). Informacije omogućavaju menadžerima da donesu bolje odluke.

Tabela 3.6. Nemerljive koristi sa ERP-om

| Nemerljive koristi | % kompanija |
|---|---|
| Informacija/vidljivost | 55 |
| Novi/poboljšani procesi | 24 |
| Osetljivost na zahteve klijenata | 22 |
| Integracija | 13 |
| Standardizacija | 12 |
| Fleksibilnost | 9 |
| Globalizacija | 9 |
| Y2K | 8 |
| Poslovne performanse | 7 |
| Lanac snabdevanja/tražnje | 5 |

*Izvor*: Fryer, Bronwyn, „The ROI Challenge", CFO, Septembar, 1999, str. 90.

## 3.4. Analiza troškova i koristi za ERP

Troškovi implementiranja ERP rešenja uključuju hardver, softver, tehničku podršku, projektni menadžment, tim koji radi na implementiranju rešenja, spoljne konsultante, obuke i dr. Jedan od metoda za analizu troškova i koristi (*cost-benefit*) ERP projekta je neto sadašnja vrednost (*net present value*). Neto sadašnja vrednost uzima u obzir vremensku vrednost novca, na primer, ukoliko firma investira $1.000.000, pri interesnoj stopi od 10%, očekuje da za pet godina dobije $1.610.510. Kada se vrši procena ERP projekta, tada se uzima vremenski okvir od 5 godina, jer se smatra da će minimum tri godine proći na implementiranje rešenja.

Na tabeli 3.7., se prikazuje analiza troškova i koristi metodom neto sadašnje vrednosti za jedan ERP projekat. Početni (*start-up*) troškovi uključuju investiranje u ERP softver. U ovom slučaju, cena softvera je $2.420.000. Investicija u hardver iznosi $1.850.000, koja uključuje servere, mrežne uređaje u klijent/server okruženju. Troškovi konsultantskog vremena provedenog na instaliranju i konfigurisanju softvera iznosi dodatnih $3.000.400.

Tabela 3.7. Neto sadašnja vrednost jednog ERP projekta [6]

| | Godina 0 | Godina 1 | Godina 2 | Godina 3 | Godina 4 | Godina 5 |
|---|---|---|---|---|---|---|
| Softver | 2.420.000 | | | | | |
| Licenciranje softvera | | 220.000 | 220.000 | 220.000 | 220.000 | 220.000 |
| Hardver | 1.850.000 | | | | | |
| Konsalting | 3.000.400 | | | | | |
| Obučavanje | 1.280.000 | | | | | |
| Tim za implement. | 400.000 | 400.000 | 400.000 | 400.000 | | |
| Ukupni troškovi | 8.950.400 | 620.000 | 620.000 | 620.000 | 220.000 | 220.000 |
| Uštede | 0 | | | | | |
| Smanjeni troškovi zaliha | | 2.750.000 | 2.750.000 | 2.750.000 | 2.750.000 | 2.750.000 |
| Smanjeni admin. troškovi | | 1.250.000 | 1.250.000 | 1.250.000 | 1.250.000 | 1.250.000 |
| **Nemerljive koristi:** | | | | | | |
| Ukupna ušteda | 0 | 4.000.000 | 4.000.000 | 4.000.000 | 4.000.000 | 4.000.000 |
| Neto bilans | -8.950.400 | 3.380.000 | 3.380.000 | 3.380.000 | 3.780.000 | 3.780.000 |
| DCF faktor | 1.000 | 0.909 | 0.826 | 0.751 | 0.683 | 0.621 |
| Diskontovani bilans | -8.950.400 | 3.072.420 | 2.791.880 | 2.538.380 | 2.581.740 | 2.347.380 |
| Kumulativni diskontni bilans | -8.950.400 | -5.877.980 | -3.086.100 | -547.720 | 2.034.020 | 4.381.400 |

Vreme projektnog menadžmenta, internog tima i upravnog odbora projekta je procenjeno da bude $400.000, a inicijalno vreme obuke je $1.280.000. Ove cifre su preuzete sa istraživanja vođenih u Americi i Švedskoj (Olhager i Selldin, 2003; Mabert et al., 2000).

Periodični troškovi uključuju troškove licenciranja softvera, održavanja, vremena projektnog menadžmenta, internog tima na projektu i konsultanata koji se koriste na periodičnoj osnovi. Na primer, licenciranje softvera je 10% od troškova originalno kupljenog softvera, što iznosi $220.000 godišnje. Vremena tima koji radi na implementaciji iznosiće $400.000 godišnje.

Ukoliko se pretpostavi da implementacija projekta traje minimum tri godine, onda se očekuje da se dobit neće realizovati pre četvrte godine projekta. U našem primeru, merljive koristi uključuju smanjenje zaliha koje predstavljaju uštedu od $2.750.000, godišnje. Zatim, redukcija kadrova predstavlja dodatnu uštedu od $1.250.000, godišnje. Nemerljive koristi mogu biti povećanje morala zaposlenih, zadovoljstvo klijenata, smanjenje dupliciranog održavanja raznih baza podataka, međutim ove koristi ne mogu biti prikazane ciframa.

Sa tabele 3.7., se može videti da će predloženi ERP projekat imati pozitivni diskontni bilans u prvoj godini, dok će u četvrtoj godini investicija

uveliko isplatiti samu sebe gde će dostići kumulativni diskontni bilans od $2.034.020. Na osnovu ove analize, može se zaključiti da je investiranje u ERP sistem, mudra investicija.

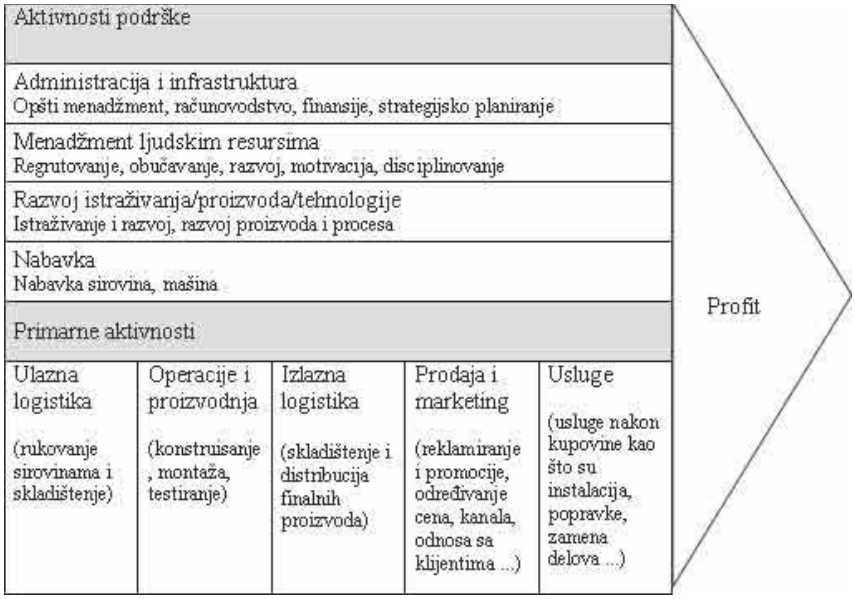
## 3.5. Poslovni informacioni sistemi i strateška prednost

Poslovni informacioni sistemi igraju krucijalnu ulogu u sticanju i održavanju konkurentske prednosti u odnosu na druge organizacije iz iste industrijske branše. Da bi stekle i održale konkurentnu prednost organizacije moraju da usvoje tri osnovne strategije: vođstvo u troškovima (*cost leadership*), diferencijacija (*differentiation*) ili inovacija (*innovation*). Vođstvo u troškovima podrazumeva obezbeđivanje dobara i usluga po najnižim mogućim troškovima. Diferencijacija podrazumeva kreiranje takvog proizvoda koji se razlikuje bilo po kvalitetu, karakteristikama ili drugim specifičnostima od proizvoda drugih konkurenata. Inovacija se bavi pronalaženjem novih načina pristupa organizacionim aktivnostima. Inovacija podrazumeva poboljšanje postojećeg proizvoda, kreiranje potpuno novog proizvoda, poboljašnje proizvodnog procesa ili ulazak na novo tržište.

Analiza lanca vrednosti organizacije može da ukaže na oblasti koje mogu da obezbede organizaciji konkurentnu prednost. **Porter-ov koncept lanca vrednosti podrazumeva seriju povezanih aktivnosti koje dodaju vrednosti organizacionim proizvodima ili uslugama.**

Da bi shvatili analizu lanca vrednosti neophodno je razmotriti organizacioni lanac snabdevanja. Upravljanje lancem snabdevanja (*Supply chain management* – SCM) podrazumeva koordinaciju svih aktivnosti snabdevanja jedne organizacije, od njenih dobavljača preko proizvodnje dobara i usluga do isporuke klijentima. Lanac vrednosti opisuje različite aktivnosti dodavanja vrednosti koje povezuju stranu nabavke sa stranom tražnje. Tradicionalna analiza lanca vrednosti (Slika 3.1) razlikuje primarne aktivnosti koje direktno doprinose isporuci dobara i usluga klijentima (na primer, logistika, proizvodnja, marketing, isporuka kupcima, podrška i servisiranje nakon prodaje) i aktivnosti podrške koji obezbeđuju infastrukturu koja omogućava odvijanje primarnih aktivnosti. Aktivnosti podrške su finansije, ljudski resursi, informacioni sistemi i dr.

Primarne aktivnosti su vidljive potrošaču, dok aktivnosti podrške nisu. Primarne aktivnosti podrazumevaju proces transformacije sirovina ili informacija u proizvode ili usluge, a zatim njihovu isporuku klijentima i partnerima kroz prodaju i marketing.




Slika 3.1. Porter-ov model lanca vrednosti[1]

Aktivnosti podrške olakšavaju nesmetano funkcionisanje primarnih aktivnosti i imaju indirektan odnos na proces dodavanja vrednosti na proizvod. Oni uključuju:

- administraciju i infrastrukturu neophodnu za obavljanje bilo kog poslovanja, kao što su računovodstvo i finansije;
- upravljanje ljudskim resursima podrazumeva prijem, obučavanje, motivisanje i rapoređivanje odgovarajućih ljudskih resursa u organizaciji;
- razvoj istraživanja/proizvoda/tehnologija poboljšavaju postojeći proizvod i reprojektuju ga na način kako bi se održala njegova privlačnost i osvojila nova tržišta;
- nabavka podrazumeva nabavku materijala po pristupačnim cenama.

1 Porter, M.,Competitive Advantage: Creating and Sustaining Superior Performance, Free Press, New York, 1998, adaptirano.

Što je veća razlika između dodatih vrednosti proizvoda i troškova organizacije, veći je i profit. Lanac vrednosti svake organizacije je inkorporiran unutar većeg sistema vrednosti. Ovo uključuje lance vrednosti njihovih dobavljača, distributera i trgovaca na malo. Lanci vrednosti svih učesnika bi trebalo da budu koordinirani sa sistemom, da rade efikasno i da isporučuju krajnje proizvode. Ukoliko bilo ko od partnera ne isporuči odgovarajuću vrednost i proizvode, čitav lanac može da se slomije, a ceo sistem čak i da propadne. Ovo je jedan od mnogih razloga zašto poslovni informacioni sistemi igraju odlučujuću ulogu u sticanju konkurentne prednosti organizacije.

## Rezime

1. Sistem se najopštije definiše kao skup objekata (entiteta) i njihovih međusobnih veza usmerenih ka ostvarivanju zajedničkog cilja.
2. Informacioni sistem (IS) je uređeni i integrisani skup ljudi, podataka, procesa, interfejsa, mreža i tehnologija koja su u međusobnoj korelaciji u cilju podrške i poboljšanja svakodnevnih poslovnih operacija i podrške menadžmentu u rešavanju poslovnih problema, planiranja, upravljanja, predviđanja, koordinisanja i donošenja odluka.
3. Informacione tehnologije (IT) opisuju kombinaciju računarske tehnologije (*hardware* i *software*), telekomunikacione tehnologije, *netware, groupware* i *humanware.*
4. CASE *alati* su programi (softveri) koji automatizuju i podržavaju jednu ili više faza životnog ciklusa razvoja sistema.
5. Arhitektura informacionih sistema obezbeđuje jedinstveni okvir (*framework*) po kome će različiti ljudi sa različitim pogledima organizovati fundamentalne blokove razvoja informacionih sistema:
   - vlasnici i korisnici sistema se fokusiraju na tri glavna cilja, a to su: poboljšanja kod poslovnog znanja, poslovnih procesa i poslovnih komunikacija.
   - projektanti i graditelji sistema se fokusiraju na tehnologije kako bi ispunili postavljene ciljeve.
6. Poslovni informacioni sistemi su informacioni sistemi koji podržavaju poslovne funkcije i obezbeđuju poslovnu inteligenciju i analitiku.

7. ERP (*Enterprise Resource Planning*) predstavlja softver koji integriše i automatizuje sve aspekte poslovanja preduzeća i omogućava svim zaposlenima dobijanje i deljenje pouzdanih informacija.
8. Poslovna dobit uvođenja ERP rešenja uključuje poboljšanu pristupačnost informacijama, pristup podacima u realnom vremenu kroz sva odelenja u organizaciji, poboljšani vremenski ciklus porudžbina, smanjeni operativni troškovi, niže nivoe zaliha i dr.
9. Poslovni informacioni sistemi igraju krucijalnu ulogu u sticanju i održavanju konkurentske prednosti u odnosu na druge organizacije iz iste industrijske branše. Analiza lanca vrednosti organizacije može da ukaže na oblasti koje mogu da obezbede organizaciji konkurentnu prednost.
10. Porter-ov koncept lanca vrednosti podrazumeva seriju povezanih aktivnosti koje dodaju vrednosti organizacionim proizvodima ili uslugama. Tradicionalna analiza lanca vrednosti razlikuje primarne aktivnosti koje direktno doprinose isporuci dobara i usluga klijentima (npr., logistika, proizvodnja, marketing, isporuka kupcima, podrška i servisiranje nakon prodaje) i aktivnosti podrške koji obezbeđuju infrastrukturu koja omogućava odvijanje primarnih aktivnosti. Aktivnosti podrške su finansije, ljudski resursi, informacioni sistemi i dr. Što je veća razlika između dodatih vrednosti proizvoda i troškova organizacije, veći je i profit.

## Pitanja za razgovor i vežbe

1. Posmatrajući sistem univerziteta, razmotriti sledeće:
    a. Koji su ciljevi tog sistema?
    b. Identifikovati inpute, procese i autpute.
    c. Koje su povratne informacije, koje utiču na inpute ovog sistema?
2. Kakva je razlika između podatka, informacije i znanja?
3. Šta su informacioni sistemi i koja je razlika između informacionih sistema i informacionih tehnologija?
4. Šta su okruženja za razvoj aplikacija i koje alate uglavnom obezbeđuju?

5. Kako projektanti definišu poslovne zahteve za podacima?
6. Šta su to poslovni informacioni sistemi i na koje osnovne resurse se oslanjaju?
7. Šta omogućava uvođenje ERP rešenja?
8. U današnjem konkurentnom okruženju kritični faktor opstanka organizacije je primena informacionih sistema. Prodiskutujte sledeće:
    a. Kako Porterov model lanca vrednosti i konkurentna strategija utiču na sticanje prednosti na tržištu?
    b. Na koji način informacioni sistemi podržavaju takvu strategiju?

## Reference


[1] Bocij. P., D. Chaffey, A. Greasley i S. Hickie, *Business Information Systems – Technology, Development and Management for the e-business*, drugo izdanje, Prentice Hall, Harlow. 2003.

[2] Dunn., L.C., J.O. Cherrington i A.S. Hollander, *Enterprise Information Systems – a pattern-based approach*, treće izdanje, McGraw-Hill, Irwin, 2005.

[3] Njeguš., A. „Sistemska analiza i projektovanje informacionih sistema“, skripta, Megatrend univerzitet, Beograd, 2002.

[4] Njeguš., A., A. Veljović, *Internet poslovno programiranje*, Megatrend univerzitet, Beograd, 2004.

[5] Porter, M.,*Competitive Advantage: Creating and Sustaining Superior Performance,* Free Press, New York, 1998.

[6] Sumner., M. *Enterprise Resource Planning*, Prentice Hall, New Jersey, 2005.

[7] Whitten J.L., L.D. Bentley i K.C. Dittman, *Systems Analysis and Design Methods,* šesto izdanje, McGraw-Hill, Irwin, 2004.

[8] XAML, http://msdn2.microsoft.com/en-us/library/ms752059.aspx, MSDN Library, Microsoft.

# DEO II: RAZVOJ POSLOVNIH INFORMACIONIH SISTEMA

## Ciljevi poglavlja

Tri poglavlja u okviru drugog dela ove knjige vas uvode u metodološke osnove razvoja informacionih sistema, prikazuju aktivnosti razvoja poslovnih rešenja prema MSF okviru i implementacije gotovih poslovnih rešenja. Sistemska analiza je najkritičnija faza projekta. Tokom sistemske analize proučavaju se postojeći poslovni sistemi, razumeju njihovi problemi, definišu ciljevi za poboljšanje sistema i definišu detaljni poslovni zahtevi koje mora da ispuni svako tehničko rešenje. Projektovanje sistema i implementacija novog sistema će zavisiti upravo od kvalitetno obavljene sistemske analize. Podpoglavlja u okviru sistemske analize vas podučavaju određenim veštinama sistemske analize sa naglaskom na logičko modeliranje sistema. Projektovanje sistema podrazumeva pripremu računarski zasnovanih specifikacija koje će ispuniti zahteve specifiranih tokom sistemske analize i podrazumeva izgradnju prototipova sistema. Finalne faze razvoja sistema se bave stabilizacijom i uvođenjem finalnog sistema u rad.

Nakon proučavanja ovog poglavlja, znaćete da:

- analizirate probleme i modelujete sistemske zahteve funkcionalnim i objektno-orijentisanim tehnikama;
- analizirate izvodljivost rešenja i upravljate rizicima na projektu;
- kreirate funkiconalnu specifikaciju poslovnog rešenja;
- analizirate modele podataka primenom normalizacije;
- dokumentujete konceptni, logički i fizički dizajn pomoću UML-a;
- objasnite discipline *Microsoft Solution Framework*-a;
- objasnite korake implementacije ERP rešenja primenom ASAP metodologije.

# 4. Metodološke osnove razvoja informacionih sistema

Metodologije razvoja informacionih sistema podrazumevaju fizičku implementaciju logičkog životnog ciklusa koji uključuje:

- aktivnosti za svaku fazu,
- individualna i grupna pravila za svaku aktivnost,
- standarde kvaliteta za svaku aktivnost i
- alate i tehnike koje treba da budu upotrebljene za svaku aktivnost.

Životni ciklus razvoja sistema (*Systems Development Life Cycle* – SDLC) je konceptualni model koji se sastoji od sekvencionalnih procesa koji opisuju faze projekta razvoja informacionih sistema počevši od inicijalne studije izvodljivosti do održavanja finalnih aplikacija. SDLC je jedan logički proces po kome analitičari, softverski inženjeri, programeri i krajnji korisnici razvijaju informacione sisteme kako bi rešili poslovne probleme i ispunili zahteve. Namena životnog ciklusa je da planira, izvršava i kontroliše sistemski razvoj projekta. On definiše faze i zadatke koji su osnovni za sistemski razvoj, bez obzira o kom tipu i vrsti sistema je reč. Sve metodologije su izvedene iz ovog logičkog procesa rešavanja problema odnosno iz životnog ciklusa razvoja sistema. Razvijene su različite SDLC metodologije koje prate procese SDLC-a, kao što su model vodopada (izvorni SDLC metod), model brzog razvoja aplikacija (*Rapid Application Development* - RAD), spiralni model (*spiral model*) i mnoge druge.

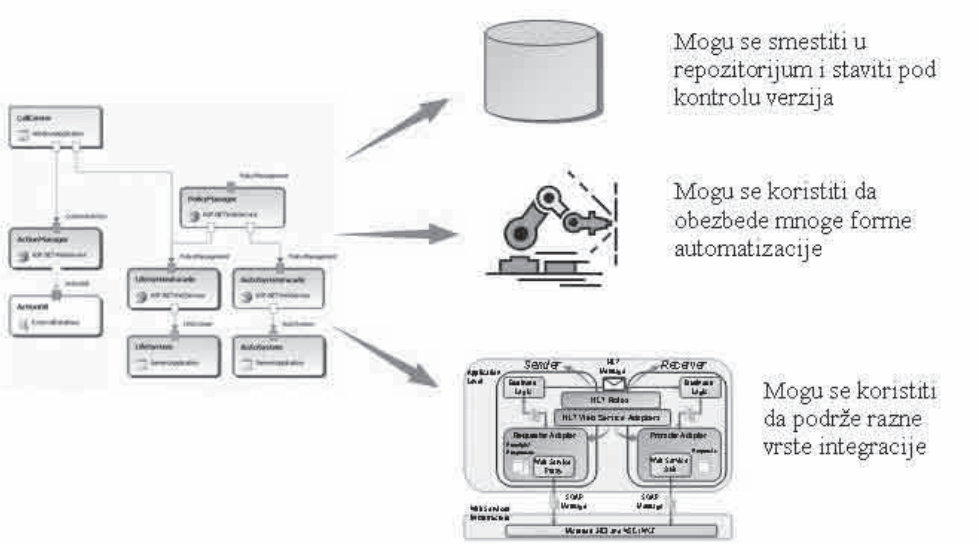
Zašto kompanije koriste metodologije? Metodologije, pre svega, obezbeđuju jedan konzistentan i reproduktivni prilaz koji može biti primenjen na sve projekte. Metodologije smanjuju rizik od grešaka. Metodologije izdaju kompletnu i konzistentnu dokumentaciju za trenutne i buduće projekte.

Usled činjenice da se projektni timovi i osoblja menjaju, metodologija omogućava da oni timovi koji nastavljaju rad svojih prethodnika, brzo i lako shvate rezultate rada svojih prethodnika.

## 4.1. Modeli i modeliranje

Modeli su apstrakcije koje odslikavaju ključne elemente kompleksnog problema ili strukture time što eliminišu manje bitne detalje i na taj način, čine problem daleko lakšim za razumevanje. Modeli se koriste da bi se opisali poslovni procesi, razumelo trenutno stanje poslovanja i modelovali novi procesi koji ne postoje, ali koji planiraju da se uvedu. Modeli su korisni za razumevanje problema, za komunikaciju među ljudima koji su uključeni u projekat, za modelovanje preduzeća, pripremu dokumentacije i za dizajn programa i baza podataka. Modelovanje omogućava bolje razumevanje zahteva, čistiji dizajn i sisteme pogodnije za održavanje.

Modeli mogu biti neformalni i formalni. Neformalni modeli se mogu koristiti samo za dokumentaciju, dok se formalni mogu izvršavati, kompajlirati ili prevesti u rezultate koji se mogu kompajlirati. Formalni modeli sadrže metapodatke i on je izražen jezikom domena problema (Slika 4.1).




Slika 4.1. Formalni modeli

Formalni model olakšava mapiranje koncepata problema i implementacije. Prednosti formalnog modela su: unapređivanje komunikacije među članovima tima, smanjivanje ručnog rada, podizanje nivoa apstrakcije na kojoj se radi projektovanje, unapređivanje praćenja odnosa zahteva i konačnog rešenja, obezbeđivanje automatizacije mehaničkih poslova itd.

Treba napraviti razliku između konceptualnih, logičkih i fizičkih modela. Konceptualni modeli pokazuju veoma grubo šta je sistem, odnosno prikazuje objekte, veze i referencijalna ograničenja. Logički modeli pokazuju tabele, redove, ključeve, ograničenja primarnih i spoljnih ključeva, normalizaciju. Oni opisuju sistem nezavisno od bilo koje tehničke implementacije. Fizički modeli pokazuju ne samo šta je sistem i šta on radi, već i kako je sistem fizički i tehnički implementiran. Prikazuju se tipovi podataka, indeksi, particije podataka itd. Sinonimi za fizički model su implementacioni model, tehnički model i sl. Da bi razjasnili navedenu razliku, razmotrimo sledeći primer. Pretpostavimo da korisnik želi da skladišti podatke o prosečnoj oceni studenata. Za svakog studenta, prosečna ocena mora biti između 5 i 10. Za svakog novog studenta (brucoša) se ne može formirati prosek, dok ne položi prvi ispit. Svi ovi iskazi su poslovne činjenice. Nas ne interesuje koju tehnologiju treba da koristimo da bi implementirali ove činjenice, činjenice su konstantne - one su logičke. Prosečna ocena je logički atribut sa logičkim domenom od 5 do 10, dok je logička difoltna vrednost prazno polje. Kod fizičkog modela prvo treba da odlučimo kako ćemo skladištiti ove podatke (atribute). Recimo da ćemo koristiti tabele baze podataka *Microsoft Access*-a. Prvo treba da se odredi naziv kolone, npr. ProsečnaOcenaStudenta. Zatim treba da se definišu fizička svojstva za kolonu ProsečnaOcenaStudenta, npr. tip podatka je Integer, da se odrede primarni ključevi, indeksi itd.

Za opis rada poslovnog sistema veliki problem predstavlja to što ne mogu da se koriste prirodni jezici, zbog jezičkih dvosmislenosti. S druge strane, precizan opis preko formalnih jezika je nerazumljiv za većinu ljudi. Stoga je potrebna tehnika koja će organizovati prirodne jezike na taj način da se eliminiše dvosmislenost i omogući efikasna komunikacija i razumevanje. Pokazalo se da je postupak modeliranja jedna od najefektivnijih tehnika za razumevanje i jednoznačnu komunikaciju između projektanata i korisnika.

**Modeliranje** je projektovanje softverskih aplikacija pre njihovog kodiranja odnosno programiranja. Poslovni problemi su veoma kompleksni što može da ometa komunikaciju između razvojnog tima i korisnika. Modeliranje može da pojasni složene probleme i samim tim olakša razumevanje. Odgovorni za razvoj softverskog projekta, korišćenjem modela su sigurni da je funkcionalnost kompletna i korektna. Istraživanja pokazuju da veliki softverski projekti koji nisu dobro izmodelirani imaju veću verovatnoću na neuspeh.

U procesu modeliranja, eliminišu se detalji, čime se umanjuje vidljiva kompleksnost sistema koji se proučava. Grafičke prezentacije (uglavnom pravougaonici i linije) koriste se da bi obezbedile da većina ljudi razmišlja o procesu modeliranja kao o slikovitoj prezentaciji. Pored grafičkog prikaza potrebno je dati i precizne definicije koji se pojavljuju u modelu, kao i propratni tekst koji ima ulogu kao sredstvo komunikacije.

Ovakav pristup nametnuo je potrebu za apstrakcijom, kojom se izvodi kontrolisano isključivanje detalja, tj. izvlače se zajedničke karakteristike u opisivanju nekog sistema. Tako je na višim nivoima apstrakcije sistem opisan jasnije, a na nižim detaljnije. Trenutno i buduće stanje sistema se modeluje kako bi se identifikovali:

- **zahtevi** – kada se modelira sistem, sve daljne diskusije i analize se fokusiraju na detalje modela i postizanje koncenzusa;
- **problemi i rizici** – ponekad postoji neslaganje oko toga šta su problemi, a šta rizici kod trenutnog sistema;
- **informacije koje nedostaju** – kreiranje modela sistema pomaže razvojnom timu da lako identifikuje ono što nije poznato.

Modeliranje je opšti pristup u svim inženjerskim disciplinama. U svakoj oblasti postoji više intelektualnih alata (jezika) za modeliranje sistema. U oblasti razvoja softvera, modeliranje se može izvoditi klasičnim pristupom – definisanjem modela podataka i modela procesa ili objektno-orijentisanim pristupom korišćenjem UML-a.

Za postupak modeliranja razvijeni su odgovarajući CASE alati, kao što su BPwin za funkcionalno modeliranje (podržava IDEF0, DFD), zatim ERwin za modeliranje podataka koji podržava IDEF1X standard, RatonalRose koji podržava UML i mnogi drugi. Ove tehnike omogućavaju:

- izvršenje sistemske analize i projektovanja informacionih sistema na svim nivoima;
- stvaranje dokumentacije kao osnova za integraciju informacionog sistema i ISO 9000 standarda;
- bolju komunikaciju između menadžera, korisnika, projektanta i programera;
- upravljanje velikim i složenim projektima.

## 4.2. Strategije sistemske analize

Sistemska analiza je najkritičnija faza jednog projekta. Tokom sistemske analize treba da se shvate problemi konkretnog poslovnog sistema, definišu ciljevi za njegovo poboljšanje i definišu detaljni poslovni zahtevi, koje mora da ispuni bilo koje tehničko rešenje. Odnosno, svako naredno projektovanje i implementacija novog sistema zavisi će od kvaliteta prethodno obavljene sistemske analize.

**Sistemska analiza je metodološki postupak dekompozicije nekog sistema na podsisteme (komponente) sa ciljem da se prouči njihov međusobni uticaj i rad.** Kada se radi sistemska analiza izvršava se i sistemska sinteza. Sistemska sinteza je integracija komponenata sistema u jedan celokupan sistem, za koji se pretpostavlja da je poboljšani sistem. Tokom sistemske analize i sinteze dodaju se, brišu i modifikuju komponente sistema prema cilju poboljšanja globalnog sistema.

Sistemska analiza se bavi rešavanjem problema. S obzirom da postoji mnogo različitih pristupa za rešavanje problema, tako postoje i mnoge strategije i tehnike sistemske analize, neke od njih su strukturna analiza, informacioni inženjering, pravljenje prototipa, objektno-orijentisana analiza i dr. Pogledajmo ukratko, šta neke od njih podrazumevaju.

**Strukturna analiza (*Structured Analysis* - SA)** je jedna od najstarijih tehnika sistemske analize koja se i danas dosta koristi. Ona je orjentisana na procese i koristi se za modeliranje poslovnih zahteva sistema. Modeli su struktuirane slike koje prikazuju procese, ulaze, izlaze i skladišta podataka. Sistem analitičari crtaju seriju modela procesa, na primer dijagrame toka podataka, da bi opisali osnovne procese sistema, tokove i skladišta podataka. Danas se za projektovanje softvera uglavnom koriste objektno-orijentisane metode. Strukturna analiza se još uvek koristi kod reinženjeringa poslovnih procesa (*Business Process Redesign*). Reinženjeringom poslovnih procesa, organizacija proučava osnovne poslovne procese kako bi eliminisala ili smanjila nepotrebne tokove podataka i procese u postojećem sistemu u cilju povećanja efikasnosti i smanjenja troškova.

**Informacioni inženjering i modeliranje podataka** je tehnika koja je usredsređena na podatke u sistemu i strateško planiranje. Modeli podataka kod informacionog inženjeringa (*Information Engineering* – IE) se nazivaju dijagrami objekti-veze (*Entity Relationship Diagrams*).

Sistem analitičari crtaju dijagrame objekti-veze kako bi modelirali podatke sistema i prikazali kako će se podaci prikupljati, skladištiti, koristiti i održavati. Strukturna analiza i informacioni inženjering pokušavaju da sinhronizuju modele podataka i modele procesa.

**Objektno-orijentisana analiza (*Object-Oriented Analysis* - OOA)** je tehnika koja je uspela da eliminiše veštački razdvojene podatake od procesa. Umesto podataka i procesa koji su kreirali, čitali, ažurirali i brisali te podatke, oni su zajedno integrisani u objekte. Jedini način da se kreiraju, čitaju, ažuriraju ili brišu podaci objekta (*properties*) jeste putem ugrađenih procesa koji se nazivaju metode (*methods*). Programski jezici kao što su C++, Visual Basic, C#, Java i mnogi drugi se baziraju na ovoj objektnoj tehnologiji. OOA je naročito postao popularan pojavom UML-a (*Unified Modeling Language*) koji obezbeđuje i grafičku sintaksu za objektne modele. UML definiše nekoliko različitih tipova dijagrama koji zajedno modeluju jedan informacioni sistem ili aplikaciju u terminima objekata.

## 4.3. Reinženjering i modeliranje procesa

ERP sistemi doprinose reinženjeringu poslovnih procesa kombinovanih sa upotrebom novih informacionih tehnologija. Reinženjering se može definisati kao temeljno razmatranje i radikalno reprojektovanje poslovnih procesa kako bi se postigla drastična poboljšanja u kritičnim merama performansi, kao što su troškovi, kvalitet, usluge i brzina. Da bi se razumeo reinženjering, veoma je važno da se shvati koncept lanca vrednosti. U tabeli 4.1, su prikazane primarne aktivnosti preduzeća koje uključuju ulaznu logistiku, operacije, izlaznu logistiku, marketing i usluge. Ove aktivnosti su osnovne za kreiranje, proizvodnju, marketing, prodaju i podršku proizvoda ili usluga. Jedan informacioni sistem podržava svaku od ovih primarnih aktivnosti. Taj informacioni sistem može da smanji troškove izvršavanja aktivnosti ili se može koristiti da obezbedi dodatnu vrednost karakteristikama proizvoda ili usluga. Na primer, aktivnost nazvana izlazna logistika se bavi obradom porudžbina klijenata. Sistem za *on-line* unos porudžbina koji omogućava klijentima elektronsko poručivanje, može da smanji troškove i vremena ove aktivnosti. Podrška ovih usluga može da se vrši putem udaljenih mašina što dodaje vrednost poslovanju obezbeđujući, pri tom, *on-line* dijagnostičku podršku.

Tabela 4.1. Lanac vrednosti: primarne aktivnosti

| **Primarne aktivnosti** | **Ulazna logistika** | **Operacije** | **Izlazna logistika** | **Marketing i prodaja** | **Usluge** |
|---|---|---|---|---|---|
| Funkcije | Rukovanje materijalima; dostava | Proizvodnja; montaža delova | Obrada porudžbina; isporuka | Reklamiranje; promocije | Usluge<br>Servisiranje |
| Informacioni sistemi podržavaju primarne aktivnosti | Automatizovani sistemi skladištenja | Sistemi za kontrolu proizvodnje | *Online* sistemi za unos porudžbina | Sistemi za analizu marketinga | Dijagnost. sa udaljenih mašina |

U tabeli 4.2 su prikazane sekundarne aktivnosti koje podržavaju primarne aktivnosti organizacije. Ove sekundarne aktivnosti uključuju organizacionu strukturu, ljudske resurse, tehnologiju i nabavku. Informacioni sistemi mogu da podrže svaku od ovih sekundarnih aktivnosti. Na primer, elektronska pošta (*email*) može da podrži organizacionu strukturu olakšavajući vremenske komunikacije. Informacioni sistemi mogu da podrže nabavku omogućavajući nabavljačima da se povežu sa bazama podataka dobavljača kako bi postavili porudžbine, odredili nivo zaliha i izvršili proveru cena.

Tabela 4.2. Lanac vrednosti: sekundarne aktivnosti

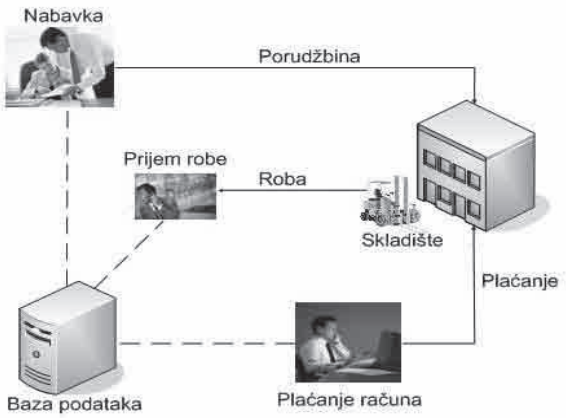
| **Aktivnosti podrške** | **Informacioni sistemi** |
|---|---|
| Organizacija | *Email* (olakšava komunikaciju kroz organizaciju) |
| Ljudski resursi | Veštine upravljanja bazama podataka |
| Tehnologija | Dizajniranje i proizvodnja pomoću računara (CAD i CAM) |
| Nabavka | *Online* veze sa bazama podataka dobavljača |

Menadžment lanca snabdevanja, uključuje planiranje i kontrolu svih zadataka poslovnog lanca vrednosti. Na ovaj način, dobavljači se mogu direktno povezati sa proizvođačima, a proizvođači sa prodavcima na malo, a oni se mogu direktno povezati sa klijentima. Ove elektronske povezanosti smanjuju troškove, poboljšavaju vremensku obradu porudžbina i isporuku, povećavajući nivoe usluga.

U današnjoj ekonomiji, među glavnim motivacijama za modernizovanjem i reinženjeringom poslovnih procesa su satisfakcija klijenata,

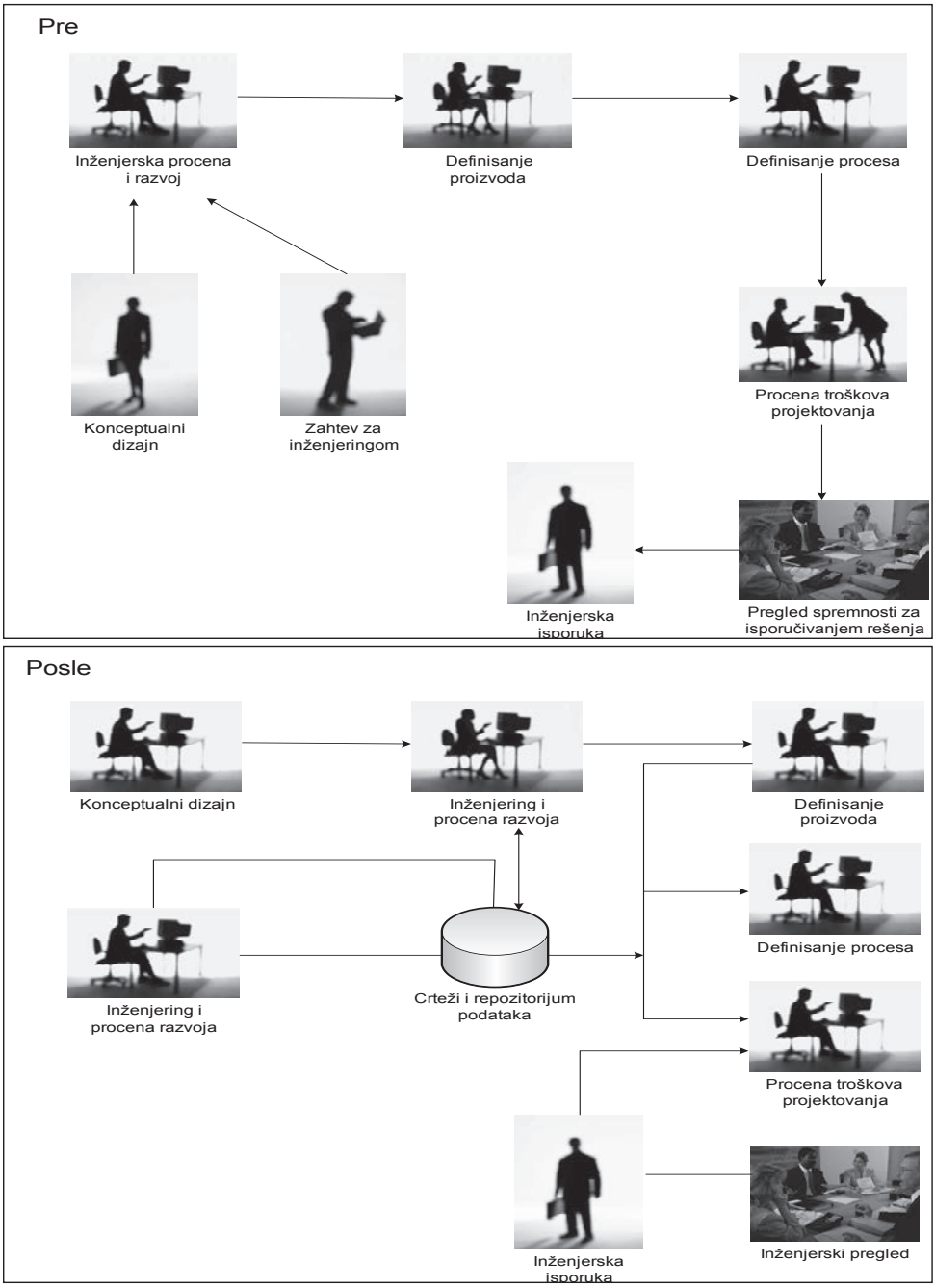
deregulacija i povećanje konkurentnosti na globalnom nivou. Generalno, osetljivost na potrebe tržišta i fokus na potrebe klijenata zahteva brzo vreme odziva i uprošćavanje i integraciju procesa.

Primeri reinženjeringa procesa se prikazuju na studijama slučajeva koji objašnjavaju situaciju pre i posle reinženjeringa. Prva studija slučaja se bavi sistemom za plaćanje računa (*Accounts Payable System*) kompanije *Ford Motors*. U situaciji pre reinženjeringa održavane su nezavisne baze podataka za nabavku, prihvatanje i plaćanje računa. Kada se roba nabavlja tada se podešava zapis u bazi podataka Nabavke. Kada se prihvati roba od dobavljača, odelenje za prijem ažurira sopstvenu bazu podataka sa primljenim iznosom. Ukoliko bi isporuka bila parcijalna, onda se zapis iz baze podataka Nabavke ne bi poklapao sa zapisom baze podataka Prihvatanja. Ovakva nekonzistencija stvara probleme za plaćanje računa koji opet ima sopstvenu bazu podataka. Nakon reinženjeringa (Slika 4.2), integrisana baza podataka podržava Nabavku, Prihvatanje i Plaćanje računa. Kompanija je postavila novo poslovno pravilo da se plaćanje računa izvrši tek po isporuci robe od dobavljača. Na ovaj način su mnoge nekonzistentnosti eliminisane. Sa uvođenjem procedura i zajedničke baze podataka, znatno je redukovano administrativno osoblje. U ovom slučaju, izmena procedura, uvođenje novih tehnologija i među funkcionalna saradnja su znatno smanjile administrativne troškove.




Slika 4.2. Primer reinženjeringa: Sistem za plaćanje računa

Drugi primer je razvoj proizvoda kompanije Xerox (Slika 4.3), gde je integrisanje podataka dovelo do smanjenja vremena razvoja proizvoda i poboljšanja odzivnosti na potrebe tržišta.




Slika 4.3. Primer reinženjeringa: Xerox

Prethodne slikovite studije poslužile su radi lakšeg razumevanja procesa reinženjeringa. Međutim, kada se radi ozbiljan reinženjering poslovnih procesa uglavnom se koriste tehnika strukturne sistem analize - modeliranje procesa.

Modeliranje procesa je tehnika koja organizuje i dokumentuje procese sistema i/ili implementira logiku, politike i procedure sistema. Neki od modela procesa sistemske analize su dijagram toka podataka (*Data Flow Diagram* - DFD), IDEF0 metodologija i dr. Najvažnija korist u primeni modeliranja aktivnosti je prototipski pristup gde se na brz i jednostavan način proveravaju alternativne ideje. Ovo je veoma bitna osobina jer brzi razvoj informacionih tehnologija i primena Internet servisa uslovljava potrebu za reinženjeringom koja zahteva radikalni redizajn aktivnosti, a koje je potrebno opisati i pre sprovođenja prototipski proveriti.

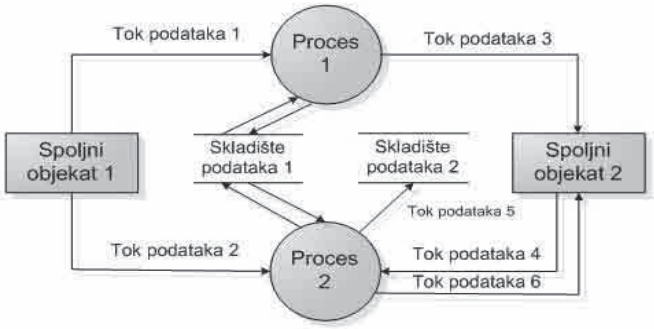
### 4.3.1. Dijagram toka podataka

**Dijagram toka podataka (DTP)** je alat koji opisuje tokove podataka kroz sistem i procese koji se izvršavaju u sistemu. Dijagram toka podataka pre-dstavlja model sistema koji sadrži četiri osnovne komponente (Slika 4.4):

- **procese** *(processes)* obrade podataka, koji predstavljaju aktivne komponente sistema (grafički simbol: krug);
- **spoljne objekte ili spoljne agente** *(external agents)* sa kojima sistem komunicira (grafički simbol: pravougaonik);
- **skladišta podataka** *(data stores)* koje procesi koriste i/ili ažuriraju (grafički simbol: dve paralelne linije) i
- **tokove podataka** *(data flows)* koji povezuju ostale komponente sistema u celinu (grafički simbol: usmerena linija).

#### Sintaksa (pravila kreiranja) dijagrama toka podataka

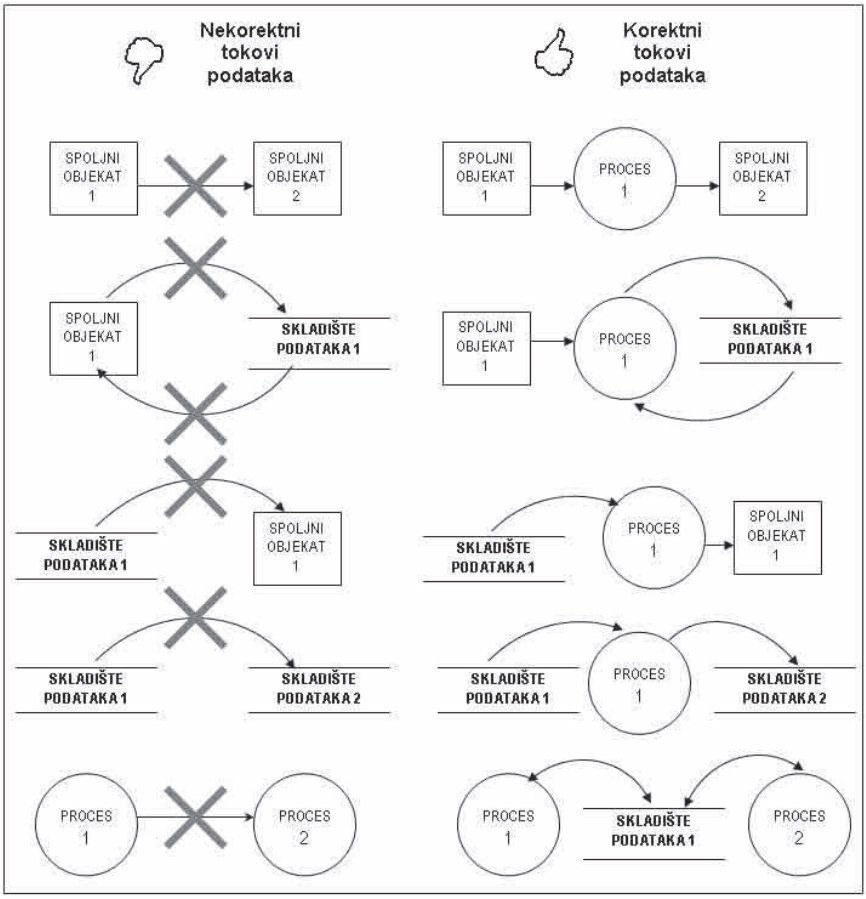
Kod kreiranja dijagrama toka podataka važe sledeća osnovna pravila:




Slika 4.4. Osnovne komponente DTP-a[1]

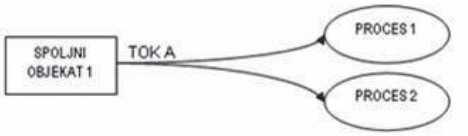
1 Korišćena je DeMarco/Yourdon notacija jer poseduje veoma jednostavne, grafičke, pa samim tim i jasne koncepte.

1. Tok podataka predstavlja podatak u pokretu. Tok podataka predstavlja ulaz podataka u proces ili izlaz podataka ili informacija iz procesa (to mogu biti različita dokumenta, formulari, tekstovi, knjige i slično). Oni se takođe koriste za prikazivanje kreiranja, brisanja ili menjanja podataka u bazi podataka ili datoteci. Tok podataka ostvaruje vezu između ostalih komponenti sistema i na DTP-u se predstavlja imenovanom, orjentisanom linijom.
2. Svaki tok podataka mora da ima svoj izvor i ponor. Međutim, za jedan tok, bilo izvor, bilo ponor (ili oba) mora da bude proces. Na slici 4.5 se prikazuju nepropisne situacije tokova podataka za koje se mogu dati sledeći komentari:
   - Direktno povezivanje spoljnih objekata, objekata van sistema, nekim tokom podataka nije dozvoljena, jer predstavlja opis odnosa objekata van posmatranog sistema, pa nije od interesa. Ako bi takva veza bila od interesa, odnosno ostvarivala se preko posmatranog sistema, tada bi se morao definisati proces u posmatranom sistemu koji takvu vezu uspostavlja.
   - U realnom sistemu spoljni objekti nemaju direktan pristup skladištima podataka, već to obavljaju procesi sistema.
   - Ukoliko je neophodno da se podaci sa jednog skladišta podataka prenesu na drugo skladište podataka, ta veza se ne može ostvariti direktnom vezom između skladišta podataka, već se mora formirati proces preko koga će se vršiti to premeštanje.
   - Procesi se ne mogu direktno povezivati, već ukoliko na osnovu rezultata koje daje jedan proces, a čiji sadržaj se smešta u skladište podataka, počinje da radi drugi proces, onda on uzima te rezultate iz skladišta u koji su smešteni i na taj način se veza između procesa ostvaruje putem skladišta podataka.




Slika 4.5. Pravilan i nepravilan način prikazivanja DTP komponenata

3. Svaki tok podataka na DTP-u mora imati ime, koje treba da odražava značenje podataka koje on nosi. Ova imena treba da budu prirodna, a ne neka specifična, kodirana, preterano skraćena imena, kako bi se ostvarila čitljivost i razumljivost DTP-a. Izuzetak su tokovi koji idu ka, odnosno od skladišta podataka koji ne moraju biti imenovani. Ukoliko tok između procesa i skladišta nije imenovan, podrazumeva se da tok nosi celokupan sadržaj i strukturu podataka tog skladišta.
4. Tok podataka se može granati. Istoimeni tokovi na DTP-u u suštini predstavljaju grananje jednog toka, pa moraju imati zajednički izvor, a mogu imati različite ponore.




5. Proces obrade podataka je aktivna komponenta sistema koja vrši transformaciju strukture i sadržaja ulaznog u izlazni ili izlaznih tokova u ulazne tokove podataka. Svaki proces ima naziv i oznaku. Naziv procesa treba precizno da označava funkciju koju on obavlja.
6. Svaki proces mora da ima barem jedan ulazni i barem jedan izlazni tok podataka. Proces koji ima ulaze, a nema izlaze je tzv. crna rupa, jer izgleda kao da je podatak jednostavno nestao. Proces koji ima izlaze, a nema ulaze, je takođe nemoguć.
7. Svako skladište podataka bi trebalo da ima barem jedan ulazni i barem jedan izlazni tok. Međutim, dozvoljava se da skladište nema ulazni tok, podrazumevajući da se formira i ažurira u nekom drugom sistemu, odnosno da nema izlazni tok, podrazumevajući da posmatrani sistem formira i ažurira skladište koje se koristi u nekom drugom sistemu.
8. Svaki spoljni objekat (spoljni agent) mora da ima barem jedan, bilo ulazni, bilo izlazni tok podataka, inače bi bio izolovan od ostalog dela sistema.
9. Da bi se izbeglo nepotrebno presecanje linija, dozvoljava se da se skladište ili spoljni objekat, na jednoj slici, višestruko ponovi.

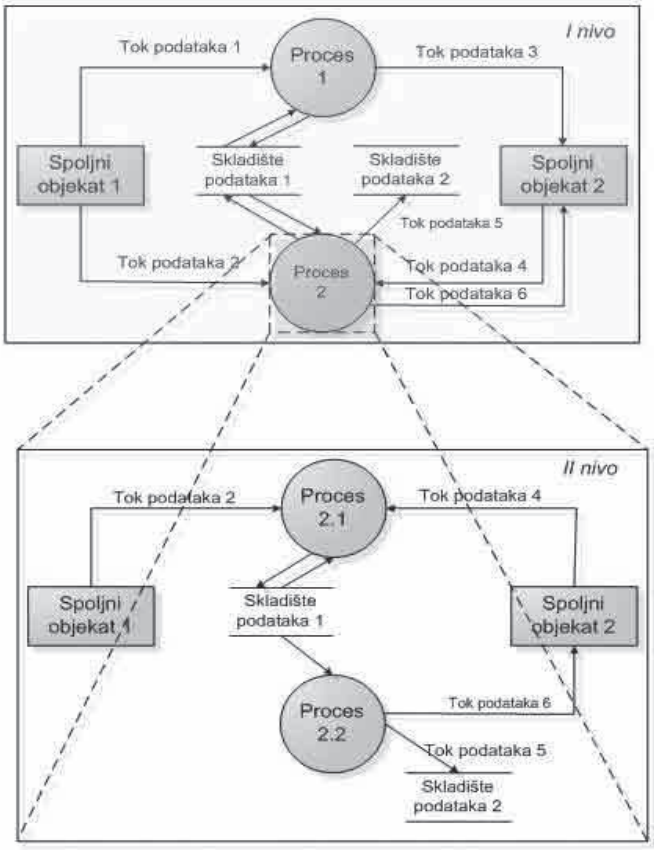
### Hijerarhijska dekompozicija dijagrama toka podataka

Kompleksni sistemi, kada se posmatraju u globalu kao jedan proces, njih je veoma teško razumeti. Stoga, se u sistemskoj analizi, sistem razlaže ili dekomponuje na svoje komponente odnosno na podsisteme. To se postiže primenom hijerarhijske dekompozicije DTP-a.

Dekompozicija je način razlaganja sistema na njegove komponente podsisteme, procese i podprocese. Svaki nivo apstrakcije opisuje, sa više ili manje detalja, celokupan sistem ili podskupove tog sistema.

**Dijagram dekompozicije** prikazuje *top-down* (sa vrha na dole) funkcionalnu dekomoziciju i strukturu sistema. Hijerarhijska dekompozicija DTP se

izvodi na taj način što se jedan proces višeg nivoa apstrakcije dekomponuje i prikazuje pomoću novog celokupnog DTP-a na nižem nivou apstrakcije. Tehnika hijerarhijske dekompozicije prikazana je na slici 4.6, gde je proces 2, dekomponovan u novi dijagram toka podataka na nižem nivou.

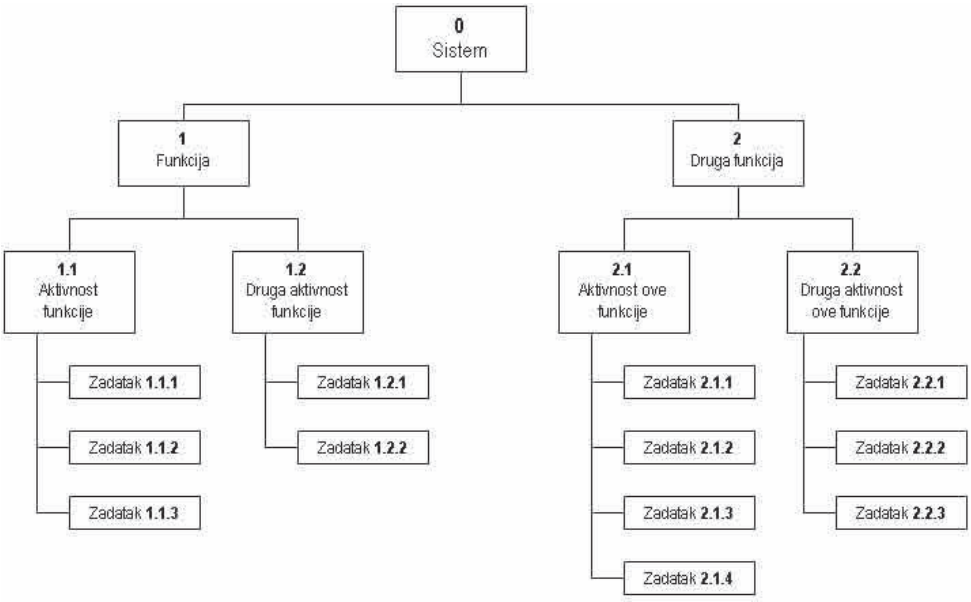



Slika 4.6. Hijerarhijska dekompozicija DTP-a

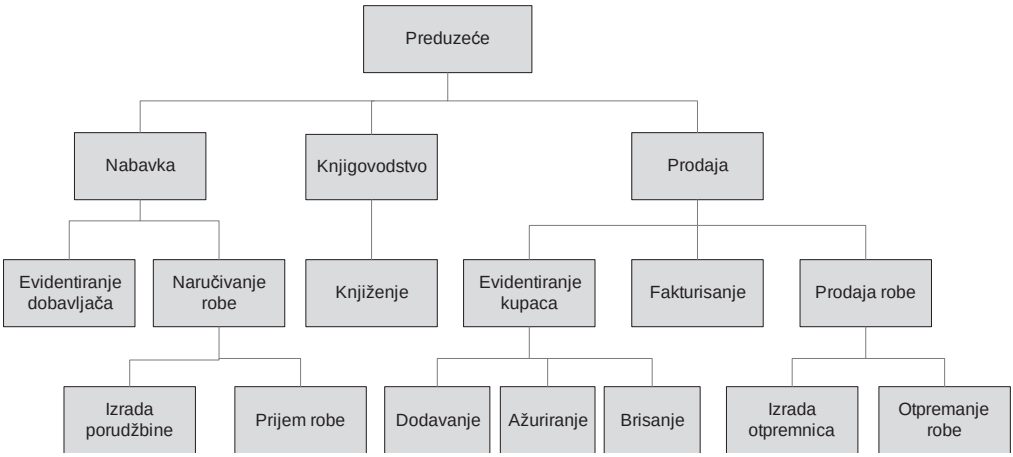
Pri dekompoziciji DTP-a moraju se poštovati sledeća pravila:

1. Dijagram najvišeg nivoa, koji po pravilu sadrži samo jedan proces koji predstavlja ceo IS, zatim spoljne objekte sa kojima IS komunicira i odgovarajuće tokove podataka naziva se **dijagram konteksta**.
2. Dijagram prvog nivoa predstavlja dekompoziciju dijagrama konteksta. Procesi se označavaju brojevima 1,2,3.
3. Svaki proces sa dijagrama prvog nivoa se dalje dekomponuje do nivoa primitivnih procesa (procesa koji se dalje ne dekomponuju). Procesi na dijagramima nižih nivoa, povlače sa sobom brojnu oznaku nadređenog procesa (npr. 2.1.).

4. Uobičajeno je da se zatim celokupan sistem predstavi dijagramom stabla aktivnosti (Slika 4.7).
5. Procesi na najnižem nivou odnosno procesi koji se dalje ne dekomponuju, se nazivaju primitivni procesi i za njih se daje specifikacija logike njihovog odvijanja.
6. Pored procesa, mogu se dekomponovati i tokovi i skladišta. Njihov opis se detaljno daje u rečniku podataka. Na primer, tok Ponuda se dekomponuje na sledećem dijagramu nižeg nivoa, u tokove Podaci o klijentu i Podaci o nekretnini.
7. Najvažnije pravilo koje se mora poštovati pri dekompoziciji procesa je pravilo balansa tokova, što podrazumeva da ulazni i izlazni tokovi nekog procesa na dijagramu nižeg nivoa moraju biti ekvivalentni sa ulaznim i izlaznim tokovima tog istog procesa na dijagramu višeg nivoa.
8. Skladišta podataka ukoliko se pojave na jednom nivou, uz jedan proces, moraju se nadalje pojavljivati na svim nižim nivoima dekompozicije toga procesa.



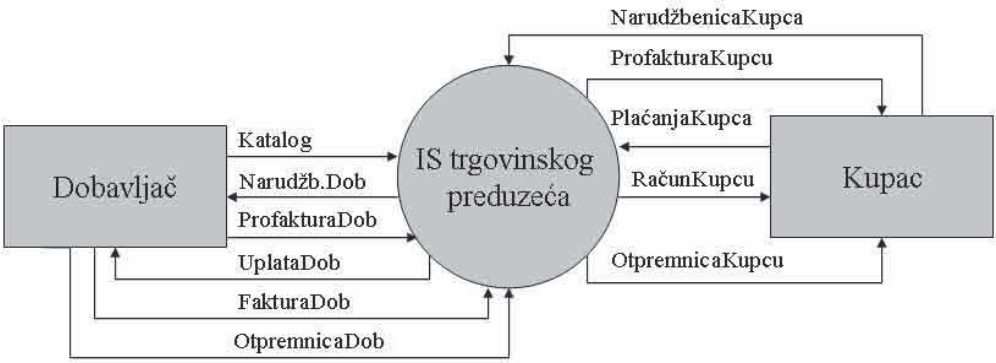

Slika 4.7. Opšti primer dijagrama stabla aktivnosti




Slika 4.8. Primer dijagrama funkcija za jedan sistem preduzeća

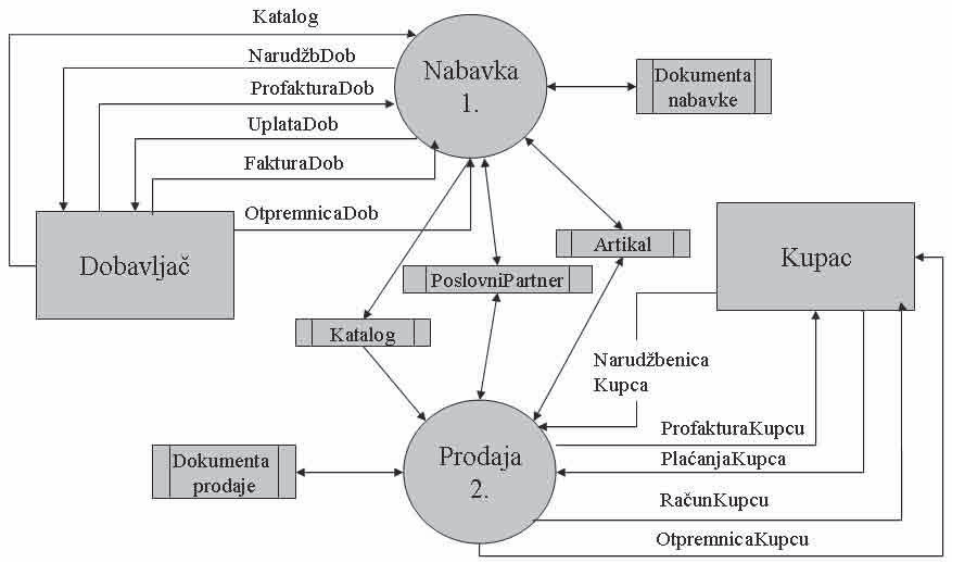
**Primer dekompozicije sistema**

Na slikama, je predstavljen primer dekompozicije IS jednog manjeg trgovinskog preduzeća. Na dijagramu konteksta uočavaju se dva spoljna objekta i to Dobavljač i Kupac i svi tokovi podataka preko kojih ovi objekti komuniciraju sa IS preduzeća.



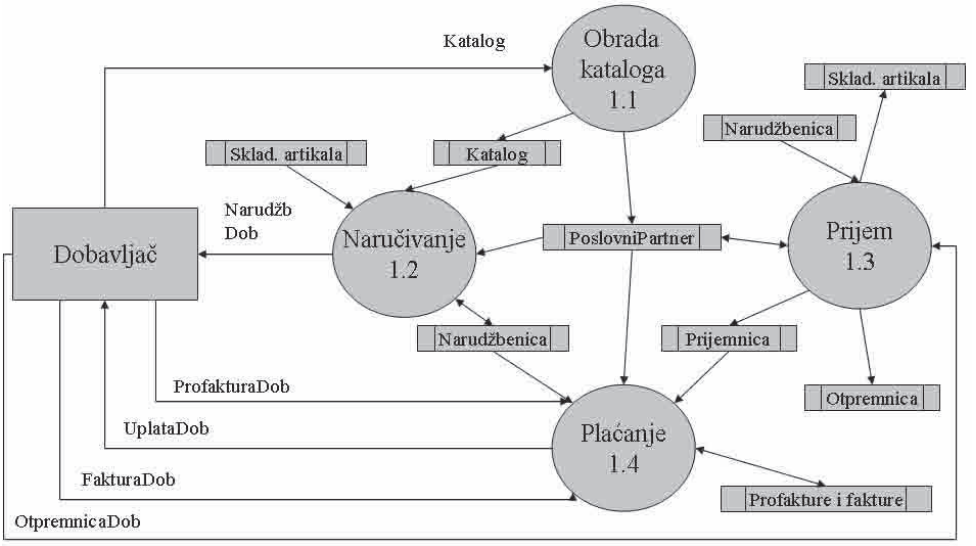

Slika 4.9. Dijagram konteksta

Na dijagramu prvog nivoa dekompozicije razlažemo IS trgovinskog preduzeća na globalne procese, u ovom slučaju ih ima dva i to: Nabavka i Prodaja. Odgovarajuće tokove podataka pridružujemo odgovarajućem procesu. Na ovom dijagramu iscrtavaju se i sva skladišta podataka koja se pojavljuju u sistemu, u našem primeru ima ih četiri: Katalog, Poslovni partner, Artikal i Dokumenta nabavke.

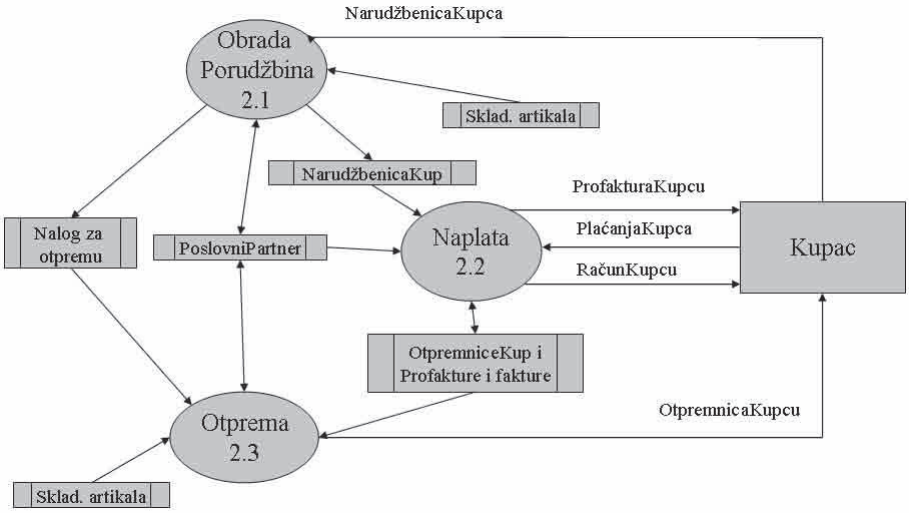



Slika 4.10. Dijagram prvog nivoa

Sledeći korak je dekompozicija globalnih procesa do nivoa primitivnih procesa (zadataka). Prvi proces Nabavka se dekomponuje na četiri nova procesa kao što se može videti sa slike 4.11. Odgovarajući tokovi i skladišta se priključuju odgovarajućem pod procesu. Drugi proces Prodaja se dekomponuje na tri primitivna procesa kao što se može videti sa slike 4.12.




Slika 4.11. Dekompozicioni dijagram procesa Nabavke




Slika 4.12. DTP Prodaje

### 4.3.2. Rečnik podataka

Rečnik podataka (*Data Dictionary*) predstavlja struktuirano skladište meta podataka, tj. podataka o podacima. Prvobitno se pojavio kao proširenje dijagrama toka podataka, za opisivanje sadržaja i strukture svih tokova i skladišta podataka. Danas, za opis rečnika podataka upotrebljava se BNF notacija (Backus-Naur Form), koja se inače koristi za opis sintakse programskih jezika. Definisanje podataka BNF notacijom se prikazuje na tabeli 4.3.

Tabela 4.3. BNF notacija

| Znak | Opis |
|---|---|
| = | Struktura sa leve strane se sastoji od delova sa desne („sastavljeno od“) |
| + | Agregacija elemenata |
| ( ) | Opcionalnost elemenata u zagradi (0 ili 1) |
| { } | Ponavljanje (iteracija) elemenata u zagradi do konačnog broja puta |
| [ ] | Alternativa (selekcija) elemenata u zagradi |
| \| | Odvajanje alternativnih elemenata u [ ] izrazu |
| - | Početna i završna vrednost raspona definisanog [ ] izrazom |
| ** | Komentar |
| @ | Oznaka ključa |

Na primer, račun i stavke računa se notacijom BNF prikazuju:

Racun = @BrRac + DatRac + BrKupca
+ { SifArt, NazArt, Cena, Kol, Vrednost }
+ (IznosRac)

Takođe, prethodni primer se može napisati i na sledeći način:

Racun = @BrRac + DatRac + BrKupca
+ { StavkaRac}
+ (IznosRac)
StavkaRac = @SifArt, NazArt, Cena, Kol, Vrednost

Sledeći primer, pokazuje alternativne elemente.

Proizvod = @ProizvodID + Naziv
+ [KolicinaPorudzbine, StopaAmortizacije]

Ukoliko je proizvod osnovno sredstvo, onda će se pojaviti polje StopaAmortizacije, a ukoliko je proizvod materijal za proizvodnju, onda će se pojaviti polje KolicinaPorudzbine.

### Elementarni procesi

Mini-specifikacije služe za opisivanje osnovnih procesa u dijagramu toka podataka. One mogu imati različite oblike, ali imaju nekoliko zajedničkih elemenata:

- naziv i broj procesa
- listu podataka koji ulaze u sistem (ulazni tokovi podataka)
- listu podataka koji izlaze iz procesa (izlazni tokovi podataka)
- telo opisa procesa

Opis procesa sadrži algoritam zadataka koje proces obavlja i može poprimiti različite oblike. Na bazi ovog algoritma može se u zavisnosti od alata, generisati kôd. Primer definicije procesa se prikazuje u tabeli 4.4. Elementarni procesi se opisuju proceduralnom logikom za čije opisivanje se koristi strukturni engleski jezik (Tabela 4.5).

Tabela 4.4. Primer definicije elementarnih procesa

| Proces 1 | Provera raspoloživosti filma |
|---|---|
| Opis: | Provera da li u video klubu postoji kopija filma koja se može iznajmiti |
| Ulazni tokovi: | Upit (Film), Rezervacije, Iznajmljivanje |
| Izlazni tokovi: | Film nije raspoloživ, Film je raspoloživ |
| Logika procesa: | **Izračunaj** broj kopija traženog filma u video klubu<br>**Ako** je broj kopija veći od nule **tada**<br>**Ako** je broj kopija veći od (broj Iznajmljivanja + broj rezervacija)<br>Rezultat = „Raspoloživ“<br>**Else**<br>Rezultat = „Nije raspoloživ“<br>**Inače**<br>Rezultat = „Ne postoji“ |

Tabela 4.5. Proceduralne strukture struktuiranog jezika

| Konstrukcija | Primer *Template* |
|---|---|
| **Sekvenca akcija** – bezuslovno se izvršava niz akcija | [Akcija 1]<br>[Akcija 2]<br>...<br>[Akcija n] |
| **Jednostavan uslov akcija** – Ako (*if*) je specifiran uslov istinit (*true*), onda (*then*) se izvršava prvi skup akcija. Drugačije (*else*) se izvršava drugi skup akcija.<br>Koristite ovu konstrukciju ukoliko imate samo dve moguće vrednosti.<br>(Napomena: Drugi skup uslova je opcioni) | **If** [istinit uslov]<br>**Then**<br>[niz akcija]<br>**else**<br>[niz akcija]<br>~~End if~~ |
| **Kompleksni uslovi akcija** – testira vrednost uslova i izvršava odgovarajući skup akcija.<br><br>Koristite ovu konstrukciju ukoliko uslov ima više od dve vrednosti. | **Uradite sledeće zasnovano na** [uslovu]:<br>**Case 1: if** [uslov] = [vrednost] then<br>[niz akcija]<br>**Case 2: if** [uslov] = [vrednost] then<br>[niz akcija]<br>...<br>**Case n: if** [uslov] = [vrednost] then<br>[niz akcija]<br>~~End Case~~ |
| **Višestruki uslovi** – testira vrednosti višestrukih uslova kako bi se odredio korektan skup akcija.<br><br>Koristite tabelu odlučivanja, umesto gnezda *if-then-else* konstrukcije strukturnog engleskog kako bi pojednostavili predstavljanje kompleksne logike.<br><br>**Tabela odlučivanja** *je tabelarni prikaz kompleksne logike u kojoj redovi predstavljaju uslove i moguće akcije, a kolone prikazuju koja kombinacija uslova rezultira u određenu akciju.* | Tabela odlučivanja: Pravilo / Pravilo / Pravilo<br>[Uslov] vrednost vrednost vrednost<br>[Uslov] vrednost vrednost vrednost<br>[Uslov] vrednost vrednost vrednost<br>[Niz akcija] X<br>[Niz akcija] X<br>[Niz akcija] X |
| **Jedna do više iteracija** – ponavlja (*repeat*) skup akcija, dok (*until*) je uslov netačan (*false*).<br>Koristite ovu konstrukciju ukoliku skup akcija mora biti izvršen najmanje jedanput, bez obzira na početnu vrednost uslova. | **Repeat** sledeće **until** [istinit uslov]:<br>[niz akcija]<br>~~End Repeat~~ |
| **Nijedna do više iteracija** – ponavlja (*Do While*) skup akcija dok je uslov istinit.<br><br>Koristite ovu instrukciju ukoliko je skup akcija uslovljen početnom vrednošću. | **Do While** [istinit uslov]<br>[niz akcija]<br>~~End Do~~<br>- ILI -<br>**For** [istinit uslov]<br>[niz akcija]<br>~~End For~~ |

### 4.3.3. IDEF0 – funkcionalno modeliranje

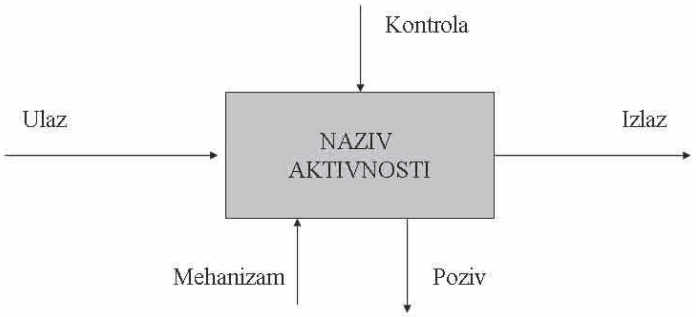
Ranih devedesetih godina, IDEF (*Integration DEFinition*) *Users Group* usvojila je standarde za funkcionalno modeliranje IDEF0 i za informaciono modeliranje IDEF1X. Softverska realizacija IDEF0 standarda je BPwin (*Business Process Windows*) firme LogicWorks. IDEF1X je semantički bogat modelar podataka koji je realizovan u okviru softvera ERwin (*Entity Relationships for windows*) CASE alata.

Funkcionalno modeliranje pomoću IDEF0 tehnike omogućava dekomponovanje poslovnih funkcija i planiranje potrebnih resursa za realizaciju funkcija. IDEF0 funkcionalni model se sastoji od hijerarhijskog niza dijagrama koji postepeno prikazuju sve više detalja o funkcijama i njihovoj međuvezi sa ostalim delovima sistema. IDEF0 metode se koriste za modeliranje funkcija, kreiranje grafičkog modela koji pokazuje ko upravlja funkcijama, ko ih izvršava, koji se resursi koriste, ko ih proizvodi i koje relacije postoje sa drugim funkcijama.

IDEF0 predstavlja sistem kao mrežu međusobno povezanih aktivnosti. Model je hijerarhijski organizovan jer započinje sa jednom aktivnošću na najvišem nivou, a zatim se ta aktivnost dalje dekomponuje do nivoa primitivnih procesa. IDEF0 se može koristiti za dobijanje struktuirane dokumentacije koju podržava ISO 9000.

IDEF0 model se sastoji od 2 elementa (Slika 4.13):

- Pravougaonik – predstavlja aktivnosti definisane kao funkcije, procese i transformacije. Svaki pravougaonik ima naziv i broj. Za naziv aktivnosti se koristi aktivan glagol ili glagolska fraza koja opisuje funkciju.
- Strelice – predstavljaju podatke ili objekte vezane za aktivnosti.



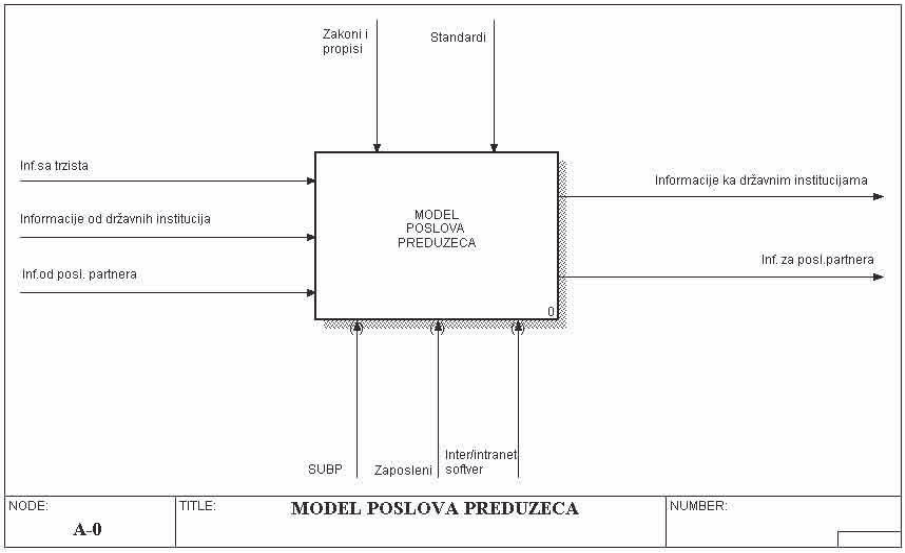

Slika 4.13. Opšti model dijagrama konteksta

Ulazi se transformišu u aktivnosti koristeći mehanizme ili resurse kao što su osoblje i mašine, da bi proizvele izlaze. Operacije aktivnosti će biti usmerene kontrolama kao što su politike i procedure. Poziv je specifičan slučaj mehanizma i označava da pozivajuća aktivnost nema vlastiti detaljniji dijagram, već daje detaljniji prikaz izveden na nekom drugom modelu. Sledi detaljniji opis tokova podataka:

- **Ulazna strelica** - predstavlja materijal ili informaciju koja se koristi ili transformiše sa ciljem definisanja izlaza. Dozvoljava se mogućnost da određene aktivnosti ne moraju imati ulazne strelice.
- **Kontrolne strelice** - strelice koje ulaze u pravougaonik odozgo definišu se kao kontrole (*Control*). Kontrolne strelice regulišu, odnosno odgovorne su za to kako, kada i da li će se aktivnost izvesti, odnosno kakvi će biti izlazi. Svaka aktivnost mora imati najmanje jednu kontrolnu strelicu. Kontrole su često u obliku pravila, politika, procedura ili standarda. One utiču na aktivnost bez mogućnosti da budu transformisane ili upotrebljene.
- **Izlazne strelice** - materijali ili informacije stvorene aktivnošću. Svaka aktivnost mora imati najmanje jednu izlaznu strelicu. Aktivnost koja ne stvara izlaz, ne treba ni modelirati.
- **Strelice mehanizama** - strelice na donjoj strani pravougaonika predstavljaju mehanizme. Strelice okrenute prema gore identifikuju značenje koje podržavaju izvršenje aktivnosti. Strelice mehanizama su oni izvori koji izvode aktivnosti, a sami se ne "troše". Mehanizmi mogu biti ljudi, mašine i/ili oprema tj. objekti koja obezbeđuju energiju potrebnu za izvođenje aktivnosti. Mehanizami mogu biti izostavljeni iz aktivnosti.
- **Strelica poziv** - strelice mehanizma koje su okrenute na dole definišu se kao strelice poziva (*Call arrows*). Strelica poziv je specifični slučaj strelice mehanizma i ona označava da pozivajući pravougaonik nema svoj vlastiti detaljniji dijagram, ali je detaljniji prikaz izveden na nekom drugom pravougaoniku u istom ili nekom drugom modelu. Više pozivajućih pravougaonika mogu pozivati isti pravougaonik na nekom drugom ili istom modelu. Imenuju se sa brojem dekompozicionog dijagrama koji sadrži pozvani pravougaonik zajedno sa brojem pozivnog pravougaonika.

Dakle, elemente prikazane na slici 4.13 mogu se opisati rečenicom: “Pod Kontrolom, Aktivnost, od Ulaza pravi Izlaze, koristeći Mehanizme”.

Dijagrami opisuju sistem. Prvo se crta dijagram konteksta (A-0) koji se koristi za opisivanje celokupne svrhe i/ili funkcije sistema. Na dijagramu konteksta postoji samo jedna aktivnost, ulaz (ili skup ulaza), kontrole neophodne za pravilno funkcionisanje, mehanizmi koji obezbeđuju sredstva za izvršavanje funkcije i izlazi koji predstavljaju rezultat funkcije. S obzirom da je kompleksne sisteme, kada se posmatraju u globalu kao jedan proces, veoma teško razumeti, dati sistem se razlaže ili dekomponuje na svoje komponente odnosno na podsisteme. To se postiže primenom hijerarhijske dekompozicije dijagrama konteksta. Na sledećim slikama se prikazuje primer funkcionalnog modeliranja pomoću IDEF0 tehnike na modelu poslovanja preduzeća.

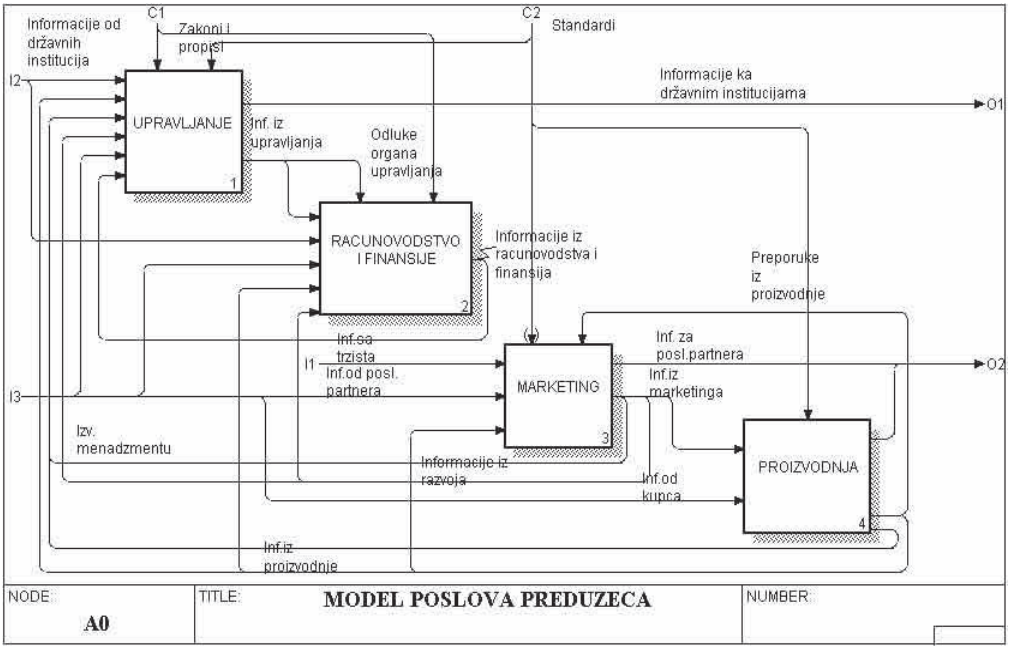



Slika 4.14. Dijagram konteksta

Sa dijagrama konteksta uočava se sledeće:

- ulazne informacije su informacije sa trišta, informacija od kupca od koga se dobija povratna informacija o upotrebi proizvoda i informacije od poslovnih parntera koji utiču na snabdevanje, a time i na lanac vrednosti organizacije.
- izlazne informacije su informacije za poslovnog partnera (informisanost o nabavci i prodaji) i informacije za kupca koje treba da zadovolje potrebe kupca za kvalitetnim proizvodom.

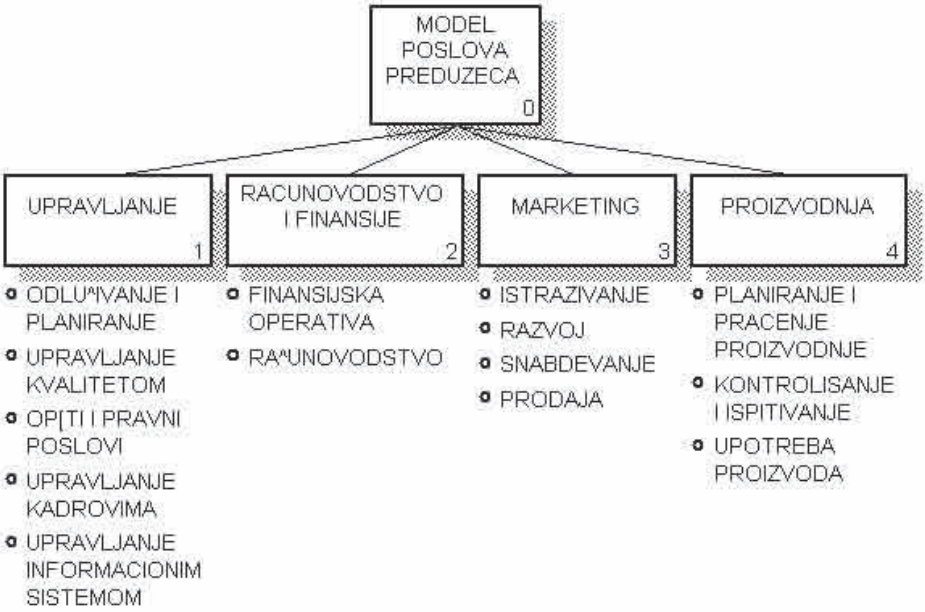
- Kontrole su zakoni i propisi (odnose se na funkciju finansija) i standardi (vezani za proizvodnju i marketing).
- Mehanizmi su postojanje odgovarajuće hardverske platforme klijent/server organizacije, sistem za upravljanje bazama podataka, zaposleni i dr.




Slika 4.15. Dekompozicioni dijagram modela poslova preduzeća

Na slici 4.15 se prikazuje dekompozicioni dijagram modela poslova preduzeća i uočavaju se četiri globalna procesa: upravljanje, finansije, marketing i proizvodnja. Za razliku od DTP-a gde se procesi nisu mogli direktno povezivati, kod IDEF0 standarda to nije slučaj. Naime, strelice koje izlaze iz procesa mogu direktno ulaziti u druge procese, s tim što treba obratiti veliku pažnju da ukoliko te izlazne strelice predstavljaju ulaze u drugi proces, treba da se i prikažu na strani ulaza, a ne na strani kontrola ili mehanizama.

Na osnovu detaljno iscrtanih dekompozicionih dijagrama, može se izvući stablo aktivnosti koje predstavlja pogled na sve procese u sistemu. Na slici 4.16 se prikazuju samo prva tri nivoa, zbog veličine slike.




Slika 4.16. Stablo aktivnosti modela poslova preduzeća

Model procesa se primenjuje za:

- strateško planiranje sistema – služi za definisanje arhitekture sistema u okviru strateškog plana, pri čemu se radi model procesa preduzeća koji je najčešće u formi dijagrama dekompozicije funkcija ili dijagrama konteksta;
- reinženjering poslovnih procesa – analiza i restruktuiranje poslovnih procesa radi poboljšanja produktivnosti pre primene informacionih tehnologija;
- analizu sistema – logički modeli procesa sistema ili aplikacije;
- dizajn sistema – fizički modeli procesa.

## 4.4. Modeliranje podataka i analiza

Modeliranje podataka je tehnika za organizovanje i dokumentovanje strukture podataka sistema. Modeliranje podataka se ponekad zove modeliranje baze podataka jer se model podataka na kraju implementira kao baza podataka. Modeliranje podataka je naše apstraktno viđenje stanja realnog sistema.

Model podataka je pojednostavljeno predstavljanje realnog sistema preko skupa objekata (entiteta), veza između objekata i njihovih atributa. Model podataka (u literaturi definisan kao Model Objekti-Veze - MOV ili *Entity-Relationship* - E-R model), preko skupa podataka i njihovih međusobnih veza, predstavlja stanje sistema u jednom trenutku vremena i sadrži skup informacija o prošlosti i sadašnjosti sistema koja je potrebna da se pod dejstvom budućih poznatih ulaza mogu odrediti njegovi budući izlazi.

Postoji nekoliko notacija modela objekti-veze. Mnogi su imenovani prema njihovim tvorcima (npr., Chen, Martin, Bachman, Merise) ili prema objavljenim standardima (npr., IDEF1X). Svi „jezici" modeliranja podataka podržavaju iste fundamentalne koncepte i konstrukcije.

### 4.4.1. Osnovni pojmovi modela podataka

Svi sistemi sadrže podatke. Podaci opisuju neke „stvari". Posmatrajmo jedan školski sistem. Školski sistem sadrži podatke koji opisuju sledeće stvari Učenik, Učitelji, Časovi, Učionice itd. Konkretni podaci koji opisuju učenika su ime, adresa, telefon, datum rođenja, pol, smer, uspeh itd. Koncept koji se koristi za apstraktno predstavljanje stvari sa sličnim podacima je *objekat*.

**Objekat** je klasa osoba, mesta, objekata, događaja ili koncepata o kojima treba da prikupljamo i skladištimo podatke. Objekat je nešto što se može videti, dodirnuti ili drugačije osetiti, koji ima svoja svojstva i ponašanja i o kome korisnici mogu da skladište podatke. Posmatrajte okruženje u kome se trenutno nalazite. Pogledajte okolo. Koji objekti su prisutni unutar vašeg okruženja? Možda vidite vrata, prozore, knjige, stolice ili stolove. Sve su to objekti koji se mogu videti i dodirniti, pa čak i vi predstavljate jedan vidan objekat. Pošto smo definisali objekat kao nešto što je moguće videti, dodirnuti ili drugačije osetiti, postavlja se pitanje koji su to objekti

koji se mogu na neki drugi način osetiti. Na primer, ukoliko očekujete telefonski poziv, to je nešto što možete da osetite. Ukoliko ste na poslovnom sastanku i poslovni sastanak je takođe nešto što se može identifikovati i na kome se može učestvovati. To znači da su i telefonski poziv i poslovni sastanak takođe objekti.

Tipovi objekata se mogu klasifikovati u osobe, mesta, stvari ili događaje. U okviru tipa objekta osobe mogu se svrstati radnici, klijenti, prodavci, studenti i dr. Odelenje, agencija, skladišta, zgrade, kampus su primeri tipa objekata mesta. Primeri tipa objekata stvari uključuju knjigu, proizvod, vozilo, mašinu, sirovine, softverski paket, opremu, DVD i dr. Na kraju objekti događaja uključuju porudžbinu, plaćanje, račun, aplikaciju, registraciju ili rezervaciju i sl. Grafički prikaz objekta po *Chen*-ovoj notaciji je u obliku pravougaonika.

Objekti imaju svoj identitet koji je definisan svojstvima ili atributima. **Atributi** su podaci koji predstavljaju karakteristike objekta. Na primer, pretpostavimo da smo zainteresovani za sledeće atribute objekta klijent: šifra klijenta, naziv klijenta, adresa, tip klijenta, telefon, status na računu. Neprikazuju se svi atributi koji opisuju dati objekat, već samo oni za koje je korisnik zainteresovan da drži u bazi podataka. Neki atributi se mogu logički grupisati u složeni atribut. Složeni atribut se sastoji od više primitivnih atributa. Na primer, adresa je složeni atribut jer se sastoji od sledećih primitivnih atributa: ulica, broj, grad, poštanski broj, država.

Za svaki atribut se uglavnom definišu: tip podatka, domen i difoltna vrednost. **Tip podatka** definiše koja klasa podataka može biti skladištena u taj atribut. Primeri tipova podataka su dati na tabeli 4.6. **Domen** definiše koje vrednosti može da ima jedan atribut. Na primer, ispitna ocena studenta uzima vrednosti iz domena 5 i 10. **Difoltna vrednost** je ona vrednost koja će biti uskladištena za dati atribut ukoliko je korisnik ne promeni.

Ljudi rado klasifikuju stvari, ne bi li našli sličnosti među njima i prema tome ih grupisali. Stvari sa sličnim svojstvima se grupišu zajedno. Na primer, Petar Petrović i Marija Marjanović imaju svoja imena, adrese i zanimanja i pošto oboje rade u istom preduzeću, možemo ih svrstati u objekat zaposlenih radnika. Konkretan objekat se naziva primerkom ili instancom (*instance*) objekta. Na primer, Petar Petrović je instanca objekta Zaposleni.

Tabela 4.6. Logički tipovi podataka za atribute

| Logički tip podatka | Logičko značenje |
|---|---|
| NUMBER | Bilo koji decimalni ili celobrojni broj. |
| TEXT | String karaktera uključujući i brojeve sa kojima se ne može obavljati aritmetička operacija ili operacija poređenja. |
| MEMO | Kao i TEXT ali sa neodređenom veličinom. |
| DATE | Bilo koji oblik datuma. |
| TIME | Bilo koji oblik vremena, odnosno časova. |
| YES/NO | Atribut koji može da ima samo ove dve vrednosti. |
| VALUE SET | Skraćeni skup vrednosti. (Na primer, JR=junior, SR=senior). |
| OLE OBJECT | Na primer slika. |
| HYPERLINK | URL adresa (npr., *e-mail*) |

Jedan objekat ima više instanci. Na primer, objekat Student će imati i nekoliko hiljada unešenih studenata (Slika 4.17). Svaka instanca objekta mora da ima jedinstveni **ključ** po kome će se pretraživati u bazi podataka. Na primer, s obzirom da atribut Broj indeksa zadovoljava osobine ključa, jer jedinstveno identifikuje svaku instancu objekta Student (svaki student ima jedinstveni broj indeksa i ne može ga deliti sa drugim studentom), tada će Broj indeksa predstavljati ključ objekta Student.

Objekat: STUDENT

| Broj indeksa | Ime | Prezime | Adresa | PttBr | Broj telefona |
|---|---|---|---|---|---|
| 122/03 | Ana | Anić | Nušićeva 2 | 11000 | 011/123456 |
| 130/04 | Pera | Perić | Nemanjina 3 | 21000 | 011/234567 |
| 140/02 | Lea | Leić | Bul. umetn. 9 | 18000 | 018/456901 |

Atributi

Instance

Slika 4.17. Prikaz atributa i instanci za objekat Student

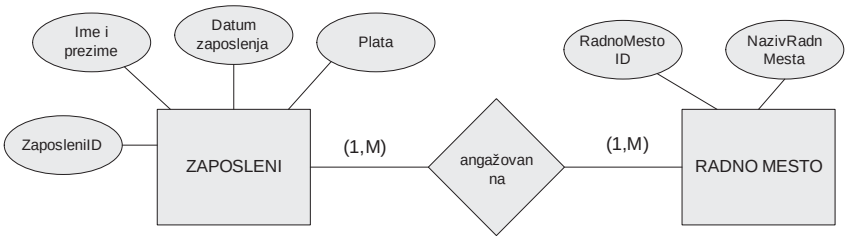
Nekada je neophodno da više od jednog atributa jedinstveno identifikuje objekat. Ta grupa atributa koja jedinstveno identifikuju objekat se zove **složeni ključ**. Na primer, ukoliko bi projektovali informacioni sistem nekog video kluba, koji ima više kopija kaseta, složeni ključ bi bio šifra naslova i broj kopije video kasete. Oba ova atributa jedinstveno identifikuju konkretnu video kasetu.

Da bi neki atribut mogao da bude primarni ključ, on mora da zadovolji sledeće uslove:

- jednoznačnost – ne smeju da postoje dve pojave za istim vrednostima svih ključnih atributa;
- minimalnost – da ne postoji podskup atributa ključa koji je takođe jednoznačan (npr., složeni primarni ključ objekta Osobe ne može biti kombinacija atributa JMBG i Prezime, jer Prezime nema osobinu jednoznačnosti);
- određenost – postojanje vrednosti u trenutku stvaranja instance;
- stabilnost – otpornost na promene tokom vremena;
- raspoloživost – dostupnost svim korisnicima;
- neutralnost – prema značenju vrednosti ključa (samogovoreće šifre).

Objekat može imati više atributa koji zadovoljavaju karakteristike primarnog ključa. Na primer, objekat Radnik se može jedinstveno identifikovati preko matičnog ličnog broja ili preko šifre zaposlenog ili preko *e-mail* adrese. Svaki od ovih atributa se nazivaju kandidati za ključ. **Kandidati za ključ** su kandidati za primarni ključ. **Primarni ključ** (*Primary key*) je kandidat za ključ koji će se najčešće koristiti da jedinstveno identifikuje dati objekat. Primarni ključ se definiše kao jedan ili više atributa čija vrednost jedinstveno identifikuje jednu instancu objekta. Primarni ključ mora da zadovolji osobine jedinstvenosti i neredundantnosti. Difoltna vrednost primarnog ključa je *Not Null*, odnosno ključ ne sme da bude prazno polje, jer onda neće moći da jedinstveno identifikuje datu instancu objekta. Svi drugi kandidati za ključ koji nisu izabrani za primarni ključ se zovu **alternativni ključevi.**

Objekti ne ekzistiraju sami već moraju biti u nekoj **relaciji** ili **vezi** sa drugim objektima. Ukoliko posmatramo objekte Zaposleni i RadnoMesto, jedna od relacija između ova dva objekta će biti Angažovan, što znači da je zaposleni angažovan na određeno radno mesto u nekoj organizaciji. Na slici 4.18 je prikazana relacija ova dva objekta sa svojim atributima, prema Chen-ovoj notaciji.



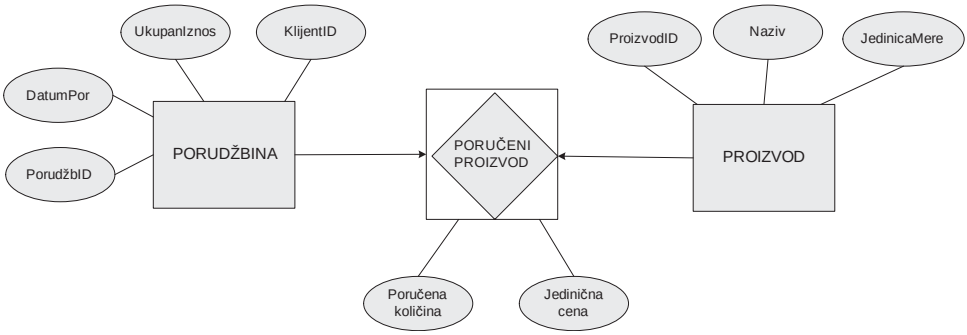

Slika 4.18. Relacija Angažovanje zaposlenih na radna mesta

Jedna od bitnih karakteristika veza između objekata je kardinalnost (tip veze). **Kardinalnost** definiše minimalni i maksimalni broj događaja jednog objekta koji se nalazi u konkretnoj relaciji sa drugim objektom. Pošto su sve relacije dvosmerne, kardinalnost se mora definisati za oba smera. Kao što je prikazano na slici 4.18, u zagradama, konkretan zaposleni (npr., Petar Petrović) može da bude angažovan minimum na jedno (1), a maksimalno na više radnih mesta (M). Na primer, Petar je angažovan na radno mesto programera i konsultanta za prodaju softvera. Međutim, na određeno radno mesto može biti minimum jedan (1), a maksimalno više zaposlenih (M). Na primer, na radno mesto konsultanta može biti angažovano više zaposlenih, odnosno u organizaciji je zaposleno više konsultanata. Grafička notacija kardinalnosti zavisi od notacija modela objekti veze. Na slici 4.19, je prikazana notacija kardinalnosti prema Martin-ovoj notaciji ER modela.

| Interpretacija kardinalnosti | Minimum | Maksimum | Grafička notacija |
|---|---|---|---|
| Tačno jedan | 1 | 1 | |
| Nula ili jedan | 0 | 1 | |
| Jedan ili više | 1 | >1 | |
| Nula, jedan ili više | 0 | >1 | |
| Više od jedan | >1 | >1 | |

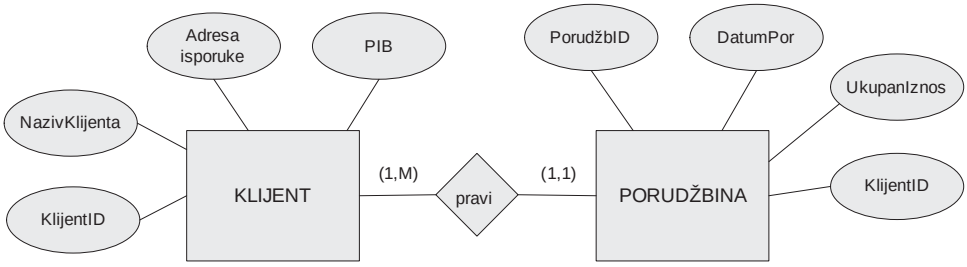
Slika 4.19. Martin-ova notacija kardinalnosti

**Agregacija** istovremeno predstavlja i objekat i vezu, odnosno udruženi ili asocijativni objekat (*associative entity*), između dva ili više objekta (Slika 4.20). Ukoliko postoji potreba da se u toku relacije čuvaju dodatni atributi, tada relacija postaje udruženi objekat. Udruženi objekat je objekat koji nasleđuje primarne ključeve svojih tzv. „roditelja“. Na primer, atributi udruženog objekta Poručeni proizvod su Poručena količina i Jedinična cena (Slika 4.20), a njeni ključevi bi bili atributi PorudžbinaID i ProizvodID.




Slika 4.20. Agregacija “Poručeni proizvod”

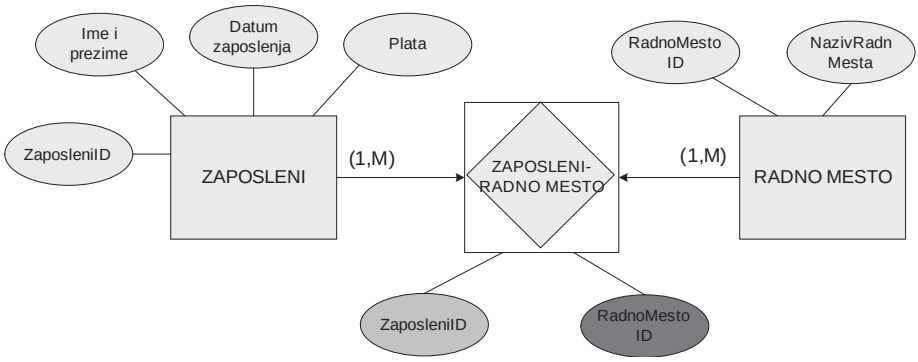
Objekti se povezuju pomoću spoljnjeg ključa. **Spoljni ključ** (*Foreign key*) je atribut ili grupa atributa jednog objekta, čija se vrednost koristi za povezivanje sa vrednošću primarnog ključa drugog objekta. Odnosno, spoljni ključ objekta 1 je primarni ključ objekta 2 sa kojim je objekat 1 u vezi. Spoljni ključevi i njima odgovarajući primarni ključevi definisani su nad istim domenom. Na primer, pogledajmo relaciju Klijent i Porudžbina sa slike 4.21. S obzirom da klijent može da ima više porudžbina, tada ćemo primarni ključ objekta Klijent (u našem primeru to je KlijentID) preneti u objekat Porudžbina, gde će taj atribut predstavljati njegov spoljni ključ. Za konkretnu porudžbinu, postojanjem spoljnog ključa KlijentID, znaće se ko je načinio konkretnu porudžbinu.




Slika 4.21. Relacija Klijent - Porudžbina

Da rezimiramo, objekat Porudžbina će imati primarni ključ PorudžbinaID, a spoljni ključ je KlijentID koji ga povezuje sa objektom Klijent. Obično, gde postoji kardinalnost tipa (1,1), onda taj objekat uzima primarni ključ objekta sa kojim je u vezi, a koji će predstavljati njegov spoljni ključ. Kao što se vidi sa slike 4.21, kardinalnost kod objekta Porudžbina je (1,1), tako da će spoljni ključ stajati na njegovoj strani.

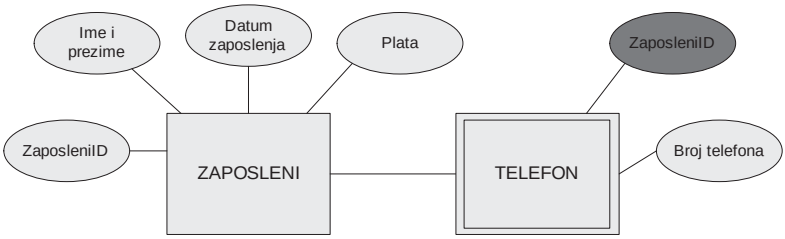
Kod onih relacija gde postoji veza više prema više, onda se ubacuje među objekat, kako bi rešio problem višeznačne zavisnosti. Ukoliko posmatramo prethodni primer opisan na slici 4.18, uočava se da postoji veza više prema više, odnosno zaposlen može da bude angažovan na više radnih mesta i na jednom radnom mestu može da bude angažovano više zaposlenih. U tom slučaju se ubacivanjem udruženog objekta pravi veza jedan prema više (1,M) prema ovim objektima. Jedini atributi ovog među-objekta su primarni ključevi objekata koje povezuje (Slika 4.22).




Slika 4.22. Rešavanje problema višeznačne zavisnosti

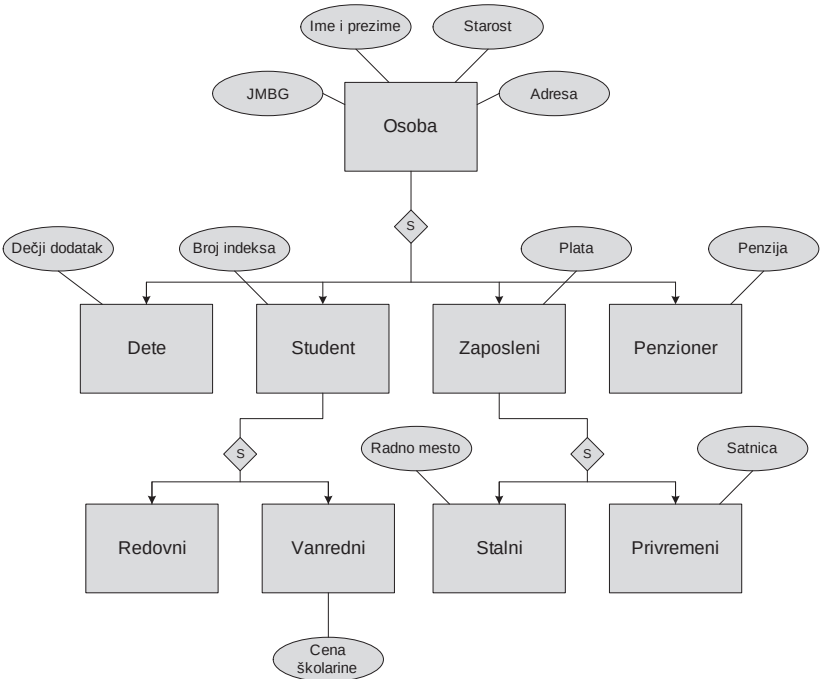
Problem određivanja spoljnih ključeva će biti detaljnije obrazložen kroz postupak normalizacije koji će kasnije biti objašnjen.

Ukoliko postoji potreba da se ponavljaju vrednosti jednog atributa u okviru određenog objekta, tada se takav atribut izdvaja i formira se slab objekat. Na primer, za objekat zaposleni koji ima atribut telefon, želimo da pamtimo više telefona. Tada ćemo napraviti slab objekat telefon koji će naslediti primarni ključ svog nad objekta (Slika 4.23).




Slika 4.23. Primer slabog objekta

**Generalizacija** je tehnika gde se objekti sa zajedničkim atributima, vezama i/ili operacijama, grupišu (generalizuju) u jedan objekat koji se zove **nadtip**. Inverzni postupak, gde se za neki tip objekta, definišu njegovi podtipovi, koji imaju neke njima specifične atribute, veze i/ili operacije, je **specijalizacija.** Na primer, objekti Student, Zaposleni, Dete i Penzioner imaju zajedničke atribute kao što su Ime i prezime, JMBG, Starost i Adresa, tako da se mogu predstaviti generičkim tipom Osoba. S obzirom da Student ima sebi svojstvene atribute kao što je Broj indeksa, a Zaposleni ima npr., Platu, tada se objekat Osoba specijalizuje u objekte Student i Zaposleni (Slika 4.24).




Slika 4.24. Primer generalizacije/specijalizacije

### 4.4.2. Modeliranje podataka

Mnoge kompanije prvo obavljaju strateško planiranje gde se razvija strateški plan informacionog sistema u kome se definišu vizija i arhitektura informacionog sistema. Arhitektura informacionih sistema uključuje **model podataka kompanije**. Model podataka kompanije identifikuje osnovne objekte bez opisivanja njegovih atributa ili ključeva, a takođe može da uključi, a ne mora, relacije između objekata.

**Prvi zadatak** u modeliranju podataka je da se odrede osnovni objekti sistema. Postoji nekoliko tehnika koje se mogu koristiti za identifikaciju objekata:

- Tokom intervjua sa vlasnicima i korisnicima sistema, treba obratiti pažnju na ključne reči njihove diskusije. Na primer, ukoliko korisnik kaže «Treba informisati naše stalne kupce, o novom kvalitetu proizvoda», primetićemo da su ključne reči u ovoj rečenici Kupci i Proizvod, a oni su ujedno i objekti sistema.
- U toku intervjua treba pitati vlasnike i korisnike sistema da identifikuju one stvari za koje žele da prikupljaju, skladište i dobijaju informacije. Često su to objekti koji treba da budu opisani u modelu podataka.
- Druga tehnika za identifikaciju objekta je da se prouče postojeći formulari i kartoteke. Neki formulari identifikuju događaj objekata. Na primer, porudžbina, uplata, depozit itd.
- Neki CASE alati takođe mogu da identifikuju objekte.

Kada se identifikuju objekti sistema, treba im dati jednostavna, značajna, poslovno-orjentisana imena.

**Drugi zadatak** u modeliranju podataka je identifikovanje atributa objekata. Predlaže se sledeći način:

- Mnoge kompanije koriste standardna imena i skraćenice. Administrator podataka obično održava takve standarde.
- Pažljivo birajte imena atributa. Mnogi nazivi atributa imaju istu osnovu, npr. ime, adresa, datum, njih bi trebalo razdvojiti, kao npr., NazivDobavljača, NazivKlijenta i sl.

**Treći korak** je da se identifikuju ključevi za svaki objekat. Predlaže se sledeće:

- Vrednost ključa ne sme da se menja u toku veka trajanja svakog objekta. Na primer, Prezime osobe se ne može uzeti kao ključ, jer osoba može da promeni svoje prezime ukoliko se venča ili razvede.
- Vrednost ključa ne može da bude prazno polje.
- Kontrole ključa moraju da budu instalirane, kako bi vrednost ključa bila uvek validna.

**Četvrti korak** je povezivanje objekata. Relacija je veza između dva ili više objekata, tj. predstavlja odnos koji postoji među objektima, bilo u realnosti ili u mislima. Nazivi relacija treba da budu glagoli, npr, upisuje, polaže, predaje itd. Objekat od koga je uspostavljena veza zove se "roditelj" (*parent*) ili domen, a objekat ka kome je uspostavljena veza zove se "dete" (*child*) ili kodomen. U toku određivanja relacija treba da se odrede i tipovi veza ili kardinalnosti.

U toku određivanja objekata sistema, njihovih atributa, ključeva i relacija iscrtava se **kontekstualni (konceptualni) model podataka**, na kome će se jasno uočiti objekti i njihove međusobne veze.

Logički model podataka je potpuno određen ukoliko su definisane sledeće tri komponente:

- Struktura podataka - kojima se definišu statičke karakteristike sistema (opis objekata, atributi i veze).
- Ograničenja - logička ograničenja na podatke (pravila integriteta) koja se ne mogu definisati preko strukture modela podataka (strukturna i vrednosna ograničenja) i odnose se na definisanje poslovnih pravila.
- Skup operatora (operacije) - definiše dinamičku interpretaciju podataka kroz njihovu obradu (održavanje baze podataka i pretraživanje) ima uticaja na definisanje fizičkog nivoa modela i verifikaciju finalnog dizajna.

**Peti korak** je normalizacija relacionog modela podataka. Normalizacija je tehnika analize podataka koja organizuje atribute na način kako bi se dobili neredundantni, stabilni, fleksibilni i prilagodljivi objekti. U toku normalizacije proverava se da li su atributi dobro postavljeni i određuju se odgovarajući spoljni ključevi koji povezuju objekte.

**Poslednji korak** je prevođenje logičkog modela podataka u fizički model (šema baze podataka) u zavisnosti od izabranog sistema za upravljanje bazom podataka (*Database Management System* – DBMS).

### 4.4.3. IDEF1X standard za modeliranje podataka

Postoji mnogo različitih verzija modela objekti veze, koji se u većini slučajeva međusobno razlikuju samo sintaksno. U praksi najčešće korišćena verzija je IDEF1X standard američke vlade za modeliranje podataka. IDEF1X je realizovan kroz ERwin CASE alat.

Kod IDEF1X se koriste pojmovi roditelj i dete. Objekat od koga je uspostavljena veza se zove roditelj, a objekat ka kome je uspostavljena veza zove se dete.

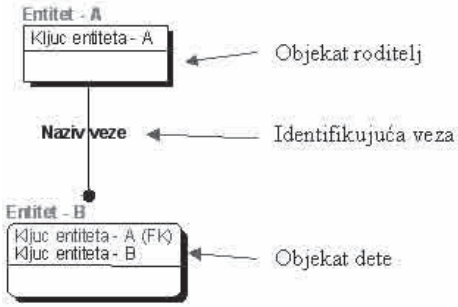
U IDEF1X definišu se dve vrste objekata:

- **nezavisni objekti** – objekti koji mogu postojati nezavisno od drugih objekata u sistemu tzv. jaki objekti.
- **zavisni objekti** – objekti čija egzistencija i identifikacija zavise od drugog ili drugih objekata. Postoje četiri grupe zavisnih objekata:
  - slab objekat – objekat koji se ponavlja više puta za određeni nezavisni objekat. Na primer, objekti Faktura i StavkaFakture. Za objekat StavkaFakture se kaže da je slab objekat, jer potpuno zavisi od objekta Faktura.
  - asocijativni objekti – predstavljaju vezu više objekata;
  - projektni objekti – slični su asocijativnim, samo što nemaju sopstvene atribute;
  - objekti kategorije (generalizacija/specijalizacija) – predstavljaju podkategoriju objekta. Kod objekta tipa kategorije definišu se: nadređeni tip koji ima zajedničke osobine (npr., Partner) i podređeni objekti (npr., FizičkoLice i PravnoLice), koji se identifikuju ključem nadređenog i poseduju svoje specifične osobine.

Kod IDEF1X standarda razlikuju se sledeći tipovi veza:

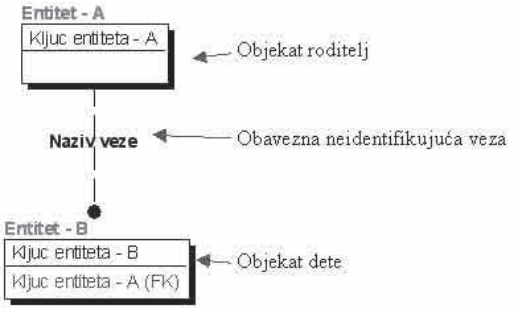
- **Identifikujuće** veze koje objekat dete identifikuje kroz njegovu vezu sa objektom roditelj. Identifikujuća veza je prikazana punom linijom i povezuje objekat roditelja sa objektom dete sa tačkom na strani objekta dete. U identifikujućoj vezi objekat roditelj ima svoj nezavisni primarni ključ (Ključ A), a objekat dete ima složeni ključ koji se sastoji od svog ključa (Ključ B) i prenesenog roditeljskog ključa (Ključ A(FK)).

Dakle, instance objekta roditelj se definišu nezavisno, a instance objekta dete se ne mogu identifikovati bez identifikatora objekta roditelj.



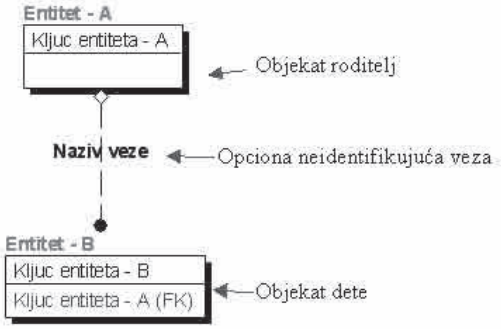

Slika 4.25. Identifikujuća veza

- **Neidentifikujuća** veza ne identifikuje dete preko identifikatora roditelja. Ukoliko se svaki primerak objekta dete može jedinstveno identifikovati bez znanja veze sa primerkom objekta roditelj, onda se takva veza definiše kao neidentifikujuća veza. Neidentifikujuće veze su prikazane isprekidanom linijom koja povezuju objekat roditelj i objekat dete sa tačkom na strani objekta dete. Neidentifikujuća ili slaba veza zavisi od načina definisanja ključeva od roditelja ka detetu i to na dva načina: obavezna i neobavezna (opciona) neidentifikujuća veza. Ako je veza obavezna (*No Nulls* ili *Mandatory*) iz perspektive roditelja, onda je dete egzistencijalno zavisno od roditelja (Slika 4.26).




Slika 4.26. Neidentifikujuća obavezna veza

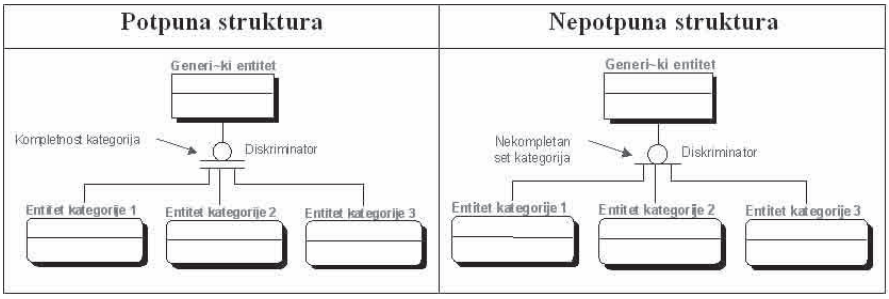
Ako je veza neobavezna (*Nulls Allowed* ili *Optional*), tada dete niti je egzistencijalno niti identifikaciono zavisno, ali poštuje tu vezu.




Slika 4.27. Neidentifikujuća neobavezna veza

ERwin koristi romb (*diamond*) da naznači slučaj egzistencijalne i identifikacione zavisnosti. Romb može postojati samo u slabim vezama (pošto je jaka veza u okviru primarnog ključa, a primarni ključ ne može da ima *Null* vrednost).

**Veza kategorije** tj. veze prema podtipovima je definisana za hijerarhijsku vezu između nadređenog generičkog objekta koji sadrži zajedničke osobine podređenih objekata kategorije. Ovaj tip veze se deli na kompletnu kategoriju ili tzv. potpune strukture gde su navedene sve moguće kategorije (podtipovi) nadređenog generičkog objekta i nekompletnu kategoriju ili tzv. nepotpunu strukturu, gde su navedene samo kategorije (podtipovi) nadređenog generičkog objekta.

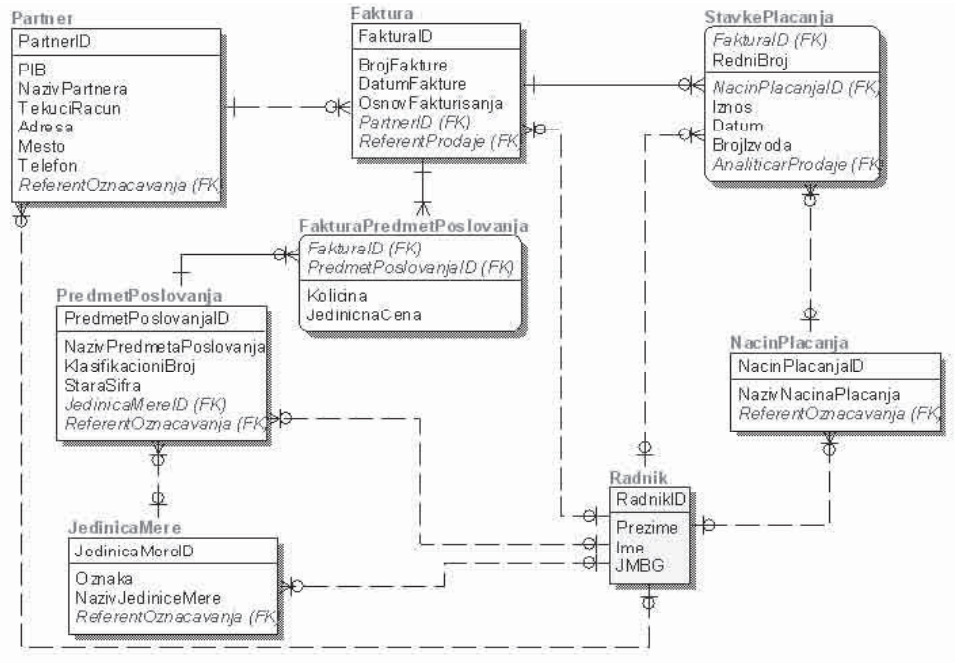



Slika 4.28. Vrste objekata kategorije

Na sledećem primeru pokazuje se model podataka prema IDEF1X standardu. Polazi se od standardizovanog obrasca za fakturu (ISO 7372) - EDIFACT, koji se koristi u međunarodnoj trgovini (Slika 4.29), na osnovu koga je napravljen logički model podataka prema IDEF1X standardu (Slika 4.30).

| | | | | |
|---|---|---|---|---|
| ISPORUČILAC<br>*PERIHARD INŽINJERING*<br>*I. MILUTINOVICA 25*<br>*11000 BEOGRAD* | | DATUM I BR. FAKTURE<br>*960321 547XRTW*<br>**DRUGE REF.**<br>*FAKTURA* | | |
| PRIMALAC<br>*SITJ KNEZA MILOŠA 9*<br>*11000 BEOGRAD* | | **KUPAC (AKO NIJE PRIMALAC)** | | |
| | | ZEMLJA POREKLA *USA* | | |
| DETALJI O PREVOZU<br>*KAMION, AVION* | | USLOVI ISPRUKE I PLAĆANJA | | |
| **OTPREMNE OZNAKE** | **BROJ I VRSTA; OPIS ROBE; PAKOVANJE** | | **MASA kg** | **ZAPREMINA m³** |
| **OPIS ARTIKLA (ŠIFRA ILI NAZIV)** | | **KOLIČINA** | **J. CENA** | **IZNOS** |
| *LASERSKI STAMPAC*<br>*TONER*<br>*RAČUNAR* | | *3*<br>*2*<br>*1* | *12000*<br>*3000*<br>*8500* | *36000*<br>*6000*<br>*8500* |
| | **AMBALAŽA** | | | |
| | **PREVOZ** | | | |
| | **OSTALI TROŠKOVI** | | | |
| | **OSIGURANJE** | | | |
| | **UKUPNO** | | | *50500* |

Slika 4.29. Obrazac EDIFACT fakture




Slika 4.30. Logički model podataka EDIFACT fakture [12]

### 4.4.4. Analiza modela podataka

Ukoliko model podataka efektivno izražava poslovne zahteve za podacima, to ujedno ne znači da je takav model i dobar model. Njegova struktura može biti takva da umanjuje fleksibilnost, proširljivost ili stvara nepotrebnu redundantnost. Kod kreiranja modela podataka treba se pridržavati sledećih kriterijuma:

- Dobar model podataka je jednostavan – atributi jednog objekta treba da opisuju samo taj objekat.
- Dobar model podataka je u osnovi neredundantan.
- Dobar model podataka treba da bude fleksibilan i adaptibilan za buduće potrebe.

Tehnika koja se koristi za poboljšanje modela podataka u fazi pripreme za fizički dizajn baze podataka se naziva analiza podataka. Analiza podataka je proces koji priprema model podataka za implementaciju jednostavne, neredundantne, fleksibilne i adaptibilne baze podataka. Ova tehnika se naziva normalizacija.

**Normalizacija** je tehnika analize podataka koja organizuje, grupiše atribute podataka tako da formiraju neredundantne, stabilne, fleksibilne i prilagodljive objekte. Ovde nećemo prikazati matematički pristup koji obuhvata relacionu algebru i relacioni račun, već ćemo se osvrnuti samo na primenu normalnih formi. Pravilno projektovana baza podataka mora imati strukturu koja osigurava da u radu sa njom ne može doći do neželjenih anomalija, kao što su na primer, uništavanje određenih podataka ili neusklađenost između memorisanih podataka kao posledica ažuriranja baze podataka.

Tvorac normalizacije je Codd koji je definisao postupak koji dovodi projektanta relacione baze podataka do željenog rezultata. Codd je definisao tri nivoa normalizacije, da bi kasnije bilo dokazano da u nekim izuzetnim primerima, relacije koje su u trećoj normalnoj formi, još uvek pokazuju izvesne anomalije. Na treću normalnu formu nadovezuje se Boyce-Coddova normalna forma koja se bavi problemima preklapajućih kandidata za ključ. Probleme višeznačne zavisnosti rešava Fagin definišući četvrtu normalnu formu (4NF). Kasnije je formulisana i peta normalna forma koja definiše zavisnost spajanja relacija. Da bi se stalo u definisanju novih normalnih formi, Fagin je 1981. godine dao najopštiju definiciju normalne forme ključeva i domena.

Da bi se mogao opisati postupak normalizacije, potrebno je prethodno opisati pojmove funkcionalne, potpune funkcionalne, tranzitivne i višeznačne zavisnosti.

### Funkcionalna zavisnost

Funkcionalna zavisnost se može definisati između jednostavnog atributa i složenog ključa (više atributa). Ako je svakoj vrednosti atributa A u relaciji R priključena samo jedna vrednost atributa B u istoj relaciji, onda je atribut B funkcionalno zavisan od atributa A. Ako je moguće svakom paru vrednost atributa A i B relacije R priključiti tačno jednu vrednost C iste relacije, tada je atribut C funkcionalno zavisan o sastavljenom atributu A i B. Na osnovu definicije funkcionalne zavisnosti, potpuna funkcionalna zavisnost se definiše na sledeći način:

**Atribut B je potpuno funkcionalno zavisan od atributa A iste relacije, ako je funkcionalno zavisan od atributa A, a ne od nekog sastavnog dela atributa A (u slučaju kada je atribut A sastavljen).**

Drugim rečima, potpuna funkcionalna zavisnost se posmatra kada je ključ tabele isključivo sastavljen od više atributa. Kao primer pokazaćemo tabelu POSLOVI_ZAPOSLENIH.

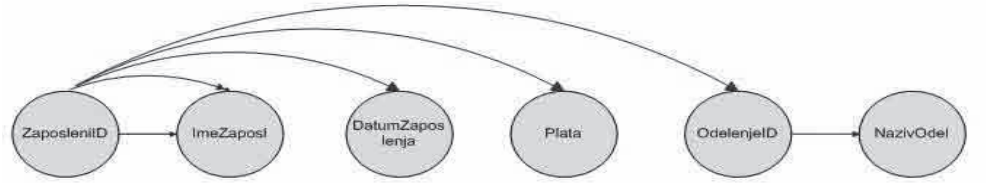
POSLOVI_ZAPOSLENIH

| ZaposleniID | PosaoID | BrojČasova | OpisPosla |
|---|---|---|---|
| 1001 | 202 | 40 | Komerc |
| 1005 | 222 | 35 | Finans |
| 1011 | 211 | 40 | Komerc |

U ovom primeru OpisPosla je funkcionalno zavisan od PosaoID (koji je deo ključa, pa zbog toga nije potpuno funkcionalno zavisan o ključu ove tabele). Atribut BrojČasova je potpuno funkcionalno zavisan o celom ključu, te je broj časova rada (40) određenog radnika (1001) na određenom poslu (202).

### Tranzitivna funkcionalna zavisnost

Atribut C je tranzitivno funkcionalno zavisan od atributa A, ako je funkcionalno zavisan od A i ako je funkcionalno zavisan od nekog atributa B koji je i sam funkcionalno zavisan od A. Grafički prikaz opšte definicije tranzitivne funkcionalne zavisnosti relacije ZAPOSLENI se prikazuje na sledećoj slici.




Slika 4.31. Primer tranzitivne funkcionalne zavisnosti

U ovom primeru NazivOdelenja je funkcionalno zavisan i od primarnog ključa ZaposleniID i od jednog njegovog neključnog atributa OdelenjeID, što dovodi do pojave tranzitivne funkcionalne zavisnosti. Tranzitivna funkcionalna zavisnost dovodi do redundantnosti podataka u bazi podataka.

Nakon definisanja postojećih zavisnosti, prelazimo na objašnjavanje postupka normalizacije koji započinje sa prvom normalnom formom. Proces normalizacije predstavlja transformaciju početne tabele u jednu ili više korektnih tabela (relacija) u kojima su svi atributi potpuno funkcionalno zavisni od ključa, a međusobno funkcionalno nezavisni. Cilj normalizacije je da eliminiše redundansu u atributima i sa time omogući lakše održavanje integriteta podataka i jednostavniju manipulaciju.

### Prva normalna forma

Osnovne operacije održavanja baze podataka su: dodavanje nove n-torke u relaciju, izbacivanje neke n-torke iz relacije i izmena (ažuriranje) vrednosti nekog atributa u relaciji. Problemi pri izvođenju ovih ope-racija kao što su ponavljanje ovih operacija više puta, izbacivanje jednog logičkog skupa podataka koji dovodi do neželjenog izbacivanja drugih podataka ili logičko dodavanje se ne može izvršiti, nazivaju se anomalije u održavanju baze podataka.

Pre nego što se krene u definisanje tabela i njihovo normalizovanje, projektant informacionog sistema u okviru analize polazi od, na primer, nekog formulara, kao što su fakture ili dosiji radnika i sl. Formirajući na osnovu toga jednu relaciju, projektant pristupa proveri date relacije, odnosno pristupa normalizaciji, primenjujući prvu normalnu formu, koja glasi:

Relacija je u **Prvoj normalnoj formi** ukoliko su sve vrednosti njenih atributa atomske ili drugim rečima nisu dozvoljeni atributi ili grupe atributa "sa ponavljanjem", tzv. "tabele u tabeli".

Na primer, posmatrajmo polaznu relaciju FUDBALSKI_SUDIJA koja je sledećeg oblika:

FUDBALSKI_SUDIJA

| Sud ID | Ime Sudije | Adresa Sud | PttBr | Mesto | Utakm ID | Datum Utakm | Domaćin | Gost | Stadion | Rezultat |
|---|---|---|---|---|---|---|---|---|---|---|
| 1444 | Delević Dejan | Nusiceva 2 | 21000 | Novi Sad | 01/03 | 15.10.03 | Crvena Zvezda | Partizan | Marakana | 3:2 |
| | | | | | 09/03 | 20.10.03 | Obilić | Sartid | Obilić | 5:1 |
| | | | | | 11/03 | 03.11.03 | Partizan | Vojvodina | Partizan | 3:2 |
| 2223 | Stanišić Dejan | Bul JNA 15 | 11000 | Beograd | 04/04 | 20.04.04 | OFK | Napredak | OFK | 3:3 |
| | | | | | 21/04 | 21.04.04 | OFK | Obilić | OFK | 1:5 |

Datu relaciju FUDBALSKI_SUDIJA možemo normalizovati, odnosno svesti na 1NF, ako u svakoj vrsti prikazane tabele, za svaku odigranu utakmicu ponovimo i sve ostale podatke o sudiji. Međutim, ova relacija sadrži izvesne anomalije u održavanju baze podataka, kao što su na primer:

- Anomalije u dodavanju – ukoliko se žele ubaciti podaci o novom fudbalskom klubu, to se ne može uraditi sve dok određeni sudija ne sudi na utakmici u kojoj igra dati fudbalski klub.
- Anomalije u izbacivanju – ukoliko se žele obrisati podaci o jednom sudiji, onda se gube i sve informacije o održanim utakmicama.
- Anomalije u ažuriranju – ukoliko se promeni adresa stanovanja sudije, to se mora učiniti na onoliko mesta koliko utakmica je sudio dati sudija.

Postupkom normalizacije, logička struktura baze podataka se dovodi u takav oblik u kome se izbegavaju anomalije u održavanju i problemi u izveštavanju.

Da bi se smanjila redundansa do koje bi, očigledno, ovakva normalizacija dovela, možemo ovako dobijenu relaciju, operacijom projekcije[1], dekomponovati na sledeće dve:

FUDBALSKI_SUDIJA (SudijaID, ImeSudije, AdresaSud, PttBr, Mesto)

UTAKMICA (UtakmicaID, DatumUtakmice, Domaćin, Gost, Stadion, Rezultat, SudijaID)

1 Projekcija je unarna operacija relacione algebre koja iz neke relacije selektuje skup navedenih atrubuta, odnosno izvlači vertikalni podskup iz odgovarajuće tabele. Operacijom projekcije eliminišu se duplikati n-torki.

Pri daljem postupku normalizacije postavlja se pitanje da li se pri takvoj dekompoziciji gubi neka informacija iz polazne relacije, odnosno postavlja se zahtev da se ovakva dekompozicija vrši bez gubljenja informacija. Relacija R se dekomponuje u svoje projekcije bez gubljenja informacija ako prirodno spajanje tako dobijenih projekcija dovodi do polazne relacije.

*Heath*-ova teorema daje uslove pod kojima se može izvršiti dekompozicija relacije bez gubljenja informacija: Relacija R(A,B,C) gde su A,B i C podskupovi atributa, u kojoj važi R.A → R.B može se uvek dekomponovati u svoje projekcije R1(A,B) i R2(A,C) bez gubljenja informacija.

Pogledajmo još jedan primer nenormalizovane relacije DOBAVLJAČI koja je sledećeg oblika:

DOBAVLJAČI

| DobavljacID | NazivDob | AdresaDob | Ptt_B | ProizvodID | NazivProizvoda |
|---|---|---|---|---|---|
| 110 | SONY | Decanska 10 | 11000 | 1234 | Televizor |
| | | | | 1456 | DVD |
| | | | | 2156 | Walkman |
| | | | | 3522 | Digitalne kamere |
| 121 | COOL | Makedonska 2 | 21000 | 2222 | Muzička linija |
| | | | | 3321 | Frižider |
| | | | | 2244 | Usisivač |
| 131 | Tehnicom | Nusiceva 10 | 18000 | 3111 | Notebook |
| | | | | 1111 | Foto aparat |

Data relacija DOBAVLJAČI pokazuje izvesne anomalije u održavanju baze podataka, kao što su na primer:

- ukoliko se promeni adresa dobavljača to se mora učiniti na onoliko mesta koliko proizvoda se nabavlja od tog dobavljača.
- ukoliko se žele ubaciti u bazu informacije o novom proizvodu, to se ne može ubaciti sve dok ga neki dobavljač ne ponudi itd.

Nenormalizovanu relaciju DOBAVLJAČ operacijom projekcije, dekomponujemo na sledeće dve:

DOBAVLJACI (<u>DobavljacID</u>, NazivDob, AdresaDob, Ptt_Br)

PROIZVOD (<u>ProizvodID</u>, NazivProizvoda, DobavljaID)

### Druga normalna forma

Relacija R je u **Drugoj normalnoj formi** (2NF) ako i samo ako je u 1NF i svi njeni neključni atributi potpuno funkcionalno zavise od primarnog ključa. Neključni atributi su atributi koji nisu kandidati za ključ, niti deo kandidata za ključ.

Svođenje na 2NF vrši se dekompozicijom na taj način što se u jednoj projekciji ostavlja primarni ključ i svi atributi koji su potpuno funkcionalno zavisni od njega, a u drugim projekcijama se realizuju one funkcionalne zavisnosti koje su prouzrokovale nepotpune funkcionalne zavisnosti.

Ukoliko posmatramo sledeću relaciju:

PORUČENI_PROIZVOD (PorudžbinaID, ProizvodID, OpisProizvoda, PoručenaKol, JedCena)

Uzrok redundansi i anomalijama u održavanju je nepotpuna funkcionalna zavisnost atributa OpisProizvoda od složenog atributa PorudžbinaID, ProizvodID. Naime,

PorudžbinaID, ProizvodID → OpisProizvoda

PorudžbinaID → OpisProizvoda

ProizvodID → OpisProizvoda

Definicija druge normalne forme zabranjuje postojanje ovakve zavisnosti. Odnosno, s obzirom da je relacija R u 2NF, ukoliko svi njeni atributi daju jednoznačne činjenice samo o celom ključu, onda relacija PORUČENI_PROIZVOD nije u 2NF, jer OpisProizvoda daje jednoznačnu informaciju o delu ključa i to ProizvodID. Iz definicije 2NF i definicije potpune funkcionalne zavisnosti očigledno je da je svaka relacija sa prostim primarnim ključem u 2NF, jer prosti ključ nema semantički moguć pravi podskup.

Svođenje na 2NF vrši se sledećom dekompozicijom:

PORUČENI_PROIZVOD (PorudžbinaID, ProizvodID, PoručenaKol, JedCena)

PROIZVOD (ProizvodID, OpisProizvoda)

### Treća normalna forma

Relacija R je u **Trećoj normalnoj formi** (3NF) ako i samo ako je u 2NF i ako svi njeni neključni atributi netranzitivno funkcionalno zavise od primarnog ključa.

Ukoliko posmatramo relaciju:

FUDBALSKI_SUDIJA (SudijaID, ImeSudije, AdresaSud, PttBr, Mesto)

Evidentne su anomalije u održavanju baze podataka i redundantnost podataka usled postojanja tranzitivne funkcionalne zavisnosti atributa Mesto od atributa PttBr. Da bi smo datu relaciju sveli na 3 NF, vršimo dekompoziciju (bez gubljenja informacija) relacije u njene projekcije, na taj način što u jednoj projekciji ostavljamo primarni ključ i sve netranzitivno zavisne atribute, a u drugim projekcijama se realizuju funkcionalne zavisnosti koje su dovele do tranzitivnih zavisnosti:

FUDBALSKI_SUDIJA (SudijaID, ImeSudije, AdresaSud, PttBr)

ŠIFARNIK_MESTA (PttBr, Mesto)

Relacije FUDBALSKI_SUDIJA i ŠIFARNIK_MESTA su u 2 NF, jer relacije sadrže prost primarni ključ i one su i u 3 NF jer nepostoje tranzitivne funkcionalne zavisnosti. Pogledajmo i relaciju:

UTAKMICA (UtakmicaID, DatumUtakmice, Domaćin, Gost, Stadion, Rezultat, SudijaID)

Atribut Stadion je zavisan i od ključa relacije i od jednog neključnog atributa Domaćin. Stoga se data relacija svodi na:

UTAKMICA (UtakmicaID, DatumUtakmice, Domaćin, Gost, Rezultat, SudijaID)

KLUB (Domaćin, Stadion)

Kod objašnjavanja pojma tranzitivne zavisnosti u prethodnom poglavlju, pomenuli smo da relacija ZAPOSLENI sadrži tranzitivnu zavisnost, usled toga što je atribut NazivOdel funkcionalno zavistan i od ključa i od jednog neključnog atributa u relaciji. Njen početni oblik je bio:

ZAPOSLENI (ZaposleniID, ImeZaposl, DatumZaposlenja, Plata, OdelenjeID, NazivODel)

Datu relaciju dekomponujemo, tako što uklanjamo zavisnost koja je dovela do tranzitivne zavisnosti, tako da se relacija dekomponuje na sledeće dve relacije:

ZAPOSLENI (ZaposleniID, ImeZaposl, DatumZaposlenja, Plata, OdelenjeID)

ODELENJE (OdelenjeID, NazivOdel)

### Boyce-Coddova normalna forma

Tvorac relacionog modela je E.F. Codd, koji je dao originalne definicije 2NF i 3NF. Međutim, pokazalo se da ove definicije nisu dovoljno precizne, naročito u slučajevima kada relacija ima tzv. "preklapajuće" kandidate za ključ (dva ili više složenih kandidata za ključ koji imaju barem jedan zajednički atribut).

Relacija je u **Boyce-Codd-ovoj normalnoj formi** (BCNF) ako i samo ako su sve determinante u relaciji i kandidati za ključ. Determinanta relacije R je bilo koji atribut, prost ili složen, od koga neki drugi atribut u relaciji potpuno funkcionalno zavisi.

Da bi prikazali preklapajuće kandidate za ključ pretpostavimo da na nekom fakultetu jedan predmet može da predaje više nastavnika, ali svaki nastavnik može da predaje samo jedan predmet. Svaki student prijavljuje više predmeta, ali polaže samo kod jednog profesora za zadati predmet. Ova situacija se može opisati sledećom relacijom:

PRIJAVA (BrIndeksa, ImeNastavnika, PredmetID)

Data relacija nije u 2NF, jer postoji nepotpuna funkcionalna zavisnost:

ImeNastavnika → PredmetID

Da bi relaciju sveli na 2 NF, onda bi mogli drugačije da izaberemo primarni ključ:

PRIJAVA (BrIndeksa, PredmetID, ImeNastavnika)

Sada je relacija u 3NF, ali nije u BCNF zbog zavisnosti:

ImeNastavnika → PredmetID

Pritom ImeNastavnika nije kandidat za ključ. Zbog odstupanja od BCNF nastaju slične anomalije kao kod odstupanja od 3NF. Ne možemo evidentirati činjenicu da određeni nastavnik predaje određeni predmet, sve dok bar jedan student ne prijavi dati predmet. Takođe, veza nastavnika i predmeta je zavisana sa velikom redundantnošću, onoliko puta koliko ima studenata, što otežava ažuriranje. Ako svi studenti odustanu, briše se evidencija da određeni nastavnik predaje dati predmet. Rešenje problema je razbijanje relacije na dve čime se prekida nepoželjna funkcionalna zavisnost.

PRIJAVA (BrIndeksa, ImeNastavnika)

PREDAJE (ImeNastavnika, PredmetID)

Pogledajmo još jedan primer praćenja organizacije turneje koncerata po turističkim mestima.

PESMA (DatumNastupa, Pevač, Pesma)

MESTO(DatumNastupa, Mesto)

U okviru relacije PESMA, ključ bi mogao da bude i kombinacija atributa (DatumNastupa, Pesma), ali tada bi imali nepotpunu funkcionalnu zavisnost, usled zavisnosti Pesma → Pevač. Da bi se izbegla navedena nepotpuna funkcionalna zavisnost, biramo za ključ atribute DatumNastupa i Pevač jer Pesma zavisi i od Pevača i od Datuma. Dakle, sada nepostoje nepotpune i tranzitivne funkcionalne zavisnosti, ali ipak, relacija nije u BCNF jer determinanta Pesma nije kanditat za ključ. Pogledajmo sve determinante (D) relacije PESMA i obeležimo postojeće kandidate za ključ (KK):

| | | |
|---|---|---|
| DatumNastupa, Pevač → Pesma | (D) | (KK) |
| DatumNastupa, Pesma → Pevač | (D) | (KK) |
| Pesma → Pevač | (D) | |

Relacija PESMA se može dovesti u BCNF razbijanjem na dve manje relacije:

PESMA (DatumNastupa, Pesma)

PEVAČ (Pevač, Pesma)

## ČETVRTA NORMALNA FORMA

Relacija R je u **četvrtoj normalnoj formi** (4NF) ukoliko u njoj nisu date dve ili više nezavisne višeznačne činjenice. Codd objašnjavanja pojam višeznačne zavisnosti:

U relaciji R(A,B,C) postoji **višeznačna zavisnost** $A \rightarrow\rightarrow B$, ako za datu vrednost A, postoji skup više vrednosti B, a taj skup vrednosti ni na koji način ne zavisi od vrednosti atributa C. Atributi A, B i C mogu biti složeni. Primer višeznačne zavisnosti se prikazuje u tabeli IZVOZI.

Na tabeli IZVOZI uočavaju se dve nezavisne višeznačne zavisnosti: kompanija proizvodi više proizvoda i kompanija izvozi u više država, odnosno.

Kompanija ↠ Proizvod  Kompanija ↠ Država

IZVOZI

| Kompanija | Proizvod | Država |
|---|---|---|
| IBM | PC | Francuska |
| | | Italija |
| | Notebook | Velika Britanija |
| | | Nemačka |
| HP | LaserJet | Španija |
| | | Irska |
| | PC | Španija |
| | | Srbija i Crna Gora |

Država gde se izvozi zavisi samo od kompanije, ali ne i od proizvoda. Takođe, proizvod zavisi samo od kompanije, ali ne i od države gde se prodaje. Data relacija IZVOZI se nalazi u BCNF jer sadrži samo primarne atribute koji jedinstveno definišu n-torku. Ukoliko pretpostavimo da sve što neka kompanija proizvede to i izvozi u sve zemlje sa kojima posluje, tada postoji velika redundantnost. Ukoliko kompanija IBM uvede proizvodnju novog proizvoda, tada bi morali da upišemo onoliko n-torki, posebno za svaku državu sa kojima IBM posluje.

Dosadašnja pravila normalizacije nam nisu pomogla u eliminaciji redundantnosti u relaciji IZVOZI, usled toga što ona nije posledica funkcionalnih zavisnosti već višeznačnih zavisnosti. Relaciju IZVOZI treba da dekomponujemo na projekcije bez gubljenja informacija.

*Fagin* navodi pod kojim uslovima možemo dekomponovati relaciju bez gubljenja informacija: Relacija R (A,B,C) može se dekomponovati bez gubljenja informacija na projekcije R1(A,B) i R2 (A,C) ako i samo ako važi A →→ B (što uključuje i A→→C, jer ako u relaciji postoji višeznačna zavisnost A→→ B, tada postoji i višeznačna zavisnost A→→ C).

Pošto relacija IZVOZI nije u 4 NF, jer u njoj postoje dve nezavisne višeznačne činjenice, očigledno je da datu relaciju treba dekomponovati na projekcije:

RADI (Kompanija, Proizvod)

PRODAJE (Kompanija, Država)

### Peta normalna forma

Postoji još opštija forma zavisnosti, tzv "zavisnost spajanja" koja u sebi obuhvata višeznačne, pa samim tim i funkcionalne zavisnosti. U relaciji R(X,Y, ... Z) postoji zavisnost spajanja ako i samo ako relacija R rezultuje iz prirodnog spajanja njenih projekcija po X, Y, ..., Z, gde su X,Y, .. Z podskupovi atributa relacije R.

Relacija R je u **Petoj normalnoj formi** (5NF) ako i samo ako se svaka zavisnost spajanja može pripisati kandidatu za ključ.

Ukoliko posmatramo prethodno pomenutu relaciju:

FUDBALSKI_SUDIJA (SudijaID, ImeSudije, AdresaSud, PttBr, Mesto, UtakmicaID, DatumUtakmice, Domaćin, Gost, Stadion, Rezultat)

Primenom normalih formi, datu relaciju smo normalizovali dekomponujući je na sledeće četiri:

FUDBALSKI_SUDIJA1 (SudijaID, ImeSudije, AdresaSud, PttBr)

ŠIFARNIK_MESTA (PttBr, Mesto)

UTAKMICA (UtakmicaID, DatumUtakmice, Domaćin, Gost, Rezultat, SudijaID)

KLUB (Domaćin, Stadion)

Polazna relacija FUDBALSKI_SUDIJA se može rekonstruisati iz njenih projekcija prirodnim spajanjem relacija FUDBALSKI_SUDIJA1 i ŠIFARNIK_MESTA po atributu PttBr koja predstavlja primarni ključ u relaciji ŠIFARNIK_MESTA, a zatim prirodnim spajanjem tako dobijenog rezultata sa relacijom UTAKMICA po atributu SudijaID koji je primarni ključ u relaciji FUDBALSKI_SUDIJA. Na kraju tako dobijena relacija se spaja sa relacijom KLUB po atributu Domaćin koja predstavlja primarni ključ u relaciji KLUB.

S obzirom da se relacija FUDBALSKI_SUDIJA može rekonstruisati prirodnim spajanjem preko ključeva relacija, onda je ova relacija u 5NF.

### Normalna forma ključeva i domena (DK/NF)

R. Fagin je 1981. godine dao najopštiju definiciju normalne forme ključeva i domena i time pokazao da je relacija koja je u DK/NF ne prouzrokuje anomalije u održavanju.

Relacija je u **DK/NF (Domain Key/Normal Form)**:

- Ako su sve zavisnosti u njoj posledica definicije ključeva i domena.
- Ako svaki atribut daje informaciju isključivo o ključu i o celom ključu.

Drugim rečima, relacija R je u DK/NF ako i samo ako je svako ograničenje na R logična posledica ograničenja na domenu i ključu relacije R. Fagin tvrdi da nam nije potrebna funkcionalna zavisnost da bi definisali tabele. Dok god znamo koje ćemo ključeve koristiti i koju vrstu ograničenja imamo nad vrednostima koja će se nalaziti u kolonama (domeni), možemo dokazati da je tabela najmanje u 5NF i neće patiti od anomalija ubacivanja i brisanja.

Na primer, relacija PORUČENI_PROIZVOD (PorudžbinaID, ProizvodID, OpisProizvoda, PoručenaKoličina, JedCena) nije u DK/NF jer je definisano ograničenje da jedan Proizvod ima jedan OpisProizvoda, a to nije posledica definicije ključa ove relacije.

Ako je relacija u DK/NF, ona je sigurno u svim ostalim. Međutim, direktna primena ove definicije je slabo primenjiva, jer otkrivanje ograničenja u relacijama zahteva otkrivanje funkcionalnih, višeznačnih i zavisnosti spajanja, a to praktično znači primenu svih navedenih definicija.

## Sinhronizacija sistemskih modela

Modeli podataka i procesa predstavljaju različite poglede na isti sistem koje treba sinhronizovati kako bi se obzebedila konzistentnost i kompletnost celokupne specifikacije sistema. Kvalitet sinhornizacije podrazumeva da svaki objekat treba da ima najmanje jedno kreiranje (C – *create*), jedno iščitavanje (R – *read*), jedno menjanje (U – *update*) i jedno brisanje (D – *delete*) da bi sistem bio kompletan. CRUD matrice dokumentuju ove zahteve i sinhronizuju modele podataka i procesa i lokacija. Na 4.32 se prikazuje primer podatak-proces CRUD matrice.

*Podatak-Proces-CRUD matrica*

| Objekat<br>Atribut | Proces aplikacije klijenta | Proces promene adrese klijenta | Proces nove porudžbine klijenta | Proces poništavanja porudžbine klijenta | Proces promene porudžbine klijenta | Proces promene cene proizvoda |
|---|---|---|---|---|---|---|
| Klijent | C | | R | R | R | |
| .Šifra klijenta | C | | R | R | R | |
| .Naziv klijenta | C | U | R | | R | |
| .Adresa klijenta | C | U | RU | | RU | |
| .Iznos kredita klijenta | | | R | | R | |
| Porudžbina | | | C | D | RU | |
| .Šifra porudžbine | | | C | | R | |
| .Datum podrudžbine | | | C | | U | |
| .Iznos porudžbine | | | C | | U | |
| Naručeni proizvodi | | | C | D | CRUD | |
| .Količina porudžbine | | | C | | CRUD | |
| .Jedinična cena naručenog artikla | | | C | | CRUD | |
| Proizvod | | | R | R | R | RU |
| .Šifra proizvoda | | | R | R | R | |
| .Naziv proizvoda | | | R | | R | |
| .Opis proizvoda | | | R | | R | |
| .Jedinica mere proizvoda | | | R | | R | RU |
| .Jedinična cena proizvoda | | | R | | R | U |

C - kreirati (create) R - iščitavati (read) U - promeniti (update) D – obrisati (delete)

Slika 4.32. Podatak-proces CRUD matrica

Pored sinhronizacije modela procesa i podataka, neophodno je identifikovati i koji podaci će se nalaziti na određenim lokacijama. Da bi identifikovali podatke i prava pristupa na svakoj od lokacija, potrebno je pronaći odgovore na sledeća pitanja:

- Koji podskup objekata i atributa su neophodni za obavljanje posla na svakoj od lokacija?
- Koji nivo pristupa se zahteva?
- Mogu li se na određenoj lokaciji kreirati, iščitavati, menjati ili brisati instance objekta?

Sistemska analiza predlaže definisanje ovih logičkih zahteva u formi podatak-lokacija CRUD matrica. Podatak-lokacija CRUD matrica (*Data-to-location*-CRUD *matrix*) je tabela u kojoj redovi pokazuju objekte i moguće atribute, kolone pokazuju lokacije, a presečna polja pokazuju nivo pristupa gde je C-kreiranje, R-iščitavanje, U-menjanje i D-brisanje ili deaktiviranje. Na slici 4.33 je prikazan primer podatak-lokacija-CRUD matrice, gde se uočava da će svako prodajno odelenje pristupati samo onim klijentima koji se nalaze u njihovom regionu.

| Objekat<br>Atribut | Klijenti | Beograd | Marketing | Skladište | Prodaja | Podgorica | Skladište |
|---|---|---|---|---|---|---|---|
| Klijent | INDV | | | | ALL | | SS |
| Šifra klijenta | R | | | R | CRUD | | CRUD |
| Naziv klijenta | RU | | | R | CRUD | | CRUD |
| Adresa klijenta | RU | | | R | CRUD | | CRUD |
| Iznos kredita klijenta | X | | | | R | | R |
| Porudžbina | INDV | | ALL | SS | ALL | | SS |
| Šifra porudžbine | RD | | R | R | CRUD | | CRUD |
| Datum podrudžbine | RD | | R | R | CRUD | | CRUD |
| Iznos porudžbine | RD | | R | | CRUD | | CRUD |
| Naručeni proizvodi | INDV | | ALL | SS | ALL | | SS |
| Količina porudžbine | UD | | R | R | CRUD | | CRUD |
| Jedinična cena naručenog artikla | UD | | R | | CRUD | | CRUD |
| Proizvod | ALL | | ALL | ALL | ALL | | ALL |
| Šifra proizvoda | R | | CRUD | R | R | | R |
| Naziv proizvoda | R | | CRUD | R | R | | R |
| Opis proizvoda | R | | CRUD | R | R | | R |
| Jedinica mere proiz. | R | | CRUD | R | R | | R |
| Jedinična cena proiz. | R | | | | R | | R |

INDV – individualno ALL – svi SS – podskup X - nema pristupa
C - kreirati (create) R - iščitavati (read) U - promeniti (update) D – obrisati (delete)

Slika 4.33. Podatak-Lokacija-CRUD matrica

## 4.5. Objektno-orijentisana analiza i modelovanje

**Objektno-orijentisana analiza (*Object-Oriented Analysis* - OOA)** je tehnika koja je uspela da eliminiše veštački razdvojene podatake od procesa. Umesto podataka i procesa koji su kreirali, čitali, ažurirali i brisali te podatke, oni su zajedno integrisani u objekte. Jedini način da se kreiraju, čitaju, ažuriraju ili brišu podaci objekta (*properties*) jeste putem ugrađenih procesa koji se nazivaju metode (*methods*). Programski jezici kao što su C++, Visual Basic, C#, Java i mnogi drugi se baziraju na ovoj objektnoj tehnologiji.

Kada se kaže da je nešto objektno-orijentisano, to u osnovi znači da se na problem gleda preko objekata koji su uključeni u taj problem. Objektno-orijentisani koncepti se koriste u mnogim profesijama. Na primer, kada projektuje kancelarije arhitekta razmišlja o radnim prostorima, temeljima, konstrukciji i vodovodu. To su objekti iz stvarnog sveta. Arhitekta ne razmišlja o procesu nalivanja temelja betonom, zakucavanju eksera ili spajanju vodovodnih cevi. To su procesi nižih nivoa koji imaju svoju važnost, ali se ne uzimaju u obzir kod projektovanja poslovne zgrade. Sa konceptima objektnog modeliranja donekle ste se već upoznali u delu modeliranje podataka, a u narednom tekstu će se, radi celine, ponoviti neki i objasniti novi koncepti.

Objektno-orijentisani pristup razvoja sistema je zasnovan na konceptu objekata koji postoje unutar sistemskog okruženja. **Objekti** su definisani kao stvari, ljudi, događaji, sistemi, strukture ili uređaji. Efektivan metod za identifikovanje objekata je da se prouče scenarija upotrebe (*usage scenarios*, koja će kasnije biti detaljnije objašnjena) i identifikuju imenice i pridruženi glagoli koji će se prevesti u objekte i pridružena ponašanja. Na primer, razmotrimo sledeći slučaj korišćenja: Zaposleni kreira ugovor sa klijentom. Mogu se identifikovati sledeći poslovni objekti: zaposleni, ugovor i klijent.

Drugi isto tako važan koncept u objektnom modelovanju jeste kategorizacija objekta u klase. **Klasa** je skup objekata koji dele zajedničke atribute i ponašanja. Ljudi rado klasifikuju stvari, ne bi li našli sličnosti među njima i prema tome ih grupisali. Stvari sa sličnim svojstvima i ponašanjima se grupišu zajedno. Na primer, Petar Petrović i Marija Marjanović imaju svoja imena, adrese i zanimanja i pošto oboje rade u istom preduzeću, možemo ih svrstati u klasu zaposlenih radnika. Petar je programer i ima listu programskih jezika koje koristi, pa ga možemo svrstati u klasu sa ostalim programerima, a Marija obavlja komercijalne poslove pa ćemo je svrstati u klasu sa drugim komercijalistma iz preduzeća. Da bi neko bio u klasi zaposlenih radnika mora imati ime, adresu, zanimanje i ponašanje «radi». Pošto Petar i Marija ispunjavaju ove kriterijume, oni su objekti koji pripadaju ovoj klasi. Određeni objekat koji pripada klasi naziva se primerak ili instanca (*instance*) klase. Svaki objekat koji je instanca klase ima atribute i ponašanja definisana u klasi.

**Atributi** ili svojstva jednog objekta su definicije vrednosti koje objekat poseduje. Da bi se identifikovali atributi objekta, treba razmotriti prideve ili imenice (koje nisu postali objekti) unutar scenarija upotrebe. Svaka instanca

objekta sadrži sopstveni set vrednosti. Na primer, ukoliko posmatramo slučaj korišćenja koji glasi: Klijent sadrži broj računa, naziv i adresu; tada se uočava da objekat Klijent sadrži atribute broj računa, naziv i adresu sa konkretnim vrednostima npr. 1007, ComTrade, Savski nasip 7.

Objekti takođe mogu i da izvršavaju neke radnje. Stvari koje objekat radi se nazivaju ponašanja (*behaviors*) ili servisi. Na primer, vrata se mogu otvarati, zatvarati, zaključavati ili otključavati. Porudžbina se može modifikovati, obrisati, dodati itd. **Servisi** su specifična ponašanja (metode) koje poslovni objekat mora da izvršava.

Ponašanja implementiraju poslovna pravila, obrađuju podatke i pristupaju informacijama. Odnose se na operacije, funkcije ili transformacije koje izvršava objekat. Da bi se uočila ponašanja treba identifikovati akcije koje objekat mora da izvršava u scenarijima upotrebe. Zatim se analizira šta objekat mora da radi i koje podatke mora da održava. Ovakve informacije vode ka neopohodnoj funkcionalnosti ponašanja. Neki od primera akcija koje objekat može da izvršava su: izračunavanje ukupnih iznosa ili određivanje troškova transporta. Ponašanja treba dodeliti odgovarajućim objektima.

Jedna od osnovnih prednosti korišćenja objekata je u tome što objekat ne mora da otkriva sve svoje atribute i ponašanja. Objekat treba da otkrije interfejse preko kojih se može komunicirati sa njim. Druge detalje, koji nisu važni za korišćenje objekata treba sakriti od ostalih objekata. Ovaj objektno-orijentisani koncept se zove enkapsulacija (*encapsulation*). Na primer, objekat koji izračunava kvadrat nekog broja mora obezbediti interfejs za dobijanje rezultata. Međutim, unutrašnji atributi i algoritmi koji se koriste za izračunavaje kvadrata ne treba da budu dostupni pozivajućem objektu.

Svaki atribut ili metode unutar neke klase mogu biti:

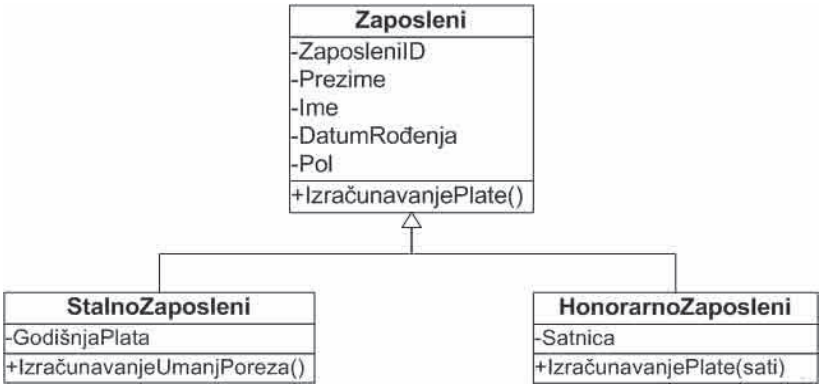
- Javni (*public*) - označava da su atribut ili metoda vidljivi za sve klase u programu (označava se sa „+“);
- Privatni (*private*) – atribut ili metoda su vidljivi samo unutar svoje klase (označava se sa „-“);
- Zaštićeni (*protected*) – atribut ili metoda su vidljivi samo za klase naslednice (oznaka “#”);

Objektno-orijentisani koncept nasleđivanje (*inheritance*) je proces kojim jedan objekat nasleđuje svojstva drugog objekta. Nasleđivanje je važno jer

podržava koncepciju hijerarhijske klasifikacije (tj. odozgo nadole). Ukoliko postoji nasleđivanje, za objekat treba definisati samo one karakteristike koje ga čine jedinstvenim u klasi, jer opšta svojstva objekat nasleđuje od roditelja. Na taj način, nasleđivanje predstavlja mehanizam koji omogućuje da jedan objekat bude poseban slučaj nekog opštijeg objekta. Na primer, ako biste želeli da životinje opišete apstraktno, rekli biste da one imaju izvesne osobine, npr. veličinu, inteligenciju i vrstu skeletnog sistema. Životinje imaju i određene aspekte ponašanja; one jedu, dišu i spavaju. Ovakav opis osobina i ponašanja predstavlja definiciju klase životinja. Ako želite da definišete određeniju klasu životinja, npr. sisare, oni bi takođe morali imati određenije osobine, npr. mlečne žlezde i tip zuba. Kažemo da su sisari potklasa (*subclass*) životinja, a životinje su sisarima natklasa (*superclass*). Pošto su sisari samo uže definisane životinje, oni nasleđuju sve osobine životinja. Potklasa koja se nalazi duboko u hijerarhiji nasleđuje osobine svih svojih predaka u hijerarhiji klase.

Objektno-orijentisani koncept polimorfizam (*polymorphism*) je reč grčkog porekla koji znači "mnogo oblika" i predstavlja koncept koji se često objašnjava frazom "jedan način pristupa, više metoda". Polimorfizam podrazumeva mogućnost pravljenja opšteg načina pristupa za rad sa grupom srodnih aktivnosti. Na primer, ukoliko posmatramo klasu Oblik, sa ponašanjem Crtaj. Kada nekom kažete da nacrta oblik, pitanje koje sledi je: Koji oblik? Pretpostavimo da imate niz od tri oblika i to Krug, Kvadrat i Zvezda. Iako sa svima postupate kao sa objektima nasleđenim iz klase Oblik i svima šaljete poruku Crtaj, konačni ishod je drugačiji, jer Krug, Kvadrat i Zvezda obezbeđuju konkretnu realizaciju. Svaka klasa je sposobna da drugačije odgovori na isti metod Crtaj.

Polimorfizam omogućava da opšta klasa definiše metode koje će biti zajedničke za sve klase koje su iz nje izvedene, ostavljajući potklasama slobodu da definišu specifične varijante nekih ili svih pomenutih metoda (tzv. redefinisanje metoda).




Slika 4.34. Generalizacija sa redefinisanim metodama

Objektno-orijentisana analiza je naročito postala popularna pojavom UML (*Unified Modeling Language*) jezika koji obezbeđuje i grafičku sintaksu za objektne modele. UML definiše nekoliko različitih tipova dijagrama koji zajedno modeluju jedan informacioni sistem ili aplikaciju u terminima objekata. Pomoću UML-a softverski sistem se može opisati sa različitih aspekata, kao što su:

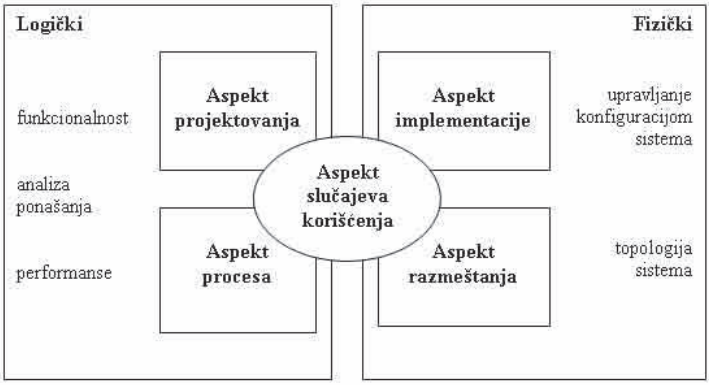
- specifikacija zahteva preko modela slučajeva korišćenja,
- specifikacija strukture sistema preko dijagrama klasa,
- specifikacija dinamike sistema preko dijagrama kolaboracije i dijagrama promene stanje,
- specifikacija implementacije preko dijagrama komponenti,
- specifikacija rasporeda komponenti u fizičkoj strukturi preko dijagrama raspoređivanja i sl.

### 4.5.1. Jedinstveni jezik modelovanja - UML

*Object Management Group* (OMG) je u Novembru 1997. godine usvojila kao oficijelnu notaciju objektno-orijentisanog modelovanja - jedinstveni jezik modelovanja, kraće UML (*Unified Modeling Language*). Cilj OMG je bio da se kreira standardizovana objektno-orijentisana arhitektura za distribuirane aplikacije koje podržavaju distribuirane objekte, odnosno cilj im je bio da komponente objektno-orijentisanog softvera imaju osobine ponovne upotrebljivosti, portabilnosti i interoperativnosti u jednom heterogenom okruženju. Konsenzus oko glavnih ideja i unifikacija glavnih

metoda objektno-orijentisanog modelovanja kao što su Booch-ova metoda particionisanja sistema u podsisteme, Rumbaugh-ov koncept klase i asocijacije i Jacobson-ov model korisničkih funkcija za definisanje zahteva sistema dovele su do pojave UML-a.

UML je metod treće generacije koji služi za specifikaciju, vizuelizaciju, konstrukciju i dokumentaciju razvoja sistema. Namena UML-a je povećanje produktivnosti, skraćenje vremena razvoja i poboljšanje kvaliteta informacionih sistema. UML može da se koristi u različitim fazama razvoja, od specifikacije zahteva do testiranja završenih, gotovih sistema.




Slika 4.35. Aspekti modela u UML-u

UML obezbeđuje brojne grafičke alate koji se mogu koristiti za vizuelizaciju i razumevanje sistema sa različitih tačaka gledišta. Dijagrami se mogu korisiti kako bi se predstavili različiti pogledi na sistem. Zajedno, višestruki pogledi na sistem predstavljaju model sistema. Modeli ili pogledi se koriste kako bi se opisala kompleksnost softverskih sistema. Različiti UML pogledi opisuju nekoliko aspekata softverskog sistema. Aspekti koji se koriste su:

- **Korisnički pogled** – opisuje se ponašanje sistema sa tačke gledišta korisnika. Predstavljaju se ciljevi sistema iz perspektive korisnika i njihovih sistemskih zahteva. Ovaj pogled predstavlja deo sistema sa kojima su korisnici u interakciji. Statički opis ovoga aspekta daje se preko dijagrama slučajeva korišćenja, a dinamički, prvenstveno tekstualno, a zatim i preko dijagrama interakcija, dijagrama promene stanja ili dijagrama aktivnosti. Ovaj pogled se još naziva aspekt slučajeva korišćenja (*use case*) i jednom rečju predstavlja funkcionalnu specifikaciju sistema.

- **Aspekt projektovanja** – predstavlja realizaciju sistema u „objektnom prostoru stanja“. Statički opis ovoga aspekta daje se preko dijagrama klasa i dijagrama objekata. Dinamički opis se daje preko dijagrama interakcija, dijagrama promene stanja i dijagrama aktivnosti.
- **Aspekt procesa** – predstavlja dinamičko stanje sistema. Opisuje način odvijanja procesa u sistemu, „niti“ procesa, konkurentnosti i sinhronizaciju. Koriste se isti dijagrami kao i u aspektu projektovanja i za statički i za dinamički opis, a najviše dijagrami aktivnosti.
- **Aspekt implementacije** – predstavlja strukturu logičkih elemenata u sistemu. Predstavlja komponente i fajlove preko kojih se sistem fizički ostvaruje. Statički opis ovoga aspekta daje se preko dijagrama komponenti. Dinamički opis se daje preko dijagrama interakcija, dijiagrama promene stanja i dijagrama aktivnosti.
- **Aspekt razmeštanja** – predstavlja raspodelu fizičkih elemenata sistema. Okruženje sistema određuje funkcionalnost sistema iz perspektive korisnika. Aspekt razmeštanja predstavlja topologiju sistema, računarsko-komunikacionu mrežu. Pokazuje se kako su u ovoj mreži razmeštene komponente koje predstavjlaju fizičku realizaciju sistema. Dijagrami razmeštaja se koriste za statički opis. Dinamički opis se daje preko dijagrama interakcija, dijagrama promene stanja i dijagrama aktivnosti. Aspekt razmeštanja se još naziva i aspekt uvođenja (*deployment*).

Svaki pogled na sistem koristi više vrsta dijagrama za prikazivanje svog sadržaja, a sa druge strane, jedna vrsta dijagrama može da se koristi za prikazivanje delova modela u raznim pogledima na sistem. Na tabeli 4.7 su grupisani UML dijagrami prema opisu strukture sistema.

Tabela 4.7. UML dijagrami prema strukturi sistema

| UML dijagrami za opis statičke strukture sistema | UML dijagrami za opis dinamike sistema |
|---|---|
| 1. Dijagrami klasa | 1. Dijagrami slučajeva korišćenja |
| 2. Dijagrami objekata | 2. Dijagrami sekvenci |
| 3. Dijagrami komponenti | 3. Dijagrami kolaboracije |
| 4. Dijagrami razmeštaja | 4. Dijagrami promene stanja |
| | 5. Dijagrami aktivnosti |

Detaljniji prikaz gupisanja UML dijagrama:

I. Dijagrami modelovanja slučajeva korišćenja:

1. Dijagrami slučajeva korišćenja (*use-case diagrams*) - prikazuju željenu funkcionalnost sistema.

II. Dijagrami statičke strukture:

2. Dijagrami klase (*class diagrams*) - opisuju različite klase i njihove veze (asocijacije).
3. Dijagrami objekta (*object diagrams*) - opisuju raziličite objekte u sistemu i njihove međusobne veze.

III. Dijagrami interakcije:

4. Dijagrami sekvenci (*sequence diagrams*) - opisuju interakcije između klasa prikazivanjem redosleda poruka koje se razmenjuju između klasa.
5. Dijagrami kolaboracije (*collaboration diagrams*) - predstavlju skup klasa i njihovih poslatih i prijemnih poruka.

IV. Dijagrami stanja:

6. Dijagrami stanja (*statechart diagrams*) - predstavljaju algoritme različitih stanja jednog objekta koji se događaju pri uticaju različitih spoljnih događaja.
7. Dijagrami aktivnosti (*activity diagrams*) - predstavljaju drugi način pogleda na stanje i uključuje i sekvence aktivnosti. Pogodan kod modelovanja konkurentnih procesa kako bi se videla neophodna sinhronizacija između klasa i zadataka.

V. Dijagrami implementacije:

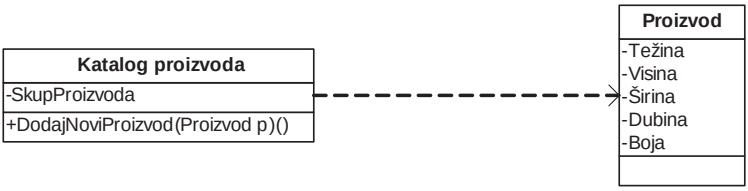
8. Dijagrami komponenti (*component diagrams*) - prikazuju različite komponente sistema i njihove veze (npr. *source code, object code i execution code*).
9. Dijagrami razmeštaja (*deployment diagrams*) - predstavljaju spajanja softverskih komponenti sa čvorovima fizičke implementacije sistema.

**Dijagrami klasa** predstavljaju kičmeni stub za skoro sve objektno-orijentisane metode. Oni prikazuju statičku strukturu sistemu. Klase u objektnom modelovanju, prema UML notaciji se prikazuju u obliku pravougaonika (Slika 4.36). Pravougaonik je podeljen u tri dela. U prvom delu se prikazuje naslov klase. Drugi, središnji deo sadrži imena zajedničkih atributa. Treći, donji deo sadrži zajednička ponašanja (metode).

Objekti ili klase su povezani preko relacija (veza). UML definiše četiri tipa relacija:

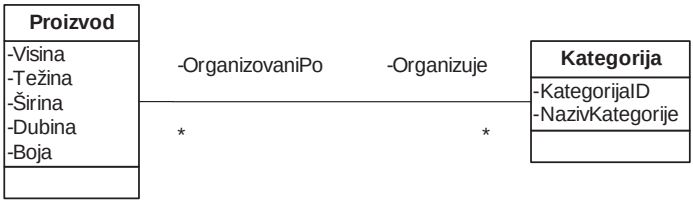
- **Zavisnost** – relacija između dva objekata u kojoj promena na jednom objektu utiče na ponašanje drugog objekta. Relaciju zavisnosti treba upotrebiti onda kada želimo da pokažemo da jedan objekat koristi drugi.

  Zavisnost između dva objekta se predstavlja usmerenom isprekidanom linijom (Slika 4.36).



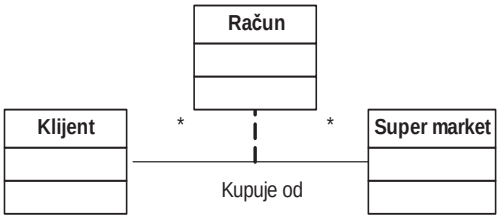

Slika 4.36. Primer relacije zavisnosti

- **Asocijacija** – kada su dve klase konceptualno vezane jedna za drugu. Na UML dijagramima asocijacija se predstavlja punom linijom između dve klase, sa nazivom asocijacije na vrhu (Slika 4.37).



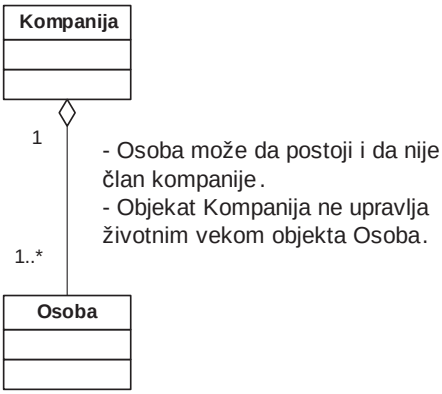

Slika 4.37. Primeri relacije asocijacije

Asocijacija kao tip veze može biti kompleksnija uključivanjem više od dve klase u relaciju. Ponekad veza između dve klase mora biti ispraćena nekim pravilom. To pravilo se najčešće stavlja dodavanjem ograničenja pored linije koja predstavlja vezu klasa (Slika 4.38).



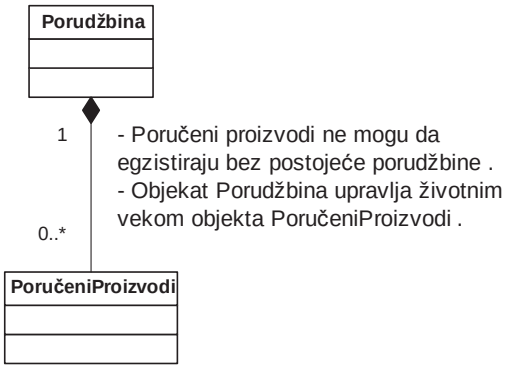

Slika 4.38. Primer složenije asocijacije

- Postoje dva tipa aoscijacije: agregacija i kompozicija. **Agregacija** predstavlja vezu između celine i njenih delova. Na primer, Osoba je zaposlena u Kompaniji (Slika 4.39). Agregacija je veza između dva jaka objekta, gde će objekat B koji je dinamički atribut objekta A ostati vidljiv i kada objekat A prestane da postoji. Agregacija je predstavljena punom linijom sa praznim rombom na kraju linije koja se povezuje sa celinom.



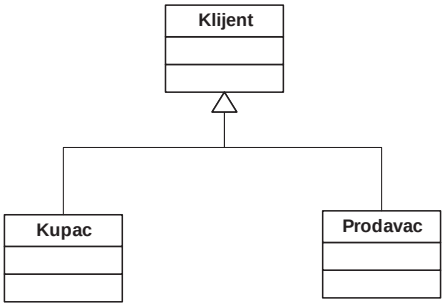

Slika 4.39. Relacija agregacije

- Ukoliko celina upravlja životnim vekom delova onda se takva veza naziva kompozicija (Slika 4.40). **Kompozicija** odgovara vezi između jakog i slabog objekta u modelu objekti veze. Objekat B koji je statički atribut objekta A je vidljiv sve dok objekat A postoji, jer je B egzistencijalno zavistan od A. Kompozicija je predstavljena punom linijom sa popunjenim rombom na kraju linije koja je povezana sa celinom.



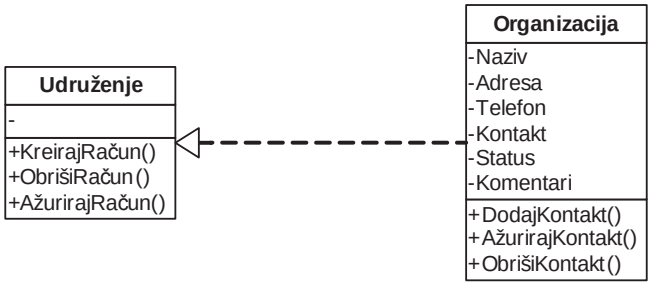

Slika 4.40. Relacija kompozicije

- **Generalizacija** – relacija između opštijeg tipa (roditelja) i određenijih tipova (dete). Generalizacija se prikazuje kao puna linija sa praznom strelicom koja ukazuje na roditelja (Slika 4.41).




Slika 4.41. Relacija generalizacije

- **Realizacija** – relacija između klasa, gde jedna apstraktna klasa određuje zadatke koje druga klasa mora da iznese. Generalizacija je predstavljena isprekidanom linijom i praznom strelicom usmerene na roditelja (Slika 4.42).




Slika 4.42. Relacija realizacije

Specijalan vid asocijacije je višestrukost ili multiplikativnost (*multiplicity)* koja pokazuje broj objekata jedne klase koji su u relaciji sa određenim brojem elemenata u drugoj klasi. Primer UML notacije višestrukosti je prikazana u tabeli 4.8.

Tabela 4.8. Primer UML notacija višestrukosti

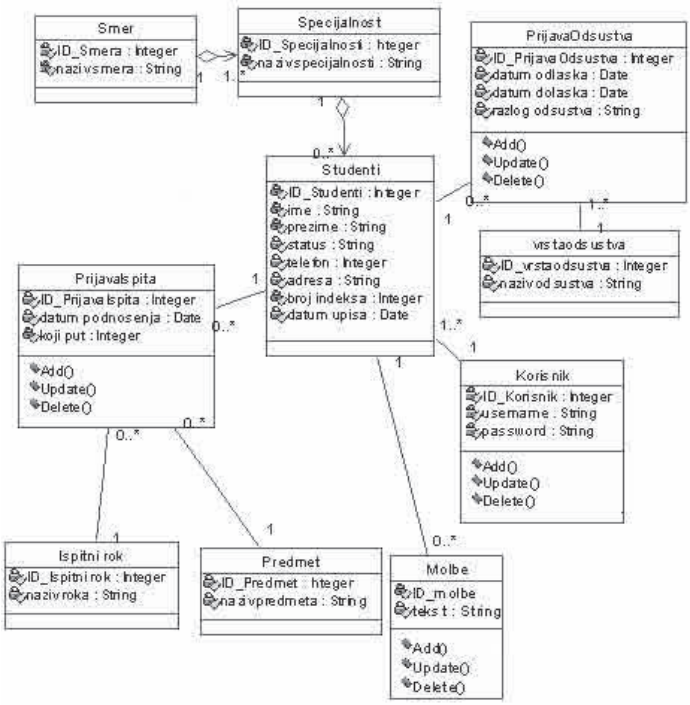
| Višestrukost | UML notacija višestrukosti | Grafička notacija | Značenje |
|---|---|---|---|
| **Tačno 1** | **1**<br>ili<br>ostaviti blanko | Zaposleni — Radi za 1 — Odelenje<br>Zaposleni — Radi za — Odelenje | Jedan radnik radi za jedno i samo jedno odelenje |
| **Nula ili jedan** | **0..1** | Zaposleni — Ima 0..1 — Suprug/ supruga | Zaposleni ili ima ili nema supruga/u |
| **Nula ili više** | **0..***<br>ili<br>* | Zaposleni — Vrši 0..* — Plaćanje<br>Zaposleni — Vrši * — Plaćanje | Zaposleni može da izvrši nijedno ili više plaćanja |
| **Jedan ili više** | **1..*** | Univerzitet — Nudi 1..* — Kurs | Univerzitet nudi jedan ili više kurseva |
| **Određeni opseg** | **7..9** | Tim — planira 7..9 — Igre | Tim ima 7, 8 ili 9 planiranih igara |

Relacije se mogu identifikovati iz slučajeva korišćenja tako što se iz scenarija izvlače informacije koje opisuju fizičke lokacije, usmerene akcije, komunikacije, posedovanje ili ukazuju na ispunjene uslove. Relacije prikazuju kako akcije objekata ili sistema utiču na druge objekte ili sisteme.

Dijagram klasa se koristi da bi se:

- predstavio osnovni rečnik sistema koji definiše pojmove koji se u tom sistemu koriste;
- opisala struktura neke kolaboracije;
- predstavila logička šema baze podataka.

Opis sistema obično sadrži više dijagrama klasa. Primer dijagrama klasa prikazan je na slici 4.43.

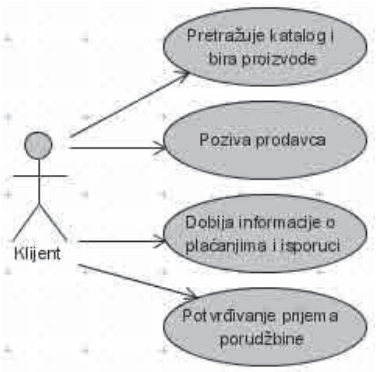



Slika 4.43. Primer dijagrama klasa

**Dijagram objekata** je varijanta dijagama klasa i koristi skoro istu notaciju, a razlika je u tome što, umesto klase, prikazuje jednu ili više instanci klase sa konkretnim podacima. Dijagrami objekata modeliraju stanje sistema u jednom trenutku vremena, tj. daju zamrznutu sliku sistema u jednom trenutku vremena. Dijagrami objekata daju statičku strukturu za opis dinamičkog ponašanja sistema preko dijagrama interakcije. Oni su

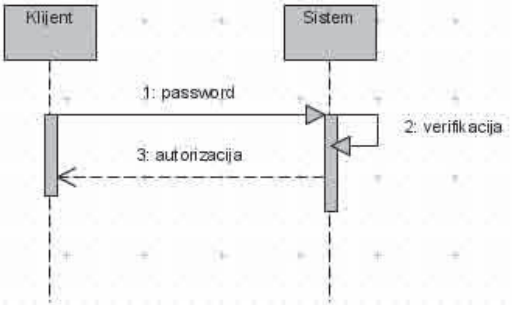
pojavljivanje odgovarajućeg dijagrama klasa i statički deo dijagrama interakcije. Predstavljaju skup objekata i pojavljivanja veza, a mogu da sadrže komentare i ograničenja i da budu podeljeni u pakete i podsisteme.

**Dijagrami slučajeva korišćenja** modeluju funkcionalnost sistema prikazujući učesnike (korisnike sistema) i njihove veze sa slučajevima korišćenja. Slučajevi korišćenja su opisi pojedinačnih funkcija viđen iz perspektive korisnika. Učesnik je bilo koji spoljni entitet, čovek, drugi sistem ili neki hardverski uređaj, koji treba da komunicira sa sistemom koji se modelira. Sam proces kreiranja modela čini: definisanje sistema, uočavanje učesnika i slučajeva korišćenja, opisivanje slučajeva korišćenja, definisanje veza između funkcija i na kraju validacija modela.



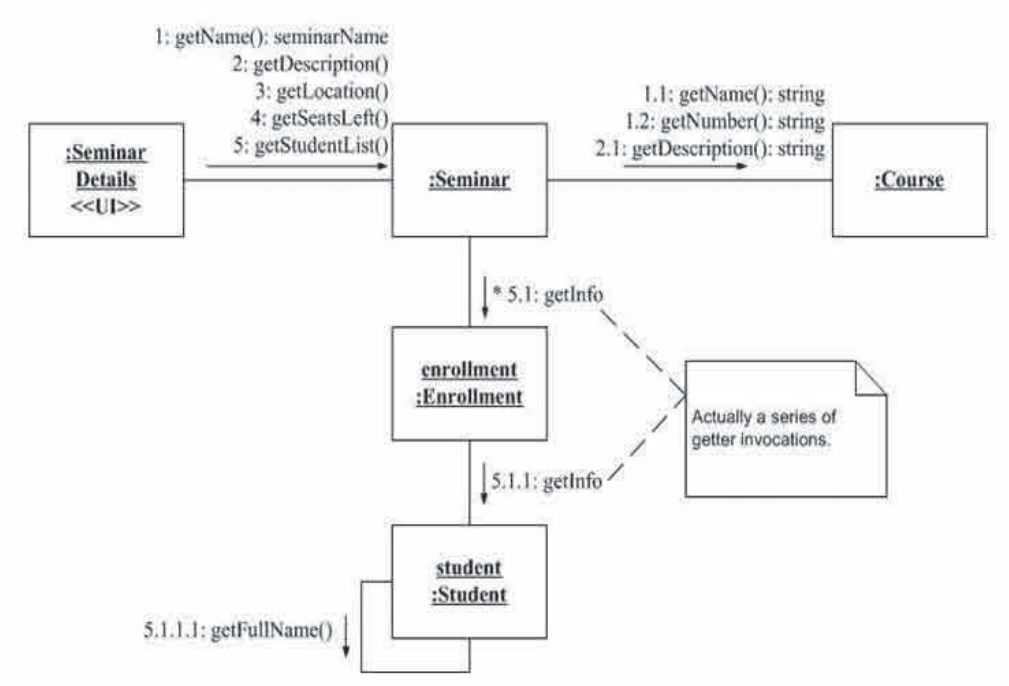

Slika 4.44. Primer dijagrama slučajeva korišćenja

**Dijagram sekvenci** prikazuje dinamičke veze objekata, redosled poruka koje objekti razmenjuju, događaje koji se dešavaju u određenom trenutku rada. Dijagram se sastoji od više objekata pri čemu se proticanje vremena prikazuje odozgo na dole, poruke horizontalnim usmerenim linijama, dok se dodatni komentari pišu na marginama dijagrama.




Slika 4.45. Primer dijagrama sekvenci

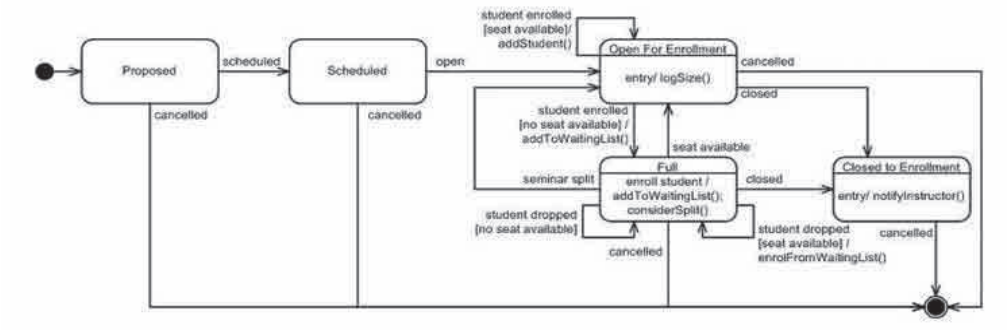
**Dijagrami komunikacije ili saradnje (kolaboracije)** opisuju statičku strukturu i dinamičko ponašanje sistema, pri čemu se prikazuju veze među objektima, odnosno konekst u kom se objekti koriste. Razlika u upotrebi dijagrama sekvenci i ovog je u aspektu koji naglašava vreme tj. redosled (koristi se sekvencni dijagram) ili kontekst (koristi se dijagram saradnje). Dijagramu se mogu dodati i uslovi, iteracije, povratne vrednosti itd.




Slika 4.46. Primer dijagrama komunikacije [1]

**Dijagram stanja** obično predstavlja dodatak opisu klase jer pokazuje sva moguća stanja u kojima se objekti jedne klase mogu naći i događaje koji prouzrokuju promene stanja. Događaj može biti poruka koju šalje drugi objekat, ispunjenje nekog uslova, proteklo vreme i slično. Promena stanja se naziva prelaz i može mu se pridružiti akcija koja specificira šta treba da se uradi pri tom prelazu. Dijagram stanja predstavlja stanja u kojima objekat može da bude tokom tranzicije iz jednog u drugo stanje i prikazuje početnu tačku i krajnju tačku sekvence koja menja stanje. Bitno je napomenuti da dijagram stanja prati stanje samo jednog objekta. Stanja se predstavljaju sa zaobljenim pravougaonikom, dok se simbol transakcije stanja predstavlja punom linijom sa strelicom. Punim krugom se predstavlja početna tačka, a krajnja sa krugom sa tačkom u sredini. Sam

zaobljeni pravougaonik se stoji od tri dela. U gornjem se nalazi naziv stanja (ovaj podatak obavezno mora biti unet), u sredini se nalazi podatak o stanju promenjive, dok je u donjem delu informacija o aktivnosti. Aktivnosti mogu imati tri opisa: *entry*- šta se dešava kada sistem ulazi u stanje, *exit*- šta se dešava kada sistem napusti stanje, *do*- šta se dešava sa sistemom kada je u stanju.



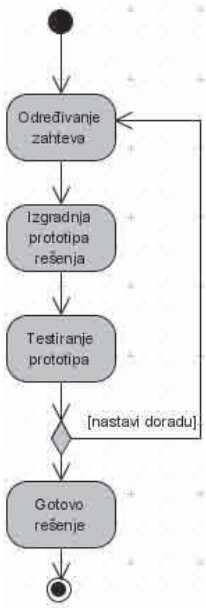

Slika 4.47: Primer dijagrama stanja [1]

Nekada su stanja u kojima se nalazi sistem previše komplikovana, te se tada predstavljaju podstanjima u jednom stanju.

**Dijagrami aktivnosti** prikazuju redosled toka aktivnosti i obično podsećaju na slike algoritama. On se sastoji od koraka (zvanih aktivnosti), tačaka odluke i grananja. Veoma je koristan za pokazivanje šta se dešava u poslovnim procesima ili operacijama. Pre svega dijagrami aktivnosti se prave da bi se lakše shvatilo šta se dešava tokom procesa. Oni na neki način predstavljaju nadogradnju dijagrama stanja. Dijagrami stanja pokazuju stanja objekata i prezentuju aktivnosti strelicama. Dijagrami aktivnosti stavljaju akcenat na aktivnostima. U UML notaciji svaka aktivnost se predstavlja elipsom. Tranzicija sa jedne na drugu aktivnost se predstavlja strelicom, dok se na vrhu svakog dijagrama aktivnosti nalazi pun krug (koji označava početak), a u dnu prazan krug sa tačkom koji označava kraj. Element grananja se prikazuje u obliku dijamanta, do koga dolazi strelica koja pokreće grananje, a iz njega izlaze dve grane, u zavisnosti da li je uslov ispunjen ili ne. Obično se na linijama koje izlaze iz dijamanta ispisuje vrednost uslova koji treba da ispuni grananje da bi tok podataka krenuo na tu stranu

Prilikom projektovanja aktivnosti veoma se često dešava da se neka aktivnost podeli na dva odvojena dela koja se izvršavaju u isto vreme

(konkurentno), da bi se na kraju izvršenja obe slile ponovo u jednu. To deljenje aktivnosti na dva dela se obeležava sa zadebljanom linijom u koju ulazi strelica iz početne aktivnosti. Iz zadebljane linije izlaze dve strelice koje predstavljaju novonastale aktivnosti i opet se završavaju u zadebljanoj liniji iz koje izlazi jedna strelica, označavajući da je konkurentni rad završen.



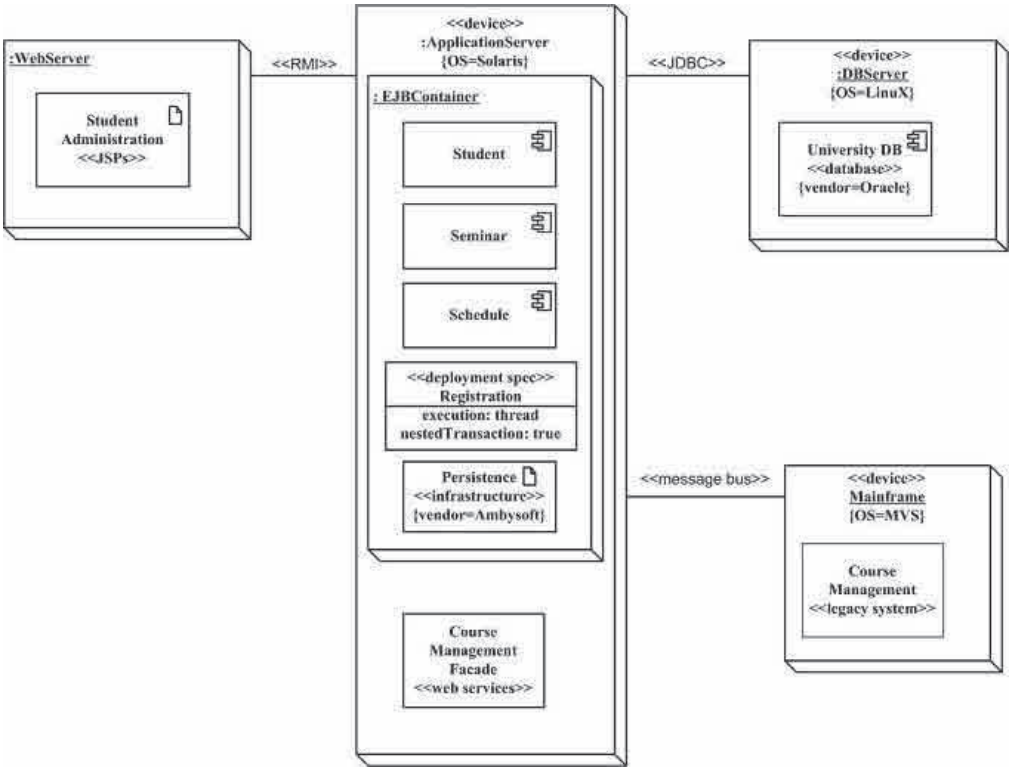

Slika 4.48. Primer dijagrama aktivnosti

**Dijagram komponenti** prikazuje fizičku strukturu kôda tj. komponenti kôda, pri čemu komponenta može biti izvorni ili izvršni program. Komponente mogu da se grupišu u pakete, a prikazane zavisnosti među njima omogućavaju laku analizu uticaja jedne komponente na ostale komponente.

**Dijagram razmeštaja** prikazuje fizičku arhitekturu hardvera i softvera u sistemu. Prikazuje se povezanost računara i uređaja, tj. čvorova sa tipovima veza. U pojedinim čvorovima se navode softverske komponente koje se u njima izvršavaju, kao i zavisnost među komponentama. Primer dijagrama razmeštaja je prikazan na slici 4.49.

### 4.5.2. Proces objektnog modelovanja

Kod objektno-orijentisane analize (OOA), kao i kod drugih metoda sistemske analize, namera je da se što bolje shvati sistem i njegovi zahtevi. Drugim rečima, OOA zahteva da se identifikuju objekti, njihovi atributi, ponašanja i relacije koji podržavaju zadate poslovne zahteve sistema. Faze obavljanja objektno-orijentisane analize su sledeće:




Slika 4.49. Primer dijagrama razmeštaja [1]

- Modelovanje funkcija sistema slučajevima korišćenja.
- Modelovanje aktivnosti slučajeva korišćenja upotrebom dijagrama aktivnosti.
- Pronalaženje i identifikacija poslovnih objekata.
- Organizovanje objekata i identifikovanje njihovih relacija.

### Modelovanje funkcija sistema

U narednom poglavlju će biti detaljnije objašnjen proces modelovanja zahteva slučajevima korišćenja. Tokom ovog procesa, dokumentovani su slučajevi korišćenja koji sadrže opšte informacije o poslovnim događajima. Cilj koji se želeo postići jeste brzo dokumentovanje svih poslovnih događaja (slučajeva korišćenja) kako bi se definisali i verifikovali svi poslovni zahtevi. Koraci kod analize modela slučajeva korišćenja su:

- identifikovanje, definisanje i dokumentovanje novih aktera;
- identifikovanje, definisanje i dokumentovanje novih slučajeva korišćenja;
- redefinisanje dijagrama slučajeva korišćenja (ukoliko je neophodno);
- dokumentovanje scenarija upotrebe slučajeva korišćenja.

Koristi od modelovanja slučajeva korišćenja su sledeće:

- Služi kao osnova za identifikovanje objekata i njihovih relacija i odgovornosti;
- Pruža pogled na ponašanje sistema iz ugla spoljnjeg korisnika;
- Predstavlja efektivan alat za validaciju zahteva;
- Predstavlja efektivan alat za komunikaciju;
- Služi kao osnova za testiranje plana;
- Služi kao osnova za korisnički priručnik.

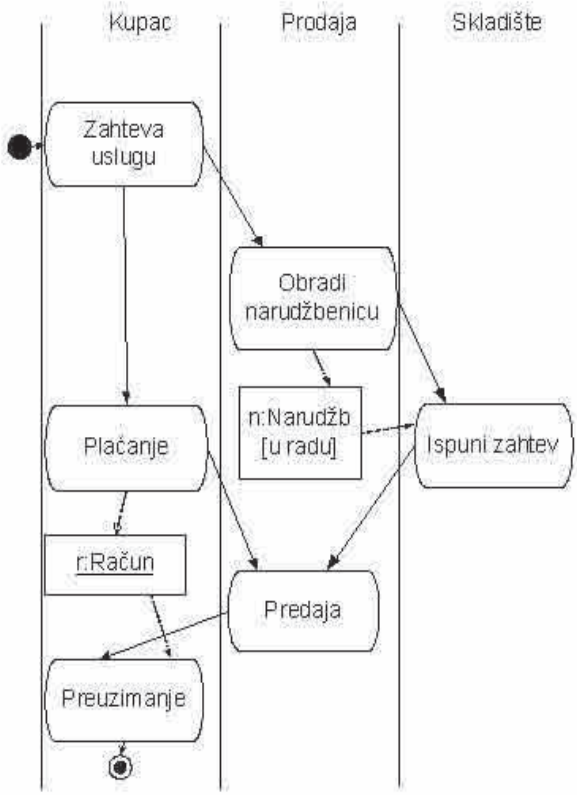
### Modelovanje aktivnosti slučajeva korišćenja upotrebom dijagrama aktivnosti

Da bi se modelovali koraci ili aktivnosti procesa sistema koristi se dijagram aktivnosti. Dijagram aktivnosti je veoma sličan algoritmu jer opisuje sekvencionalni tok aktivnosti svakog poslovnog procesa ili slučaja korišćenja. Za razliku od običnih algoritama, dijagrami aktivnosti opisuju aktivnosti koji se paralelno odvijaju. Primer dijagrama aktivnosti je dat na slici 4.50.

### Pronalaženje i identifikovanje poslovnih objekata

Koraci su:

- pronalaženje potencijalnih objekata;
- selekcija predloženih objekata;




Slika 4.50. Primer dijagrama aktivnosti

### Organizovanje objekata i identifikovanje njihovih relacija

Nakon identifikovanja poslovnih objekata sistema, neophodno je iscrtati dijagram klasa koji opisuje objekte i njihove asocijacije. Dijagram klasa uključuje i višestrukosti, relacije generalizacije/specijalizacije i agregacije.

Koraci su:

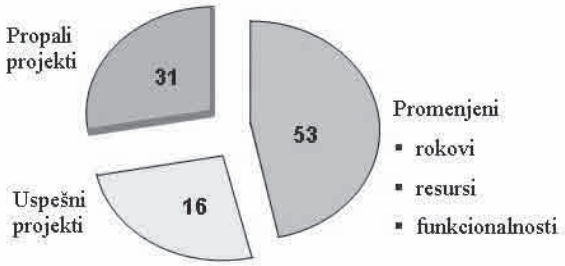
- identifikovanje asocijacija i višestrukosti;
- identifikovanje relacija genralizacije/specijalizacije;
- identifikovanje relacija agregacije.

# 5. Razvoj poslovnih rešenja prema MSF okviru

S obzirom da se u ovoj knjizi bavimo poslovnim informacionim sistemima, osvrnućemo se na metodologije koje obezbeđuju skup modela, principa i smernica za projektovanje i razvoj poslovnih rešenja. Jedan od poznatijih je MSF (*Microsoft Solution Framework*) kompanije *Microsoft*.

Zašto se većina kako inostranih tako i domaćih firmi okreće primeni MSF okvira u razvoju poslovnih informacionih sistema ili kraće poslovnih rešenja? Naime, u praksi se pokazalo da je svega 16% uspešnih projekata, dok je 31% propalih, a ostatak čine oni projekti koji se na neki način mogu nazvati uspešnim, ali su ili probijeni rokovi ili su isporučeni sa lošijom funkcionalnošću i slično (Slika 5.1). Ono što je i Microsoft hteo da postigne, a to je da je primenom MSF okvira zabeležen rast na 20% uspešnih projekata sa tendencijom porasta. Microsoft je u MSF ugradio najbolje prakse uspešnih projekata.

MSF nije metodologija već okvir (*framework*). Za razliku od metodologija koje daju tačne smernice, korake kojih se treba striktno pridržavati, *framework* je kao kompas, potvrđuje napredovanje i daje upravljačke smernice, služi kao podrška metodologijama.



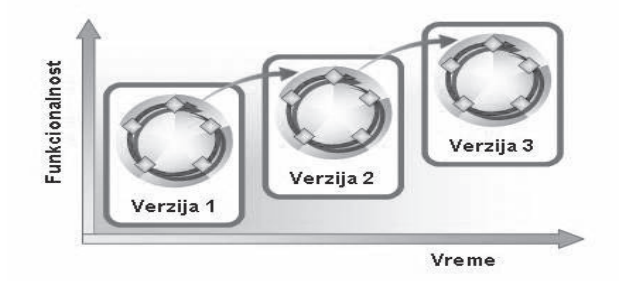

Slika 5.1. Procentualni prikaz uspešnosti projekata (Izvor: *Standish* grupa)

MSF obezbeđuje skup modela, principa i smernica za projektovanje i razvoj poslovnih rešenja na način koji osigurava da se svim elementima projekta (npr., ljudi, procesi, alati) uspešno upravlja. MSF takođe obezbeđuje dokazane prakse za planiranje, projektovanje, razvoj i uvođenje uspešnih poslovnih rešenja.

MSF kombinuje dva različita pristupa životnog ciklusa projekta:

- **model vodopada (*waterfall model*)** – kod ovog modela svi zadaci u jednoj fazi se moraju završiti pre nego što se pređe na sledeću fazu. Ovaj model se koristi kod onih projekata gde su zahtevi jasno definisani i nisu skloni modifikacijama u budućnosti.
- **spiralni model (*spiral model*) -** ovaj model se bazira na kontinualnoj potrebi za usavršavanjem projekta. Spiralni model je efektivan kada se koristi brzi razvoj aplikacija (*Rapid Application Development* - RAD) kod vrlo malih projekata. Ovaj pristup stvara veliku sinergiju između razvojnog tima i klijenata, jer klijent obezbeđuje povratne informacije i odobrava svaku fazu.

MSF procesni model opisuje generalne sekvence aktivnosti za izgradnju i uvođenje poslovnih rešenja. On kombinuje najbolje principe modela vodopada i spiralnog modela. Spiralnost se definiše i u okviru samog projekta, a i u okviru kritičnih tačaka (*milestones*) (Slika 5.2).




Slika 5.2. *Microsoft*-ov primer verzionisanja istog proizvoda

## 5.1. MSF discipline

MSF obuhvata discipline za upravljanje timovima, procesima, tehnologijom i rizicima, tj. elementima sa kojima se projekti najčešće susreću. Tri ključne MSF discipline su:

1. **Upravljanje rizicima (*Risk Management*)** – podržava proaktivno upravljanje rizicima, kontinualnu ocenu rizika i odlučivanje tokom životnog ciklusa projekta. Projektni tim kontinualno procenjuje, nadgleda i aktivno upravlja rizicima. Upravljanje rizicima prema MSF-u definiše šest koraka tokom kojih tim upravlja tekućim rizicima, planira i izvršava strategije upravljanja rizicima:

a. **Identifikovanje rizika –** primenom *brainstorming*-a mogu se identifikovati svi potencijalni rizici.

b. **Analiza rizika –** prema proceni verovatnoće događaja rizika i njegovog uticaja na sistem, rizici se sortiraju prema prioritetu.

c. **Planiranje rizika –** koriste se informacije dobijene analizom rizika kako bi se formulisali planovi, strategije i akcije. Za svaki rizik se procenjuje njegov uticaj na ishod projekta, navode se načini njegovom umanjenja i koraci koje treba sprovoditi ukoliko do rizika dođe.

d. **Praćenje rizika –** nadgleda se status određenih rizika.

e. **Kontrolisanje rizika –** proces izvršavanja planova akcija i njihovog izveštavanja.

f. **Učenje iz rizika –** formulišu se naučene lekcije kako bi se to znanje ponovo upotrebilo u sličnim slučajevima kod budućih projekata.

2. **Upravljanje spremnošću tima (*Readiness Management*)** – podrazumeva način podizanja nivoa spremnosti cele ekipe (motivacija, edukacija, interesantnost, iskustvo i dr.). Upravljanje spremnošću uključuje procese koji pomažu da se razviju znanja, veštine i sposobnosti neophodne za kreiranje i upravljanje projektima i rešenjima. Prate se sledeća četiri koraka:

   a. **Definisanje (*Define*)** – Definišu se scenarija, kompetencije i nivoi stručnosti neophodnih za uspešno planiranje, kreiranje i upravljanje rešenjima.

   b. **Ocena (*Assess*)** - Za svaku ulogu u organizaciji se analiziraju kompetencije i porede sa prethodnim korakom. Prave se planovi učenja.

   c. **Promene (*Change*)** – Tokom ovog koraka članovi tima poboljšavaju veštine i podižu nivo stručnosti do željenog nivoa. Prema planu učenja određeni članovi tima se obučavaju i prati se njihov napredak.

   d. **Procena (*Evaluate*)** - Tim procenjuje da li su planovi obuke bili efektivni i da li je stečeno znanje uspešno implementirano na poslu.

3. **Upravljanje projektima (*Project Management*)** – Kod MSF modela ne postoji uloga projekt menadžera, niti rigidna hijerarhijska struktura u procesu odlučivanja. MSF se bori protiv rigdnog, diktatorskog stila koji je prisutan kod projekt menadžmenta. Sve uloge u timu ispunjavaju određeni cilj i svi se smatraju podjednako bitnim. Glavne odluke se donose konsenzusom ključnog tima. Ukoliko se konsenzus ne može postići, program menadžer donosi finalnu odluku. Program menadžment je taj koji donosi odluke tako da one budu u skladu sa zahtevima klijenata. Upravljanje projektima je proces koji kombinuje skup veština i tehnika kako bi se ispunili sledeći zadaci:
   a. Integracija planiranja i upravljanje promenama
   b. Defininisanje i upravljanje ciljevima i oblašću projekta
   c. Pripremanje budžeta i upravljanje troškovima
   d. Pripremanje i praćenje rasporeda odvijanja projekta
   e. Osigurati da su odgovarajući resursi alocirani na projektu
   f. Olakšati timsku i spoljnu komunikaciju
   g. Olakšati proces upravljanja rizicima
   h. Dokumentovanje i nadgledanje procesa upravljanja kvalitetom tima.

Slika 5.3. MSF discipline

### 5.1.1. MSF procesni model

MSF procesni model (MSF *Process Model*) opisuje aktivnosti za izgradnju i uvođenje poslovnih rešenja u sistem. Ovaj model se može primeniti i na razvoj i uvođenje klasičnih aplikacija, poslovnih rešenja za elektronsku trgovinu (*e-commerce*) ili na Web distribuirane aplikacije.

MSF procesni model se sastoji od pet različitih faza:

1. **Stvaranje vizije (*Envisioning*)** – identifikuju se ciljevi, domen i ograničenja projekta, razmatra izvodljivost projekta, procenjuju neophodni resursi, formira se projektni tim i utvrđuju odgovornosti članova tima, analiziraju potencijalni rizici i dr.
2. **Planiranje (*Planning*)** – pravi se konceptualni, logički i fizički dizajn rešenja, projektuju sloj prezentacije, sloj podataka i specifikacije bezbednosti.
3. **Razvijanje (*Developing*)** – kodiranje rešenja.
4. **Stabilizovanje (*Stabilizing*)** - pregled pilot rešenja, pišu se korisnički *help*-ovi, priručnici za obuku i dokumentacija za rukovanje, vrši se testiranje i izveštava o bagovima.
5. **Uvođenje (*Deploying*)** – priprema se okruženje za uvođenje rešenja, ažurira se dokumentacija, pravi se plan testiranja, plan bezbednosti, *backup* plan, plan oporavka od šoka i vrši obučavanje.

Svaka faza završava sa jednom kritičnom tačkom (*milestone*). Na slici 5.4. se ilustruju faze i kritične tačke MSF procesnog modela. Da bi se prešlo iz prve u drugu fazu, fazu planiranja, neophodno je da je odobren Vizija/domen (*Vision/Scope*) dokument, odnosno dokument koji opisuje ciljeve projekta i ograničenja (zahtevi koji će se ispuniti, funkcionalnosti i inicijalni vremenski raspored). Da bi se iz faze planiranja prešlo u fazu razvoja rešenja neophodno je da se odobri projektni plan koji sadrži funkcionalnu specifikaciju konkretnog rešenja, plan projekta (*master project plan*), raspored projekta (*master project schedule*) i ažurirani dokument ocene rizika (*risk assessment*). U toku faze razvoja prate se performanse developera putem dnevnog rasporeda (*daily bill*), koji prikazuje koje funkcionalnosti moraju dnevno da budu završene. Kada se postigne željena funkcionalnost rešenja u okviru određenih granica projekta prelazi se na sledeću fazu – fazu stabilizacije.

Faza stabilizacije se finalizira kada su rešeni svi kritični i ozbiljni bagovi u rešenju i odobravanjem dokumenta spremnosti puštanja u rad (*Release Readiness Approved*). Faza uvođenja (*Deploying*) je završena kada je stabilizovano uvedeno rešenje i dobijeno i finalno odobrenje od klijenata.

Slika 5.4. MSF procesni model [9]

### 5.1.2. MSF timski model

Pored procesnog modela MSF obezbeđuje i timski model za organizaciju projektnih timova. Ovaj model ističe važnost jasnih uloga (rola), odgovornosti i ciljeva pojedinačnih članova projektnog tima. Fleksibilnost MSF timskog modela omogućava da se brže prilagodite ciljevima projekta, veličini tima i veštinama članova tima.

MSF timski model specifira šest različitih uloga (Slika 5.5):

1. **Proizvodni menadžment (*Product Management*).** Proizvodni menadžer vodi početne razgovore sa naručiocima posla, prikuplja zahteve, dogovara šta će se raditi, uspostavlja rokove, ograničenja.
2. **Program menadžment (*Program Management*).** Program menadžer sastavlja funkcionalnu specifikaciju (*master project plan* i *master project schedule*). Odgovoran je za razvoj i isporuku rešenja.
3. **Razvoj (*Development*).** Razvojni tim ili tzv. developeri su odgovorni za razvoj tehnološkog rešenja prema specifikacijama dobijenih od strane program menadžera.

4. **Testiranje (*Testing*).** Testiranje je jedna od važnijih rola. Tim koji vrše testiranje pronalazi *bug*-ove, probaju sve module kako bi finalno rešenje bilo što stabilnije, upoređuju funkcionalnosti programa sa ciljevima i vizijom projekta.
5. **Menadžment puštanja rešenja u rad *(Release management).*** Prave *Help*-ove, implementiraju i instaliraju sistem. Odgovorni su za rukovanje i održavanje sistema.
6. **Iskustveni korisnik *(User experience).*** Poznaje obe strane procesa, proizvodnje i upotrebe. Unosi podatke, analizira performanse, koriguje ekrane kako bi rešenje bilo što upotrebljivije i dr.

Kod manjih projekata, pojedinci projektnog tima mogu da imaju više uloga, ali to može dovesti do određenih rizika na projektu. Nije dobro na primer, da jednoj osobi budu dodeljene uloge program menadžera i developera.

Slika 5.5. MSF timski model

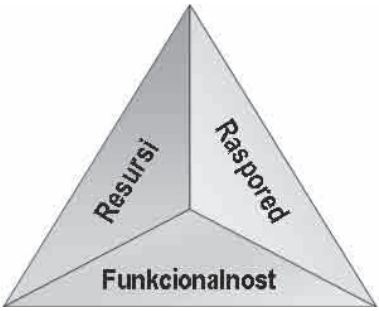
### 5.1.3. Upravljanje kompromisima

Projekti često propadaju jer su ili kasno završeni ili prevazilaze planirani budžet. Ovi problemi su često posledica nedovoljno jasno postavljenih ciljeva i granica projekta. Granice ili domen projekta tačno određuju šta će rešenje da radi, a šta ne. Da bi efektivno definisali i upravljali granicama projekta, neophodno je:

- identifikovati ograničenja projekta
- upravljati kompromisima (*tradeoffs*)
- uspostaviti kontrolu promena
- pratiti napredak projekta.

U projektima postoji jasan odnos između resursa, vremenskog rasporeda i funkcionalnosti projekta. S obzirom da je skoro nemoguće ostvariti istovremeno sve ciljeve, neophodno je upravljati kompromisima. MSF alati za upravljanje kompromisima su:

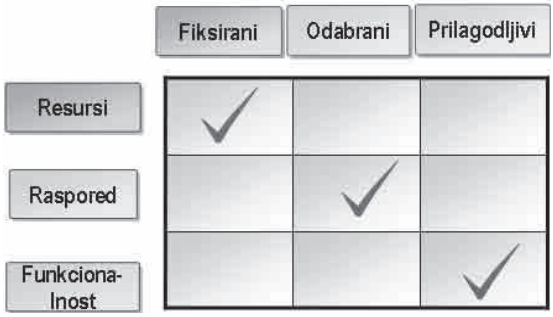
- **trougao kompromisa (*tradeoff triangle*)** – svaka promena na jednoj komponenti odražava promenu na drugoj. Projektni tim zajedno sa klijentima treba da odredi i održava korektan balans između resursa, datuma uvođenja rešenja i funkcionalnosti. Trougao kompromisa pomaže da se objasne ograničenja i predstave opcije kompromisa.




Slika 5.6. Trougao kompromisa

- **matrica kompromisa projekta (*project tradeoff matrix*)** – alat koga projektni tim i klijenti koriste kada treba da donesu odluke o kompromisima. Veoma je važno da se ove odluke donesu u ranim fazama projekta.

  Na slici 5.7. se prikazuje matrica kompromisa za slučaj da su resursi fiksirani, vremenski raspored određen, a funkcionalnosti su onda prilagodljive.




Slika 5.7. Matrica kompromisa projekta

## 5.2. Tehnike prikupljanja informacija i analiziranje poslovnih problema i zahteva

Efektivno prikupljanje podataka je krucijalno kod razvoja projekata. Sistemska analiza zahteva jedan organizovan metod prikupljanja činjenica. Kada prikupljate određene informacije, treba da budete svesni činjenice da postoje različiti tipovi i karakteristike informacija. Preduzeće je jedan dinamički poslovni sistem u okviru koga se informacije mogu svrstati u četiri kategorije:

- **Poslovne** – opisuje kako se radi posao, odnosno opisuje funkcije i unakrsne aktivnosti koje posao obavlja. Informacije iz ove kategorije opisuju i primarne ciljeve, proizvode i usluge, finansije, integrisane poslovne funkcije i procese, organizacionu strukturu i njihove međusobne veze. Ovde su uključene i poslovne strategije i planovi za prelazak iz postojećeg u buduće stanje sistema.
- **Aplikacije** – automatizovane i neautomatizovane usluge koje podržavaju poslovne procese. Uključuju gotove aplikacije, komponente, izvorne kodove koje su dostupne za analizu informacija ili funkcionalnosti zadatka.

  Često se dešava da se u nekoj organizaciji, identični zadaci više puta izvršavaju putem različitih alata. U toku prikupljanja podataka o procesima, treba istražiti različite aplikacije koje se koriste za obavljanje poslovnih aktivnosti kako bi se uočile potencijalne neefikasnosti ili redundantnosti. Ova kategorija obezbeđuje informacije o trenutnoj upotrebi sistema i usluga.

- **Operacije** – informacije koje su neophodne da bi se pokrenuo poslovni proces. Ova kategorija uključuje standardne modele podataka, politike upravljanja podacima i opisuje izvore informacija i njihovo korišćenje u poslovanju. Često korisnici ne znaju adekvatno da odrede koje su im informacije neophodne i šta treba da rade sa informacijom kada je dobiju. Identifikovanje porekla informacija, vlasništva i njihove upotrebe, omogućava praćenje i analiziranje informacija koje mogu poslužiti kao osnova za donošenje odluka vezanih za distribuciju podataka, particionisanje, repozitorijume, skladištenje podataka (*data warehousing*) i dr. Razumevanje ključnih poslovnih procesa i informacija koje su neophodne za njihovo obavljanje, omogućava da se postave standardi i smernice za kreiranje, prikupljanje, ažuriranje i brisanje informacija i podataka, za deljenje kritičnih dokumenata i podataka i za definisanje nivoa bezbednosti i standarda za pristup. Treba biti svestan činjenice da se sve informacije ne mogu naći na jednom mestu niti su pristupačne, većina kritičnih informacija se nalazi na serverima baza podataka, radnim stanicama ili često u glavama zaposlenih.
- **Tehnologije** – tehničke usluge neophodne za izvršenje i podršku poslovne misije. Uključuju topologije, razvojna okruženja, bezbednost, usluge mreže, DBMS, tehničke specifikacije, operativne sisteme, hardvere i dr. Ova kategorija takođe obezbeđuje informacije o standardima i smernicama za nabavku i uvođenje radnih stanica i serverskih alata, aplikacija, infrastrukturnih usluga, mrežnih komponenata i platformi. Tehnologije obezbeđuju vezu između aplikacija i informacija. Aplikacije se kreiraju i baziraju na različitim tehnologijama. One koriste tehnologije kako bi pristupale informacijama. Informacije iz ove kategorije se mogu koristiti za određivanje interfejsa, usluga i aplikacionih modela.

Kategorije informacija se mogu pronaći u različitim oblicima. Brojnost i raznovrsnost izvora informacija zavisiće od veličine poslovanja. Neke od informacionih izvora su:

- **Činjenice (*artifacts*)** – obezbeđuju informacije o zadacima, procesima, poslovnim potrebama i ograničenjima, na primer to mogu biti priručnici, finansijski izveštaji, dokumentacije, programske datoteke, video snimci itd. Prvi dokument koji bi analitičar želeo da vidi je organizaciona struktura. Organizaciona struktura služi da identifikuje ključne uloge na projektu kao i njihove odnose u

izveštavanju. Sledeće što bi analitičar prikupio jesu dokumenta koja opisuju probleme koje su dovele do projekta. Ta dokumenta mogu biti interna prepiska, predlozi, žalbe klijenata, izveštaji koji ukazuju na oblast problema, performanse, merenje rada, račune, zahteve za informacionim sistemima itd. Pored dokumenata koji opisuju probleme, proučavaju se i dokumenta koja opisuju poslovne funkcije. Ta dokumenta uključuju misiju kompanije, strategijski plan, ciljeve organizacionih jedinica, pravilnike, standardne procedure rada, instrukcije za obavljanje dnevnih operacija, forme koje predstavljaju transakciju u razlitim tačkama ciklusa obrade, baze podataka, ekrane i izveštaje. Analitičar će takođe proučavati i dokumentaciju svojih prethodnika kao što su različiti tipovi algoritama i dijagrama, rečnici projekata, repozitorijumi, dokumentacije projektovanja (inputi, autputi i baze podataka), programske dokumentacije, priručnici za obučavanje itd.

- **Sistemi** – pokazuju kako se obavljaju svakodnevne poslovne aktivnosti. Određeni sistem je skup diskretnih procesa koji izvršavaju jednu akciju. Sistem može biti sastavljen od podsistema. Sistem može da bude opipljiv proces, kao što je na primer sistem za praćenje zaliha ili neopipljiv kao što su na primer metode koje menadžer koristi da bi identifikovao i rešio probleme u okviru svog odeljenja. Sistemi mogu biti složeni jer mogu da sadrže više podsistema i mnoge kategorije informacija.
- **Ljudi** – Ljudi mogu biti vredan izvor informacija jer omogućavaju uvid u celokupno poslovanje preduzeća. Njih čine izvršioci, developeri, menadžeri, klijenti i korisnici. U slučaju gde je dokumentacija nekompletna ili čak ne postojeća, jedini izvor informacija su ljudi.

  Pre nego što priđete ljudima, odvojite vreme da identifikujete različite uloge (role) koje ljudi obavljaju na poslu. Na primer, grupa ljudi koja pruža podršku i informacije korisnicima, mogu da ukažu na oblasti koje predstavljaju najviše poteškoća korisnicima. Zaposleni sa iskustvom mogu da ukažu na to kako različite aktivnosti utiču jedna na drugu. Takođe, ukoliko preduzeće koristi spoljne prodavce (vendore) onda oni mogu da ukažu na efikasnost poslovanja i njihove procese.

### 5.2.1. Tehnike za prikupljanje informacija

Postoji šest glavnih tehnika za prikupljanje informacija:

- **Praćenje (*Shadowing*)** je tehnika koja podrazumeva direktno posmatranje kako pojedinac obavlja svoj posao u svom radnom okruženju kako bi se uvidela praksa i eventualni problemi. Da bi ste prikupili što više informacija podstaknite korisnika da vam detaljno objašnjava razloge obavljanja svih zadataka. Praćenje može biti:
  - **pasivno** - podrazumeva posmatranje korisnika i pasivno slušanje svih objašnjenja vezanih za obavljanje posla;
  - **aktivno** – pored posmatranja, vi postavljate i pitanja dok korisnik objašnjava događaje i aktivnosti. Takođe vam se može ukazati prilika da izvršavate neke od zadataka.

  Tehnika praćenja je korisna kod zadataka koji se često događaju. Praćenje je dobar način da se nauči kako korisnik ispunjava svoje dnevne zadatke. Međutim, može se desiti da korisnik u toku sesije ne izvrši sve zadatke koje su mu dodeljene, već samo određene. Na primer, računovođe mogu praviti određene izveštaje tek na kraju meseca, developeri mogu praviti nedeljne izveštaje itd. U toku praćenja poželjno je prikuptiti i dokumenta, ekranske forme trenutnog rešenja i dr. Ovom tehnikom se mogu uvideti i elementi trenutnog rešenja kao i procesi koji izazivaju kod korisnika nezadovoljstvo i frustracije. Tokom procesa praćenja prikupite što više informacija o poslu i zadacima koji se izvršavaju u sistemu, karakteristikama sistema, prioritetima, potrebama za dokvalifikacijom korisnika, kao i dokumenta i druge činjenice (artifakte).

  Tokom prikupljanja informacija tehnikom aktivnog praćenja, podstičite korisnika da odgovori na sledeća pitanja:
  - Kako je strukturian vaš posao?
  - Koje odluke treba da donesete pre nego što počnete da izvršavate određeni zadatak?
  - Kako implementacija rešenja definiše način obavljanja zadataka?
  - Koje varijacije u proceduri obavljanja zadataka postoje? Pod kojim uslovima se koriste varijacije?
  - Tokom date aktivnosti sa koliko ljudi ste u interakciji?
  - Kako prekidi utiču na vas? Da li možete brzo da nastavite tamo gde su vas prekinuli?

- Šta bi vam olakšalo obavljanje posla?
- Kako bi procesi mogli da budu efikasniji? Da li zadaci koji se obavljaju ručno mogu biti automatizovani?
- Koji zadaci mogu uticati na projektovanje rešenja?
- Koji su kriterijumi za merenje performansi?
- Kako bi funkcionalnosti rešenja trebale da budu strukturirane?
- Kako bi moglo da se poboljša trenutno stanje?
- Koje funkcionalnosti sistema se često koriste, a koje retko?
- Šta se korisnicima sviđa, a šta ne kod trenutnog sistema?
- Koja su korisnikova radna okruženja?
- Kako bi mogli da se smanje troškovi obuke i podrške?
- Koje informacije o korisnicima nisu dokumentovane?
- Koje su karakteristike i prednosti korisnika?
- Koje koncepte i terminologiju koriste korisnici?
- Kako novi korisnici uče da koriste trentni sistem? Da li novi korsnici idu na obuku ili uče na poslu? Koliko obučavanje utiče na njihovu sposobnost da koriste sistem i izvršavaju zadatke?

Praćenje ima i svojih ograničenja. Na primer, s obzirom da se informacije prikupljaju u toku redovnog obavljanja posla, možda nećete imati dovoljno vremena da postavite sva pitanja kako bi razjasnili svaku tačku. Ukoliko hvatate beleške može se desiti da ispustite neku kvalitetnu informaciju. Snimanje sesije kamerom može biti korisno kod prikupljanja podataka koji su ispušteni tokom sesije.

- **Intervju** je koristan kod prikupljanja podataka o procesima u okviru kojih se zadaci odvijaju relativno brzo, zatim o poslovnim temama iz ugla menadžera i o aspektima poslovanja ili zadataka koji se ne mogu direktno posmatrati, zbog toga što su ili automatizovani ili su se dogodili pre dužeg vremenskog perioda. Kvalitet prikupljenih informacija će zavisiti od veštine vođenja intervjua, kao i od kompententnosti osobe sa kojom se vodi intervju. Intervjuista može da nauči dosta o ograničenjima i poteškoćama trenutnog rešenja. Pre nego što počnete da vodite intervju evo nekoliko važnijih napomena:
  - Počnite za otvorenim pitanjima (pitanja na koje korisnik neće davati kratke odgovore) i ohrabrujte korisnike da razmišljaju o svim zadacima koje izvršavaju i informacijama koje mogu da daju.

- Koristeći odgovore na pitanja, pitajte korisnike da poređaju veće zadatke i da ih razbiju na nekoliko manjih.
- Posebno upitajte korisnike da ukažu na informacije koje često nedostaju i alternative koje se mogu dogoditi.
- Ponovite nekoliko puta korake obavljanja zadatka i zapitajte korisnika šta bi tu moglo još da bude uključeno.
- Pitajte korisnika o idejama za poboljšanje situacije.

Neka od pitanja koja bi mogli da postavite tokom intervjua su:

- Na koje probleme nailazite kada izvršavate zadatke?
- Koje poslovne politike vam pomažu ili vas ometaju u obavljanju svog posla?
- Koja dokumenta ili pojedinci vam pružaju bitne informacije neophodne za obavljanje vašeg posla?
- Kako drugi korisnici ili sistemi, kao na primer spoljni vendori, utiču na vaš posao?
- Koja vrsta pomoći vam je neophodna dok radite na terenu?
- Da li imate neke zahteve koji nisu dokumentovani?

Svrha intervjua je da se istraži, a ne da se ocenjuje i kritikuje. Koristite jasne i koncizne rečenice. Nemojte davati vaša mišljenja u okviru pitanja. Izbegavajte duga i složena pitanja. Trudite se da ne postavljate pitanja koja iziskuju više od jednog tipa informacija.

Vođenje intervjua se sastoji od tri faze:

- **Otvaranje intervjua** – namera je da se motiviše sagovornik stvaranjem idealnog okruženja. U toku stvaranja okruženja od obostranog poverenja i poštovanja, intervjuista treba da ukaže na svrhu i dužinu intervjua i da objasni kako će se prikupljeni podaci dalje koristiti. Evo nekoliko načina kako bi mogao efektivno da se počne intervju:
  1. Sumirajte očigledan problem i objasnite kako je problem otkriven.
  2. Ponudite određenu stimulaciju za učestvovanje.
  3. Pitajte za savet i pomoć.
- **Telo intervjua** – ova faza oduzima najviše vremena. Intervjuista mora pažljivo da sluša, posmatra i beleži kako verbalne tako i neverbalne odgovore. Poželjno je da se intervju snimi.
- **Zatvaranje intervjua** – zatvaranje je veoma važno za održavanje

dobrih odnosa i poverenja sa intervjuisanom osobom. Intervjuista treba na kraju da izrazi zahvalnost i odgovori na eventualna pitanja.

Primer vođenja intervjua je prikazan u tabeli 5.1. Jedini nedostatak kod intervjua je ukoliko se intervjuiše osoba koja nije ekspert iz domena koji se proučava, jer se tada dobija subjektivno viđenje posla i problema.

Tabela 5.1. Primer vođenja intervjua

| | | |
|---|---|---|
| Intervjuisan: | Mirko Mirković, menadžer kreditnog odeljenja | |
| Datum: | 10. Mart 2006. | |
| Vreme: | 10.30 | |
| Mesto: | Kancelarija 22, Zgrada administr | |
| Tema | Politika provere kredita | |
| **Dodeljeno vreme** | **Intervjuista**<br>**Pitanja ili ciljevi** | **Odgovori intervjuisanog** |
| 1-2 min. | **Cilj**<br>*Otvaranje intervjua:*<br>▫ Predstaviti se<br>▫ Zahvaliti se gospodinu Mirkoviću na dragocenom vremenu<br>▫ Izložiti svrhu intervjua – razumevanje postojeće politike provere kredita. | |
| 5 min. | **Pitanje 1**<br>Koje uslove klijent mora da ispuni da bi mu se odobrio kredit? | |
| 3 min. | **Pitanje 3**<br>Kako se klijenti obaveštavaju da im kredit nije odobren? | |
| 1 min. | **Pitanje 4**<br>Nakon što je odobren kredit i unešeni podaci u fajl, klijent može da zahteva neke izmene u podacima. Da li treba da se ponovi cela procedura odobravanja kredita iako je nova suma ista kao prethodna? | |
| 1-3 min. | **Pitanje 6**<br>Sa vašim dopuštenjem, da li bih mogao da razgovaram sa tim pojedincima kako bih naučio proces provere kredita?<br>Ako da: Kada bi bilo odgovarajuće vreme da se sastanem sa svakim od njih? | |
| 1 min. | **Cilj**<br>*Zatvaranje intervjua:*<br>▫ Zahvaliti gospodinu Mirkoviću za saradnji i uveriti ga da će dobiti kopiju zapisnika sa intervjua. | |
| 21 min. | Vreme za pitanja i ciljeve | |
| 9 min. | Vreme za podpitanja i preusmeravanja | |
| 30 min. | Vreme intervjua (10.30-11.00) | |
| **Komentari i beleške:** | | |

- **Fokus grupe** koristi tehnike grupnog intervjuisanja. Intervju se radi sa većom grupom kako bi se dobila prečišćena informacija, otkrila ponašanja i deljena mišljenja. Da bi se prikupile korisne informacije, neophodno je da učesnici fokus grupe predstavljaju korisnike posmatranog poslovanja. Fokus grupe obezbeđuju najkorisnije informacije kada se učesnici osećaju ugodno da dele svoja iskustva i diskutuju njihove različite poglede na poslovanje. Intervjuista treba da obrati pažnju da održava fokusiranost grupe na temu intervjua.
- **Istraživanje** podrazumeva prikupljanje detaljnih i statističkih podataka. Informacije koje se mogu prikupiti tehnikom istraživanja su:
  - organizacione strukture, politike i prakse koje olakšavaju obavljanje zadataka
  - posebne potrebe za hardverom i softverom
  - frustracije sa tehničkom podrškom
  - efektivnost trenutnog programa obučavanja, tipove programa koje korisnici vole i programe obuke koji daju najbolje rezultate u korisničkom radnom okruženju.
  - povratne informacije od klijenata i dr.
- **Instrukcije korisnika** – ova tehnika podrazumeva da se prema instrukcijama korisnika obavlja njegov posao. Krajnji korisnik vas podučava kako oni rade sa sistemom. Ova tehnika vam omogućava da tokom obavljanja aktivnosti sagledate svaki korak procesa iz ugla korisnika. Takođe, možete naučiti ono što je korisnik naučio vremenom, a to znanje se ne može naći ni u kakvim dokumentima niti u sistemu. Nedostatak ove tehnike je u tome što zahteva dodatno vreme korisnika, zatim se može desiti da korisnik ne zna da prenese svoje znanje ili da različiti korisnici različito obavljaju isti zadatak. Informacije koje se mogu prikupiti ovom tehnikom su:
  - dizajn korisničkog interfejsa,
  - potrebe za obukom u trenutnom i budućem sistemu,
  - kriterijumi performansi sistema,
  - uticaj fizičkog okruženja na zadatak.

- **Prototipovanje** podrazumeva prikupljanje informacija simuliranjem predloženog sistema koji nije moguće testirati direktno. Pravi se prototip koji se usavršava. U slučajevima kada je nemoguće pratiti osobu u normalnom radnom okruženju koristi se prikupljanje podataka na prototipu. Na primer, korišćenje prototipa može biti veoma korisno kada se prikupljaju informacije o softveru koji je napravljen za medicinske analize. U nekim slučajevima troškovi pravljenja prototipa mogu biti vrlo visoki. Informacije koje se mogu dobiti ovom tehnikom su:
  - zahtevi klijenta za kvalitetom
  - vreme odgovora na zahteve i ciljeve korisnika
  - lakoća korišćenja
  - integracija tehnologija i aplikacija
  - funkcionalnosti kod korisničkog interfejsa koje bi korisnik želeo da doda na aplikaciju.
  - vrednovanje *workflow* procesa.
- **Instrumenti (*Instrumented versions*)** – kod ove tehnike se koriste metode praćenja koje se ugrađuju u aplikaciju kako bi se snimilo kako korisnik izvršava zadatke. Ova tehnika vam omogućava da odgovorite na sledeća pitanja:
  - Koliko često se koristi određena funkcionalnost?
  - Koliko vremena je potrebno da bi se uradio zadatak?
  - Koje funkcionalnosti se najčešće koriste da bi se zadatak obavio?
  - Koje funkcionalnosti se ne koriste?
  - Koje teme *Help*-a se najčešće koriste?

  Informacije koje se mogu prikupiti ovom tehnikom mogu da ukažu, na primer, na određene funkcionalnosti koje se uopšte ne koriste, a projektovane su da bi se ubrzao proces obavljanja zadatka.

### 5.2.2. Analiza poslovnih problema

Da bi se lakše razumela priroda poslovnih problema, ovde se prikazuje scenario jedne multinacionalne proizvodne kompanije koja proizvodi i distribuira svoje proizvode na tržišta Evrope, Azije i Severne Amerike. Kompanija zapošljava oko 500 ljudi. Pre dve godine kompanija je kupila manji proizvodni pogon koji proizvodi nekoliko kritičnih podkomponenti za proizvodnu liniju kompanije. Ove komponente se transportuju na drugu lokaciju gde se obavlja finalna montaža. Već naredne godine ova fabrika postaje jedini proizvođač i distributer određene vrste proizvoda. Nakon uspešne fiskalne godine, kompanija želi da proširi tržišni udeo tako što će fokusirati prodaju na najbolje klijente, proširiti dostupnost proizvoda putem Web sajta i smanjiti troškove prodaje smanjivanjem proizvodnih troškova.

Poslovni problemi se mogu kategorizovati prema različitim odelenjima unutar kompanije kao što su Prodaja, Ljudski resursi, Nabavka, Informacioni sistemi, Proizvodnja, Administracija i Inženjering.

**Prodaja**. Kompanija poseduje regionalne prodajne kancelarije na teritoriji Australije, Kanade, Engleske, Francuske, Nemačke i Amerike. U svakoj regionalnoj kancelariji se nalaze reprezentanti prodaje i menadžeri. Reprezentanti koriste *laptop* kompjutere i *handheld* PC-jeve na kojima je instaliran *Microsoft Windows* CE. Tipičan radni dan jednog reprezentanta prodaje započinje tako što prvo učitava sve tekuće podatke vezane za zalihe, proizvode i promocione informacije. U toku dana, sve porudžbine od klijenata unosi u *laptop* ili *handheld* PC. Na kraju svakog radnog dana, reprezentant planira sastanke za sledeći dan ili nedelju, proverava sastanke drugih reprezentanata radi moguće saradnje i ažurira listu kontakata. Reprezentant na kraju unosi ažurirane informacije i proverava elektronsku poštu ili bilo koji drugi vid interne komunikacije sa glavnom ili regionalnim kancelarijama. Za elektronsku poštu kompanija koristi *Micorsoft Outlook*. Prodajni tim je identifikovao sledeće zahteve koji bi im omogućili da bolje rade svoj posao:

- **Segmentacija klijenata i profilisanje.** Prodajni tim treba da izvuče važne informacije iz sirovih podataka koji se nalaze u raznim bazama podataka, kako bi mogao da odgovori na sledeća pitanja:
  - Koji su rani znaci problema?
  - Koji su najbolji kupci kroz sve proizvodne linije? Prema kojim kupcima prodajni tim treba da fokusira svoj trud kako bi izgradio dugoročniju saradnju?

- Koja su pitanja klijenata, kategorizovane prema demografskim grupama (geografska lokacija, istorija prihoda itd.)?
- Koje proizvode kupuju klijenti i u kom odnosu?

- **Aktivnosti prodaje:**
  - Trenutna politika odobravanja popusta omogućava reprezentantima diskreciju da ponude popust do 15%. Menadžer prodaje može da poveća njihove popuste i popuste klijenata do 20%.
  - Da bi se podržala aktivnost prodaje širom sveta, prodajnom odeljenju je neophodna međunarodna podrška, što podrazumeva da informacije o proizvodima, datumima i cenama budu na više jezika i tipova valute.
- **Interna komunikacija.** Svaki reprezentant mora da dobija podatke o klijentima i prodaji. Svaki menadžer prodaje mora da prima relevantne podatke o klijentima, sastancima i detaljne informacije za svakog reprezentanta prodaje u timu. Menadžer treba da dodeli klijente reprezentantima prema njihovim odnosima sa klijentima (obično se klijenti dodeljuju prema regionima).
- **Upravljanje prilikama.** Reprezentantu prodaje je potreban metod za skladištenje i pristup podacima o prodajnim prilikama, a kada se generiše prodaja, da prenese određene ili sve informacije na porudžbinu bez potrebe za ponovnim unošenjem informacija.
- **Sistem za podršku odlučivanju.** Sistem za podršku odlučivanju treba da obezbedi sledeće funkcionalnosti:
  - da omogući marketing i prodajnom osoblju da postavljaju upite i koriste podatke klijenata za generisanje standardnih izveštaja; da izvršavaju kastomizirane upite; da dobiju informacije koje se odnose na praćenje promocije, prognoziranje prodaje i segmentacije klijenata; da pristupaju izvorima podataka i alatima za finansijsku evaluaciju.
  - da predstave sve aktivnosti klijenata (kontakti, konverzacije, transakcije) na jedan ujedinjen način.
  - omoguće marketing osoblju da iniciraju nove promocije i programe na multinacionalnoj osnovi. Trenutno, reprezentanti ne znaju kako na najbolji način da povežu ove programe sa određenim oblastima.
  - identifikuju, analiziraju i dele sve aspekte odnosa sa klijentima.

**Odeljenje za ljudske resurse.** Menadžment ove kompanije predviđa da će morati da zaposli još 35% radnika tokom naredne fiskalne godine kako bi se izašlo u susret planiranom povećanju proizvodnje. Takođe, menadžment je odredio da im je neophodno jedno aplikativno rešenje koje će pratiti biografije zaposlenih i prijave kandidata i koje će poboljšati prognoziranje neophodnog broja zaposlenih. Odeljenje za ljudske resurse zahteva sledeće funkcionalnosti od rešenja:

- **Upravljanje prijavama i biografijama**. Sve prijave i biografije se trenutno nalaze u dokumentima različitih formata. Sistem treba da obezbedi:
  - jedinstvenu formu i skladište za sve tipove fajlova;
  - pristup postojećim podacima zaposlenih sa mogućnošću povezivanja na njihove priložene biografije i prijave;
  - alate za konvertovanje svih tipova datoteka u dokumenta koja se mogu interno deliti kroz odeljenja;
  - bezbednost određenih polja u dokumentima, kao što su na primer informacije o platama;
  - pretraživanje prijava i biografija po ključnoj reči ili frazi.
- **Analiza i planiranje**. Odeljenje za ljudske resurse zahteva podršku za izvršavanje sledećih tipova analize:
  - analiza dobiti kroz međunarodni odnos valuta;
  - planiranje i procena neophodne radne snage;
  - prognoze i simulacije troškova plata.

**Nabavka**. Da bi proširila svoju infrastrukturu, kako bi podržala očekivani rast, kompanija je kupila fabriku koja proizvodi nekoliko kritičnih komponenti koje oni koriste u svojoj proizvodnoj liniji. Iako je fabrika odvojena poslovna jedinica kompanije, neophodno je da se neke aplikacije i podaci dele između ove dve institucije. Poslovni problemi u odeljenju nabavke su sledeći:

- **Prenos podataka**. Kompanija ne može redovno da premešta i prenosi podatke zbog toga što:
  - fabrika nema uređaje za visoko brzinski prenos podataka kako bi se preneli podaci iz njihove lokalne baze podataka u *Microsoft* SQL *Server* bazu podataka u centrali;
  - kompanija ima centralizovano okruženje; neophodno je da se poveća skalabilnost baze podataka proizvodnje prelaskom na distribuirano okruženje;

    - podaci se ne prenose uređajima za bezbednu mrežnu konekciju.
- **Administracija i podrška**. Fabrika ima limitiranu podršku za informacioni sistem. Odeljenja održavaju svoje radne stanice, a servere koji podržavaju *Oracle* bazu podataka, nadgledaju i održavaju dva administratora i menadžer za informacione sisteme. Fabrici nedostaju mnogi standardizovani procesi koje kompanija koristi u svom dnevom vođenju posla.

**Prodaja**. Kompanija trenutno koristi nekoliko dobavljača kako bi nabavila različite komponente i sirovine za svoju proizvodnu liniju. Odeljenje prodaje ima glavnog dobavljača koji je zainteresovan da uspostavi elektronsku razmenu podataka (*Electronic Data Interchange* – EDI) sa kompanijom kako bi prenosili kritična dokumenta i podatke kao što su porudžbine, račune, plaćanja i specifikacije o proizvodima. Agenti prodaje i računovođe utroše više od 50% svog vremena na rad sa glavnim dobavljačima. Nakon što se EDI rešenje u potpunosti implementira, menadžer prodajnog odeljenja predviđa da će doći do povećanja efikasnosti zaposlenih do 30%. Aplikacija treba da obezbedi sledeće funkcionalnosti:

- mogućnost da kompanija i dobavljači prenose i primaju mnoštvo podataka i tipova datoteka, uključujući i struktuirane i polustruktuirane podatke;
- mogućnost za dobavljače da direktno prenose podatke u *Microsoft* SQL *Server* tabele nabavke;
- Web zasnovani sistem koji će obezbediti pouzdane informacije dobavljačima;
- mogućnost da se automatski detektuje kada dobavljači imaju dolazeće fajlove ili druge podatke.

**Odeljenje za informacione sisteme**. Kompanija je razvila svoj Internet sajt, gde klijenti mogu da pristupaju osnovnim informacijama o proizvodima, pronađu najbližu prodavnicu i štampaju informacije. Oni ne mogu da naruče proizvod *on-line* ili da pregledaju status postojeće porudžbine. Mogućnost pretraživanja Web sajta prema informacijama o proizvodu je ograničena pretragom proizvoda po kategorijama. Kako bi omogućili dodatne funkcionalnosti njihov Internet sajt treba da uključi sledeće:

    - *on-line* porudžbinu proizvoda za klijente;
    - *on-line* proveru statusa porudžbine;
    - bolju pretragu informacija o proizvodima;

    - mogućnosti pristupa određenim tehničkim specifikacijama proizvoda;
    - mogućnost da pregledaju informacije o proizvodima i cenama u međunarodnim valutama.

**Proizvodnja**. Proizvodna linija kompanije se sastoji od devet proizvodnih grupa. Svaki proizvod sadrži jednu ili više podkomponenti, u zavisnosti od porudžbine klijenta. Trenutno, proizvodnja prihvata specifikacije od Inženjering odeljenja i koristi ga za sastavljanje proizvoda. Putem Web sajta, klijentima su na raspolaganju pojedine osnovne informacije o proizvodima. Međutim, klijenti nemaju pristup svim specifikacijama proizvoda. Specifikacijama proizvoda ima pristup prodajno osoblje, putem internog sajta. Neki od problema su:

- **Planiranje rasporeda i proizvodnje**. Da bi se izbegla kašnjenja u proizvodnji, tim treba da:
    - lako pristupa proizvodnim podacima, kao što su radni nalozi, nivo zaliha i dr.
    - omogući analizu podataka *workflow*-a.
- **Praćenje zaliha**. Umesto ručnog praćenja i ažuriranja sistema zaliha, kompanija želi da koristi skeniranje barkodova, koristeći pri tom *handheld* PC sa instaliranom aplikacijom za praćenje zaliha.

**Administracija**. Administracija nema centralnu kontrolu nad funkcijama upravljanja bazom podataka. Kao posledica toga, svaka grupa baze podataka razvija sopstvene prakse i procedure. Ovaj decentralizovani pristup ne koristi resurse optimalno, te stoga osoblje za informacione sisteme zahteva bolji način praćenja resursa kroz sve operativne sisteme. Administracija treba da obezbedi sledeće funkcionalnosti:

- dostupnost servisa za podršku ključnim sistemima kompanije;
- *backup* i sisteme oporavka za sve baze podataka;
- praćenje svih resursa putem svih operativnih sistema.

**Inženjering**. Odeljenje inženjeringa je odgovorno za projektovanje glavnih komponenti za sve proizvode kompanije. Timu treba takav sistem koji će im omogućiti upravljanje dokumentacijom. Neki od zahteva ovog sistema su:

- **kolaborativno upravljanje sadržajem dokumenata**. Odeljenju treba jedan automatizovan sistem koji će im omogućiti da kontrolišu kako se crteži i specifikacije pregledaju, odobravaju i šalju u proizvodnju na korišćenje.

- **pristupanje i skladištenje višestruktih tipova fajlova**. Odeljenje zahteva konzistentan način za skladištenje, pronalaženje, ispitivanje i arhiviranje različitih fajlova kao što su CAD crteži i XML specifikacije proizvoda. Ovaj sistem mora da bude povezan sa podacima koji se koriste za praćenje različitih verzija crteža i specifikacija.

### 5.2.3. Analiza zahteva

Aplikacija je Web zasnovana i namenjena je klijentima i drugim poslovnim jedinicama. Aplikacija treba da omogući i kandidatima da šalju i ažuriraju svoje prijave za posao.

Aplikacija mora da podrži sledeće poslovne zadatke:

- mogućnost slanja porudžbine od strane klijenta;
- mogućnost brisanja i odbijanja porudžbine;
- mogućnost modifikacije postojeće porudžbine koja je već obra-đena;
- pregledanje postojećih porudžbina;
- mogućnost kreiranja novih zapisa o klijentima;
- mogućnost da klijenti ažuriraju informacije o klijentima;
- prosleđivanje prijava za zapošljavanje na razmatranje;
- ažuriranje i pregledanje prethodno podnešene prijave;
- bezbednost pristupa informacijama.

Internet Web sajt kompanije treba da podrži sledeće funkcionalnosti:

- svaka strana na Web sajtu mora da prikaže opciju za pretraživanje sa odgovarajućim kontrolama za pretragu i navigacionim barom;
- klijentu treba omogućiti pretragu proizvoda po unosu dela opisa proizvoda ili opsega cena;
- *Home* strana mora da prikaže proizvode koji se prodaju pod specijalnim uslovima kao i odgovarajuće slike i opise;
- Web strana nazvana Grupe mora da prikaže hijerarhijsku listu linkova ka grupama modela proizvoda ili delovima proizvoda u strukturi drveta (*tree structure*);
- stranica Detalji proizvoda treba da prikaže informacije o pojedinim modelima proizvoda (naziv proizvoda, opis, slike i opseg cena). Ukoliko je model raspoloživ u više veličina, boja i sl., tada se mora prikazati *drop-down* lista, kako bi klijent mogao da bira veličinu ili boju;

- stranica Tekuća porudžbina mora da prikaže proizvode koje je klijent poručio, količinu svakog proizvoda, jediničnu i ukupnu cenu. Klijentu se mora omogućiti da menja količine ili ukloni artikle sa strane. Dugme Nastavak kupovine mora da obezbedi povratak na poslednju instancu posećene stranice o proizvodima;
- strana Prijavljivanje klijenta mora da omogući, registrovanim klijentima, prijavljivanje unošenjem *e-mail* adrese i šifre, a novim klijentima mogućnost registrovanja;
- strana Informacije o prijavi mora da omogući klijentima unos ili izmenu *e-mail* adrese, šifre i drugih ličnih informacija;
- strana Održavanje adrese treba da omogući klijentima kreiranje, pregledanje, ažuriranje i brisanje adrese fakturisanja i isporuke;
- strana Izbor adrese treba da omogući klijentima da biraju ili uklone adresu isporuke i fakturisanja sa porudžbine;
- strana Pregled porudžbine, treba da prikaže porudžbinu (da uključi i porez, troškove isporuke i detalje o popustima za preprodavce). Kupcima treba se omogući menjanje količine i uklanjanje artikala;
- strana Priprema plaćanja mora da omogući klijentima upotrebu kreditnih kartica ili unos novih informacija o kreditnim karticama. Preprodavcima da omogući biranje tipova plaćanja;
- strana Potvrda porudžbine mora da omogući pregled podnešene porudžbine (datum, broj potvrde, status porudžbine i dr.);
- strana Status porudžbine prijavljenom klijentu prikazuje listu svih unešenih porudžbina. Odabirom na odgovarajuću porudžbinu klijentu se prikazuje status željene porudžbine;
- strana Zaposlenje treba da omogući kandidatima podnošenje prijava.

## 5.3. Modelovanje sistemskih zahteva slučajevima korišćenja

Jedan od primarnih izazova od vitalne važnosti za tim koji se bavi razvojem informacionih sistema, naročito sistem analitičare ili proizvodne i program menadžere, je sposobnost korektnog prikupljanja sistemskih zahteva i njihovo opisivanje na način koji će biti razumljiv korisnicima kako bi ti zahtevi bili validni i verifikovani. IT zajednica je oduvek imala problem pokušavajući da odredi funkcionalne zahteve korisnika.

Industrija razvoja softvera je shvatila da bi uspešno planirala, analizirala, projektovala, izgrađivala i uvodila informacione sisteme, sistem analitičari moraju prvo da razumeju potrebe korisnika i razloge zašto sistem treba da bude razvijen – koncept nazvan korisničko orjentisan razvoj (*user-centred development*). Fokusiranjem na korisnike sistema, analitičar se koncentriše na to kako će se sistem koristiti, a ne na to kako će biti izgrađen. Jedan od načina da se struktuiraju analizirani zahtevi i problemi jeste da se nacrtaju modeli. Modeli služe da se bolje razumeju sistemi i dokumentuju poslovni zahtevi. Kao što slika vredi hiljadu reči, tako i većina modela predstavljaju slikovite predstave stvarnosti.

### 5.3.1. Model slučajeva korišćenja

Slučajevi korišćenja opisuju funkcije sistema iz perspektive korisnika na način koji je njima razumljiv. Preko slučajeva korišćenja se opisuje niz događaja koji se dešavaju kada korisnik koristi sistem da bi dovršio proces. Svaki slučaj korišćenja treba da se odnosi na jedinstveni merljivi zadatak ili cilj. Slučajevi korišćenja se sastoje od elemenata koji se nalaze unutar sistema i koji su odgovorni za funkcionalnost i ponašanje sistema.

Model slučajeva korišćenja pomaže učesnicima na projektu da se slože oko granica i funkcionalnosti sistema. Veoma je važno da se prate zahtevani zadaci od korisnika, rezultati tih zadataka i mogući izuzeci. Slučajevi korišćenja opisuju učesnike (*actor*), objekte (*objects*) i akcije (*actions*) koji dostižu ciljeve u sistemu.

Modelovanje slučajeva korišćenja uključuje:

- **narativne slučajeve korišćenja** – tekstualni opis poslovnih događaja i načina kako će korisnici biti u interakciji sa sistemom kako bi obavili zadatke. Primer iskaza slučajeva korišćenja se prikazuje na slici 5.8.

| | A |
|---|---|
| 1 | **Primeri slučajeva korišćenja** |
| 2 | |
| 3 | Agent prodaje pravi porudžbinu |
| 4 | Agent prodaje dodaje proizvod na porudžbinu |
| 5 | Kupac pretražuje proizvode |
| 6 | Kupac pregleda detalje o proizvodu |
| 7 | Kupac bira proizvod |
| 8 | Kupac kreira porudžbinu |
| 9 | Kupac dodaje proizvod na porudžbinu |

Slika 5.8. Primer iskaza slučajeva korišćenja

- **dijagrame slučajeva korišćenja** – grafički opisuju sistem preko slučajeva korišćenja, učesnika (aktera) i njihovih veza. Učesnik (akter) je korisnik koji je u interakciji sa sistemom. Akter može biti korisnik (čovek) ili neki drugi sistem. Treba napraviti razliku između korisnika i aktera. Korisnik je čovek koji koristi sistem, dok je akter specifična uloga koju korisnik ima u komunikaciji sa sistemom. Relacije ili veze povezuju dva simbola na dijagramu slučajeva korišćenja. Veze mogu biti:
  - **asocijacije (*associations*)** – označava komunikaciju aktera sa slučajevima korišćenja;
  - **proširivanje (*extends*)** – usled kompleksne funkcionalnosti, slučaj korišćenja se može podeliti u nekoliko koraka kako bi se pojednostavio i bio razumljiviji. Izvučeni koraci se predstavljaju kao posebni slučajevi korišćenja koji proširuju funkcionalnost originalnog kompleksnog slučaja korišćenja. Relacija *extend* se koristi da prikaže opciono ponašanje, ponašanje koje se pokreće samo pod određenim uslovima i nekoliko različitih tokova koji mogu biti pokrenuti na osnovu izbora aktera. Na primer, slučaj korišćenja koji prati tok paketa na traci može se proširiti slučajem korišćenja Aktiviranje alarma ukoliko se paketi zaglave.
  - **uključivanje (*includes, uses*) –** kada dva ili više slučajeva korišćenja izvršavaju korake od identične funkcionalnosti tada je najbolje da se oni predstave jednim apstraktnim slučajem korišćenja. Osnovni slučajevi korišćenja uključuju ponašanje opisano sa apstraktnim slučajem korišćenja. Služi da bi se izbeglo višestruko opisivanje istog ponašanja. Na primer, svaki

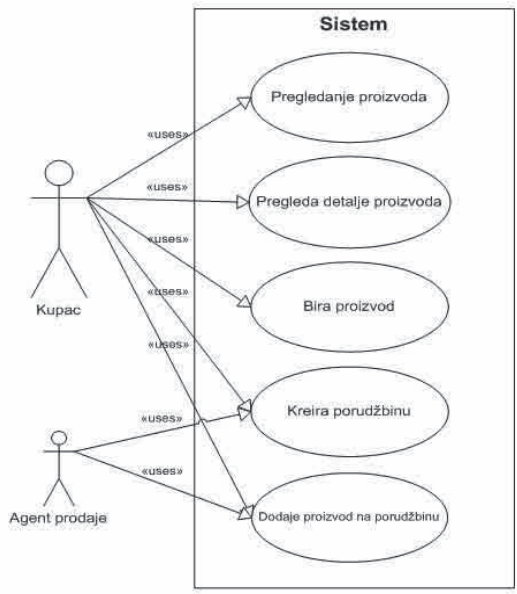
slučaj korišćenja u sistemu univerziteta za registraciju kurseva počinje sa proverom korisnika (*user verification*) koju zatim po potrebi koriste drugi slučajevi korišćenja.

- **zavisne od (*depends on*)** – veoma je korisno znati koji su slučajevi korišćenja zavisni od drugih, kako bi se odredio redosled razvoja slučajeva korišćenja. Na primer, slučaj korišćenja Podizanje kredita nije moguće izvršiti ako se prethodno nije dogodio slučaj korišćenja Otvaranje računa u banci.
- **nasleđivanje (*inheritance*)** – kada dva ili više aktera imaju ista ponašanja tj. iniciraju iste slučajeve korišćenja, onda je najbolje da se oni generalizuju u jednog apstraktnog aktera. Na ovaj način se smanjuje redundantnost komunikacije u sistemu. Na primer, član biblioteke može da pretražuje bazu knjiga, pregleda detalje knjige i izvrši rezervaciju. S obzirom da su biblioteke javne institucije, one omogućavaju i posetiocima da koriste njihove usluge kao što je pretraga baze knjiga, ali im se ne dopušta korišćenje drugih usluga (npr., Pregled detalja knjige). Kreiranjem apstraktnog aktera Klijent koga nasleđuju članovi i posetioci, samo se jedanput modelira veza koja inicira slučaj korišćenja Pretraga baze knjiga.

Dijagrami slučajeva korišćenja identifikuju granice sistema, sekvencu zadataka i hijerarhije. Na osnovu prethodno napisanih iskaza formira se dijagram slučajeva korišćenja (Slika 5.9).

### 5.3.2. Modelovanje zahteva slučajevima korišćenja

Cilj kod modelovanja zahteva slučajevima korišćenja je da se izvuku i analiziraju zahtevi tako da model komunicira sa onim što se zahteva iz perspektive korisnika, ali da bude oslobođen specifičnih detalja o tome kako će sistem dalje da se razvija i implementira. Koraci kod modelovanja zahteva slučajevima korišćenja su:




Slika 5.9. Primer dijagrama slučajeva korišćenja

**Prvi korak** je da se analiziranjem izvora informacija pronađu zadaci. Na primer, analiziranjem odlomka iz intervjua koji glasi: „Da bi identifikovali naše najbolje kupce i razloge zbog čega su oni najbolji, prodajno osoblje treba da pronađe način da pristupi i analizira našim podacima o prodaji.“, treba uočiti neophodne zadatke.

**Drugi korak** kod modelovanja je da se za svaki zadatak odrede učesnici, akcije i objekti akcije (Slika 5.10).

| Učesnik | Akcija | Objekat |
|---|---|---|
| Agent prodaje | pristupa | podacima o prodaji |
| Agent prodaje | analizira | podatke o prodaji |
| Agent prodaje | identifikuje | najbolje kupce |

Slika 5.10. Primer narativnih slučajeva korišćenja

**Treći korak** kod modelovanja je da se napravi dijagram slučajeva korišćenja. Prethodno je potrebno identifikovati sledeće:

- **Sistem** – skup međusobno povezanih podsistema usmerenih ka ostvarenju određenog cilja. Prvo treba identifikovati sistem ili podsistem. Skup slučajeva korišćenja ukazuje na veze između podsistema koji čine jedan sistem.

- **Učesnike (aktere)** – čine integralan deo slučajeva korišćenja. Slučajevi korišćenja predstavljaju veze između učesnika i sistema. Učesnik je entitet koji je u interakciji sa sistemom radi izvršenja nekog događaja. Da bi identifikovali učesnike, postavite sledeća pitanja:
  - Ko koristi, inicira i održava sistem?
  - Koji drugi sistemi koriste sistem?
  - Ko dobija informacije iz sistema i ko obezbeđuje informacije sistemu?
  - Da li se nešto dešava automatski u unapred određenom vremenu?
  - Ko ili šta inicira događaje sa sistemom?
  - Ko ili šta je u interakciji sa sistemom kako bi pomogao odzivu na događaje?
  - Da li postoje odnosi izveštavanja?
  - Da li postoje bilo kakvi odnosi sa administrativnim sistemom?
  - Da li sistem mora da bude u interakciji sa nekim postojećim sistemima?
  - Da li postoje neki drugi hardver ili softver koji su u interakciji sa sistemom i koje bi trebalo modelovati tokom analize?
- **Interakcije između učesnika i sistema** – nakon što ste identifikovali učesnike i sistem, sledeće je da se opišu interakcije između njih. Za svaku interakciju se pravi jedan slučaj korišćenja. Treba opisati samo one interakcije koje su važne za poslovanje.
- **Granice sistema** – jedan od najtežih aspekata modelovanja slučajeva korišćenja je određivanje tačne granice sistema koji će se razvijati. Da bi se definisale granice sistema, tim bi trebao da pokuša da odgovori na sledeća pitanja:
  - Šta se događa sa slučajevima korišćenja koji su pridruženi svim učesnicima?
  - Ko ili šta je trenutno u interakciji sa tim slučajevima korišćenja?
  - Da li će novi zahtevi biti deo sistema?
  - Da li su ti zahtevi neophodni za ovaj sistem?
  - Da li su ti zahtevi nešto što bi sistem logično radio?

- Da li bi tim zahtevima moao da rukuje jedan učesnik?
- Da li su ti zahtevi nešto što korisnici očekuju da sistem radi?

**Četvrti korak** je pisanje scenarija upotrebe (*usage scenarios)* koji obezbeđuju dodatne informacije o aktivnostima i sekvencama zadataka koje čine jedan proces. Jedan slučaj korišćenja predstavlja skup sekvenci događaja. Jedna sekvenca događaja se naziva scenario. Postoji osnovni scenario i skup mogućih izuzetaka i alternativnih funkcionisanja. Da bi se dokumentovao jedan slučaj korišćenja potrebno je napisati više scenarija upotrebe. Formalan opis slučaja korišćenja je moguće dati i preko dijagrama kolaboracije i dijagrama promene stanja. Međutim, preporučuje se, naročito u prvoj fazi, da se koristi struktuirani opis slučajeva korišćenja.

Scenarija upotrebe opisuju objekte i izuzetke. Objekat može da bude nešto na čega sistem utiče, nešto što utiče na sistem ili nešto što je neophodno sistemu da bi pravilno funkcionisao. Izuzeci su netipične ili alternativne sekvence zadataka koje obezbeđuju kompleten opis slučajeva korišćenja. Na primer, izuzetak se može dogoditi kada prilikom unosa novog kupca u bazu, sistem padne, te agent prodaje mora da koristi druge načine kako bi snimio informacije o kupcu.

Da bi se kreirao scenario upotrebe, neophodno je izvršiti sledeće zadatke:

- **utvrditi preduslove** - određivanjem informacija ili uslova koji moraju biti ispunjeni da bi se scenario izvršio;
- **identifikovati posledice** - identifikuju se posledice uspešno obavljene akcije;
- **razbiti aktivnosti na korake** - raspored svih poslova odnosno svakog pojedinačnog posla, kako bi posao bio obavljen;
- **identifikovati izuzetke** – izuzeci koji se mogu dogoditi na svakom koraku. Možda će biti neophodno da se naprave scenarija upotrebe za ove izuzetke;
- **identifikovati zahteve** na koje scenario ukazuje;
- **identifikovati izvor** za određeni scenario radi dalje diskusije i razjašnjenja. Odrediti na koji se slučaj korišćenja odnosi.

Da bi ste uspešno upravljali izuzecima u sistemu, evo nekoliko saveta:

- postavljajte "šta-ako" pitanja kako bi ste prikupili izuzetke za određene zadatke ili korake;

- odredite, za sve izuzetke, relativnu verovatnoću da se izuzetak dogodi;
- diskutujte kako upravljati izuzecima i koje su alternativne metode;
- uključite izuzetke sa visokom verovatnoćom događaja u projekat rešenja.

Mogu se kreirati scenarija za trenutno i buduće stanje poslovnog okruženja. **Scenario trenutnog stanja (*current state usage scenarios*)** opisuje kako se poslovne aktivnosti trenutno odvijaju odnosno kako trenutno proces izgleda. **Scenario budućeg stanja (*future state scenario*)** predstavlja željene buduće aktivnosti odnosno kako bi scenario trebao da izgleda. Scenario trenutnog stanja vam pomaže da:

- idetifikujete probleme u sistemu;
- odredite ciljeve i
- otkrijete nepodudarnosti između percepcije trenutnog i stvarnog problema.

Primer osnovnog scenarija za slučaj korišćenja Podizanje novca sa bankomata bi izgledao:

- Provera kartice: Komitent ubacuje karticu u automat. Automat čita karticu i proverava da li je prihvatljiva. Ako je prihvatljiva, zahteva se od komitenta da unese šifru.
- Proveravanje šifre: Komitent unosi tajnu šifru i ukoliko je ispravna zahteva se biranje željene transakcije.
- Unos tipa transakcije: Komitent bira transakciju Podizanje novca i automat šalje računaru banke tajnu šifru da bi se dobili brojevi komitentovih računa.
- Podizanje novca: Komitent bira račun i unosi iznos koji podiže. Automat šalje računaru banke zahtev za podizanje datog iznosa sa datog računa. Priprema se štampanje izveštaja za komintenta.
- Kraj: Automat vraća karticu komitentu, traženi novac i izveštaj o podizanju.

Alternativna scenarija kod slučaja korišćenja Podizanje novca bi bila:

- Kartica nije prihvatljiva: Kartica se vraća korisniku sa zvučnim signalom.

- Neispravna šifra: Na ekranu se prikazuje odgovarajuća poruka i korisniku se daje mogućnost ponovnog unosa šifre. Dozvoljena su tri pokušaja, a potom se kartica vraća korisniku.
- Prekid: Korisnik može u svakom trenutku da prekine transakciju. Sve prethodne akcije će biti poništene, a kartica će biti vraćena komitentu.
- Pad sistema: Komitentu se prikazuje odgovarajuća poruka.

Finalni izgled scenarija upotrebe na primeru slučaja korišćenja Upravljanje porudžbinama bi izgledao kao na slici 5.11.

**Peti korak** je identifikovanje zahteva korisnika. Zahtevi se mogu prikupiti analiziranjem intervjua, rezultata fokus grupa i drugih tehnika za prikupljanje informacija. Treba uočiti da slučajevi korišćenja i scenarija dokumentuju zahteve iz perspektive krajnjih korisnika ili objekata, a ne i zahteve samog sistema. Tako prikupljeni zahtevi se moraju dalje modifikovati kako bi reflektovali zahteve sistema. U ovoj fazi se uglavnom opisuju, organizuju i prema prioritetu sortiraju zahtevi i želje. Kasnije, u procesu razvoja proizvoda, razvojni tim će odrediti jasne karakteristike proizvoda. Koraci kod identifikovanja zahteva su:

- Kreirati listu zahteva tokom procesa prikupljanja informacija i identifikovati izvor odakle je zahtev došao.
- Zahtevi moraju da budu kratke (ostvarljive) rečenice.
- Proširiti listu zahteva preispitivanjem svih prikupljenih informacija i pronalaženjem novih zahteva.

**Sistem multinacionalne proizvodne kompanije**

**Autor:** _____ **Datum:** _____

| | | |
|---|---|---|
| **Naziv slučaja korišćenja:** | Upravljanje porudžbinama | Tip slučaja korišćenja<br>Poslovni zahtev: ☑ |
| **Slučaj korišćenja ID:** | Por002.00 | |
| **Prioritet:** | Visok | |
| **Izvor:** | Dokument zahteva Por-Z1.00 | |
| **Primarni učesnik:** | Klijent | |
| **Drugi učesnici:** | Agent prodaje | |
| **Opis slučaja korišćenja:** | Slučaj korišćenja Upravljanje porudžbinama opisuje događaj kada klijent kreira porudžbinu na Web sajtu kompanije. Proveravaju se sve informacije o klijentu i njegovom računu. Nakon čekiranja proizvoda koji se žele poručiti i popunjavanja ostalih neophodnih podataka na porudžbini, klijent šalje porudžbinu. Klijentu se šalje informacija o prihvaćenoj porudžbini sa pratećim podacima. | |
| **Preduslov:** | Kompanija ili pojedinac koji kreira porudžbinu mora da bude prethodno registrovan. | |
| **Triger:** | Slučaj korišćenja se inicira kada se pošalje porudžbina. | |
| **Sekvenca događaja:** | **Akcija učesnika** | **Odzivi sistema** |
| | **Korak 1:** Klijent prosleđuje informacije o sebi, porudžbini i plaćanju. | **Korak 2:** Sistem potvrđuje da su ispravno unešene sve tražene informacije.<br>**Korak 3:** Za svaki poručeni proizvod sistem proverava njegovo stanje na zalihama.<br>**Korak 4:** Za svaki proizvod sistem određuje cenu.<br>**Korak 5:** Nakon što su unešeni svi poručeni proizvodi, sistem određuje ukupnu cenu porudžbine.<br>**Korak 6:** Sistem proverava stanje na računu klijenta.<br>**Korak 7:** Sistem obrađuje porudžbinu i šalje klijentu potvrdu o prihvaćenoj porudžbini. |
| **Alternativne sekvence:** | **Alt.Korak 2:** Klijent nije obezbedio sve neophodne informacije. Sistem šalje klijentu odgovarajuću poruku. | |
| **Završetak:** | Ovaj slučaj korišćenja se završava kad klijent primi potvrdu o prihvaćenoj porudžbini. | |
| **Post-uslov:** | Porudžbina je obrađena. | |
| **Poslovna pravila:** | Klijentu se može odobriti poseban popust. | |
| **Ograničenja implementacije i specifikacije:** | Treba napraviti Web forme.<br>GUI treba da bude prilagođen prema prosečnim tehničkim znanjima klijenata. | |
| **Pretpostavke:** | Web forme će biti napravljene u C#.NET. | |
| **Otvorene teme:** | | |

Slika 5.11. Primer scenarija upotrebe

- Zahteve podeliti od želja. Zahtevi ukazuju na karakteristike procesa koji su osnovni za dostizanje ciljeva poslovanja. Želje se zasnivaju na iskustvu korisnika i predstavljaju idealno stanje posla. Želje su zahtevi koji mogu biti uključeni ukoliko to dozvoljavaju vreme i resursi. Želje su važne, ali ne i neophodne za dostizanje poslovnih ciljeva ili rešavanja poslovnih problema. U toku diskusije sa korisnicima, neke želje mogu postati zahtevi, a neki zahtevi mogu postati želje.

- Prepoznati ograničenja i pretpostavke. Ograničenje je fiksni limit. Ukoliko se ograničenja neadekvatno identifikuju, projektni tim može projektovati rešenje koje ne može biti uvedeno u poslovanje. Primeri mogućih ograničenja koje treba dokumentovati su:
  - limiti budžeta;
  - karakteristike postojećih sistema;
  - arhitektura mreže;
  - zahtevi bezbednosti;
  - operativni sistemi;
  - planirana poboljšanja (*upgrade*) u tehnologijama;
  - održavanje i podrška;
  - širina prohodnosti mreže;
  - nivo znanja razvojnog tima i tima podrške;
  - ograničenja učenja kod korisnika.

  Treba razjasniti pretpostavke kako bi se zaštitili od nesporazumevanja. Na primer: „Novo rešenje ćemo razvijati na Microsoft .NET tehnologiji“. Tim pretpostavlja da će razvijati rešenje koristeći .NET.
- identifikovati skrivene zahteve - neki zahtevi nisu odmah vidljivi (npr., zakonske regulative koje se menjaju).

**Šesti korak** je analiziranje informacija. Kada se prikupi dovoljno informacija, sledeći korak je filtriranje i određivanje onih informacija koje su relevantne za poslovanje.

Pre nego što se krene u analizu prikupljenih informacija treba utvrditi da postoji dovoljno informacija koje ukazuju na poslovne zahteve i zahteve za rešenjem, a koje uključuju:

- potrebe sigurnosti;
- podršku za rešenjem i karakteristike;
- planirane promene u poslovanju koje mogu da utiču na projektovano rešenje;
- performanse koje korisnici očekuju ili koje su neophodne da bi se održala konkurentnost;
- integraciju aplikacija – postojeće aplikacije koje treba da se povežu sa novim rešenjem;
- kako postojeći poslovni procesi utiču na rešenje.

**Sedmi korak** je dokumentovanje zahteva. Već tokom prikupljanja informacija od korisnika, započećete beleženje liste potencijalnih zahteva. U tom grubom, draft dokumentu će se nalaziti i zahtevi i želje koji će se u toku narednih faza analizirati i iščistiti. Mnoge želje će biti odložene do naredne verzije.

Na primeru multinacionalne proizvodne kompanije koju smo ranije spomenuli, analiziraćemo intervju koji je vođen sa menadžerom prodaje.

**Pregled intervjua:**

„Imali smo dobru proteklu godinu. Poslovi su nam dosta olakšani od kada nam je kompanija nabavila laptop računare i PDA. Ove godine smo radili prognoze i uvideli smo blagi pad u prodaji, naročito ukoliko bi nastavili da radimo na način koji smo radili do sada. Odeljenje prodaje je postavilo brojne ciljeve za novu fiskalnu godinu. Najvažniji ciljevi su (a) fokus na najbolje klijente (b) efektivnije korišćenje vremena prodajnog osoblja i (c) bolje upravljanje prodajnim prilikama. Analiza podataka o našim klijentima, nam je dosta otežana, jer se podaci trenutno nalaze na različitim sistemima i ne postoji lakši način prikupljanja podataka i mogućnost njihove analize i pregledanja iz različitih uglova.

Drugi problem koji smo iskusili tokom proteklih meseci je predstavljanje naših proizvoda i njihova prodaja na stranim tržištima. Trenutno su sve informacije, u našim računarima, memorisane na engleskom jeziku. Neophodno je da se memorišu i višejezične i multiregionalne informacije o proizvodima kako ne bismo zavisili od sposobnosti prevođenja prodajnog osoblja. Naš tim treba svakodnevno da dobija ažurirane informacije o cenama proizvoda. Trenutno se to radi tako što se agent prodaje, svakog jutra, konektuje na mrežu kompanije i *download*-uje novu listu cena. Međutim, na toj listi nije naznačeno koje cene su modifikovane i za kog agenta prodaje one važi.

Druga oblast, koju treba poboljšati, je upravljanje prodajnim prilikama. Zaposleni bi trebalo da koriste naš sistem za upravljanje klijentima kako bi planirali, izvršavali i pratili prodajne i marketing strategije. Agent prodaje bi trebao da instalira veliku aplikaciju na svoj laptop i da se konektuje na mrežu kompanije da bi mogao da je koristi. Želeli bismo da budemo mobilniji i da pristupamo tim informacijama preko Web sajta. Informacije moraju biti validne, značajne i pristupačne kako agentima prodaje tako i kompaniji, uopšte.

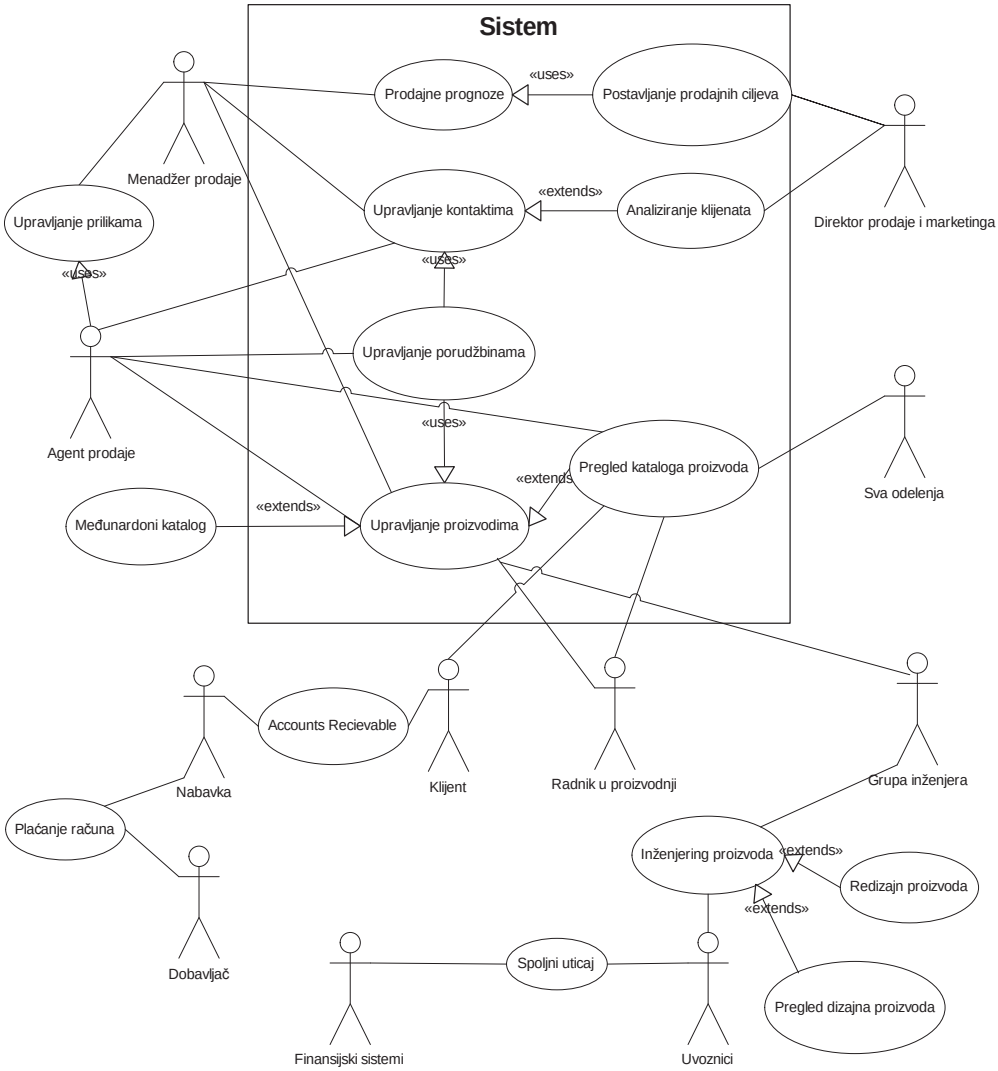
Sa novim sistemom bi želeli da postignemo sledeće ciljeve: da minimiziramo tehnička znanja potrebna za pristup podacima, dobijanje standardnih izveštaja, generisanje upita, praćenje promocija i pregledanje informacija o segmentaciji klijenata. Sistem treba da bude fleksibilan kako bi mogli da dodajemo podatke sa spoljnih izvora i alate za finansijsku procenu. Bez obzira ko gleda podatke o klijentima, taj pojedinac mora da ima jasan i jedinstven pogled na kijenta i njihove odnose sa nama."

Sledeća lista prikazuje preliminarni skup slučajeva korišćenja koji su uočeni tokom intervjua. Na ovoj listi još uvek nije izvršena analiza i validacija informacija.

Slučajevi korišćenja dobijeni sa intervjua:

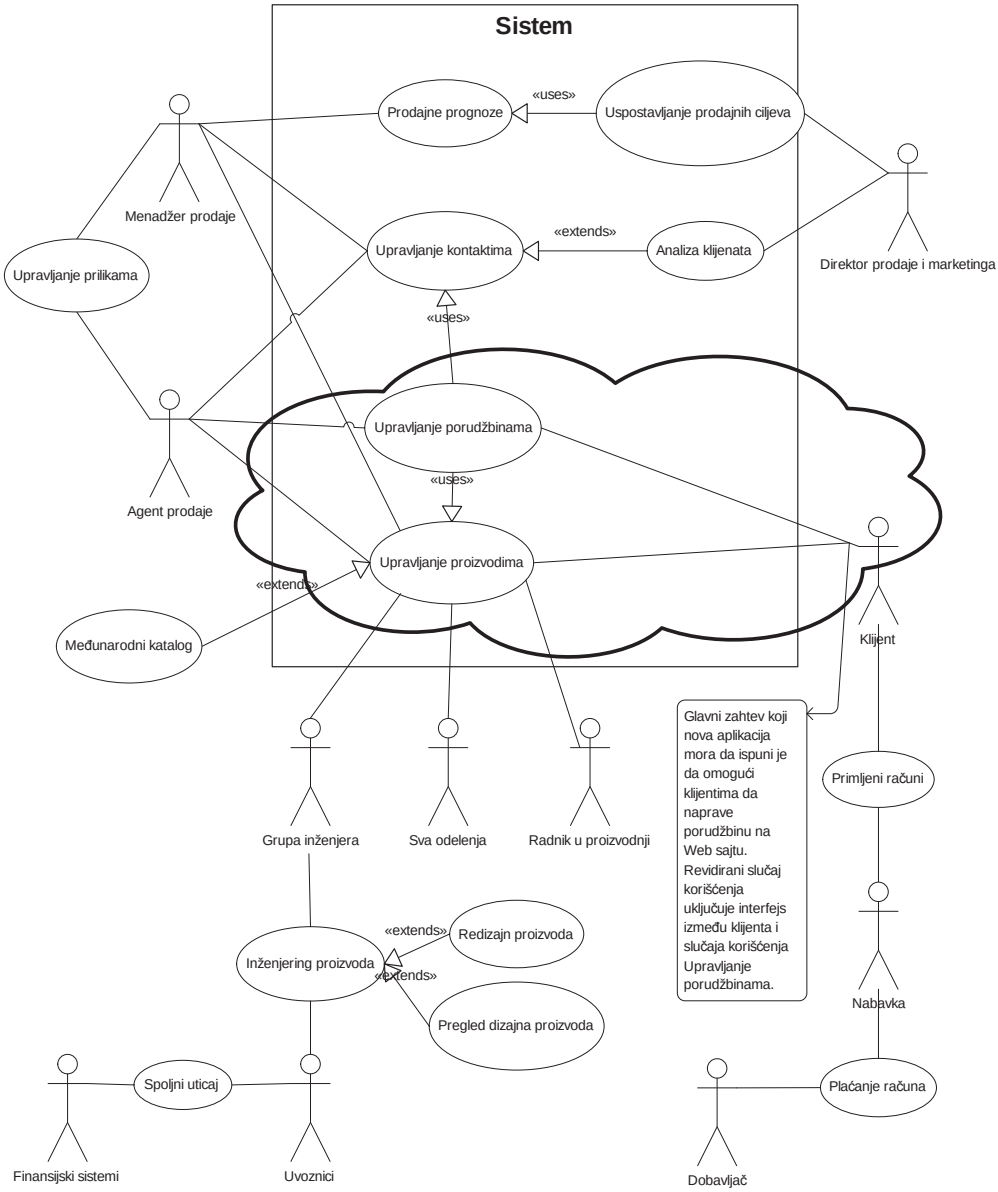
- Prodajno osoblje se bavi prognoziranjem prodaje.
- Prodajno osoblje postavlja ciljeve prodaje.
- Prodajno osoblje izvlači podatke o klijentima iz baze.
- Prodajno osoblje pristupa, pregleda i analizira podatke.
- Prodajno osoblje identifikuje najbolje klijente.
- Prodajno osoblje predstavlja proizvode na strana tržišta.
- Prodajno osoblje skladišti višejezične i multiregionalne informacije.
- Prodajno osoblje dobija dnevno informacije o cenama.
- Agent prodaje se konektuje na mrežu kompanije.
- Agent prodaje *download*-uje novu listu cena.
- Sistem identifikuje modifikovane cene.
- Klijent pregovara sa agentima prodaje.
- Zaposleni koriste sistem za upravljanje klijentima.
- Zaposleni planiraju prodajne strategije.
- Zaposleni planiraju, izvršavaju i prate marketing strategiju.
- Agenti prodaje pristupaju informacijama sa Web sajta.
- Prodajno osoblje dobija standardne izveštaje.
- Prodajno osoblje generiše upite.
- Prodajno osoblje prati promocije.
- Prodajno osoblje gleda izveštaj o segmentaciji klijenta.
- Prodajno osoblje dodaje podatke iz eksternih izvora.
- Prodajno osoblje dodaje alate za finansijsku evaluaciju.
- Prodajno osoblje procenjuje finansije klijenta.

Odgovarajući dijagram slučajeva korišćenja je prikazan na slici 5.12.




Slika 5.12. Dijagram slučajeva korišćenja izvučen iz intervjua [9]

Ovaj inicijalni dijagram slučajeva korišćenja predstavlja početno stanje odnosno prikazuje tekuće stanje sistema. Revidirani dijagram slučajeva korišćenja koji predstavlja procese onako kako bi trebalo da budu reinženjeringom procesa u sistemu se prikazuje na slici 5.13.




Slika 5.13. Revidiran dijagram slučajeva korišćenja dobijen reinženjeringom procesa [9]

Deo draft dokumenta zahteva izvučenih iz intervjua i slučajeva korišćenja je prikazan na slici 5.14.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | | | | |
| 2 | **Orig ID** | **Opis zahteva** | **Prioritet** | **Izvor** |
| 3 | | | | |
| 4 | 1 | Prikupljanje podataka od klijenata i mogućnost njihovog pregledanja na različite načine | da | Menadžer prodaje |
| 5 | 2 | Pronalaženje najboljih klijenata | | Menadžer prodaje |
| 6 | 3 | Dobijanje svih informacija o proizvodima | | Menadžer prodaje |
| 7 | 4 | Koji klijenti su najbolji kupci određenog modela | da | Menadžer prodaje |
| 8 | 5 | Predstavljanje naših proizvoda i njihova prodaja na stranim tržištima | | Menadžer prodaje |
| 9 | 6 | Treba memorisati višejezične informacije o proizvodima kako se ne bi oslanjali na sposobnosti prevođenja prodajnog osoblja | | Menadžer prodaje |
| 10 | 7 | Tim treba svakodnevno da dobija najnoviju listu cena | | Menadžer prodaje |

Slika 5.14. Grubi (*draft*) dokument

U toku analize informacija, tim kreira i nekoliko internih dokumenata i to:

- **katalog učesnika (aktera)** – sadrži informacije o svim učesnicima koji će se nalaziti na dijagramu slučajeva korišćenja. Katalog učesnika sadrži naziv učesnika, njegove odgovornosti i izvor ovih informacija. Na primeru naše kompanije, agent prodaje je jedan učesnik čije su odgovornosti da uspostavlja kontakte sa klijentima unutar regiona i posedovanje znanja o proizvodima i njihovim specifikacijama kao i informacija o svim klijentima. Na slici 5.15 je prikazan jedan primer kataloga učesnika.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | **Učesnik ID** | **Učesnik** | **Odgovornosti** | **Izvor** |
| 2 | 1 | Agent prodaje | Uspostavlja kontakte sa klijentima | Menadžer prodaje |
| 3 | | | Treba da zna sve o proizvodima i njihovim specifikacijama | |
| 4 | | | Upravlja porudžbinama | |
| 5 | | | Pregleda katalog proizvoda | |
| 6 | | | Upravlja prilikama | |

Slika 5.15. Primer kataloga učesnika

- **katalog poslovnih pravila** – dokument koji lista sva poslovna pravila rešenja. Ovaj dokument prethodi dokumentu zahteva i kreira se tokom faze prikupljanja informacija. Polja poslovnog kataloga su: identifikacioni broj poslovnog pravila, kratak naziv poslovnog pravila, opis poslovnog pravila, ovlašćeno lice, SK/PP (slučaj korišćenja ili poslovno pravilo na koje se odnosi) i trenutnu funkcionalnost.

Na primer poslovno pravilo koje se uočilo tokom vođenja intervjua i konverzacije sa menadžerom prodaje je to da agent prodaje može da ponudi popust od 10% svojim klijentima. Dok, menadžer prodaje može da odobri popust od 20% svojim klijentima. Na slici 5.16 je prikazan izgled kataloga poslovnih pravila.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | Poslovno pravilo ID | Naziv poslovnog pravila | Opis poslovnog pravila | Ovlašćeno lice | SK/PP | Trenutna funkcionalnost |
| 2 | 100 | Odobravanje popusta | Agent prodaje može da odobri popust do 10% | Menadžer prodaje | SK 5.01 | |

Slika 5.16. Primer kataloga poslovnih pravila

- **rečnik** – tim će u toku projektovanja poslovnog rešenja, uočiti termine koje korisnici najčešće koriste. Rečnik sadrži listu ovih termina i njihova značenja. Svrha rečnika je da svako koristi iste termine i da razume njihovo značenje.

## 5.4. Analiza izvodljivosti rešenja i upravljanje rizicima

Informacija je glavna kapitalna investicija koja mora biti opravdana, baš kao što marketing mora da opravda novi proizvod ili proizvodnja novi proizvodni prostor ili opremu. Proizvodni menadžer treba da pomogne u odgovorima na pitanja: Da li će investicija isplatiti samu sebe? Da li će se pojaviti druge investicije koje će povećati troškove? Analiza izvodljivosti treba da odgovori na ova i mnoga druga pitanja koja se javljaju u toku razvoja rešenja.

Izvodljivost (*feasibility*) je mera korisnosti i primenjivosti razvoja informacionog sistema u organizaciji. Analiza izvodljivosti (*feasibility analysis*) je proces kojim se meri izvodljivost. Izvodljivost mora da se meri tokom svih faza životnog ciklusa razvoja rešenja. Prva analiza izvodljivosti se sprovodi u toku faze definisanja vizije rešenja.

### 5.4.1. Faza definisanja vizije

Uspeh projekta zavisi od sposobnosti članova projektnog tima i korisnika da dele jasnu viziju ciljeva projekta. U ovom delu će se razmatrati uloge i odgovornosti članova tima tokom definisanja vizije projekta prema MSF procesnom modelu.

Faza definisanja vizije (*Envisioning*) je period tokom koga tim, korisnici i sponzori definišu prioritetne poslovne zahteve i sveukupne ciljeve projekta. Glavna svrha ove faze jeste da se osigura zajednička vizija i da se dostigne konsenzus između članova tima oko toga da je projekat važan za organizaciju i da se uspešno može realizovati. Tokom ove faze, fokus je na jasnoj definiciji problema. Tim sagledava kako poslovni problemi utiču na poslovanje, korisnike i okruženje.

Razmotrimo sledeći primer. Organizacija želi da se razvije Web sajt kompanije. „Treba nam Web sajt" nije dobra vizija. Da bi se kreiralo efektivno rešenje, tim treba da identifikuje ciljeve koje organizacija želi da postigne razvojem i uvođenjem Web sajta. Da li organizacija želi složen sajt ili sajt koji će da podrži rastući broj korisnika? Tim treba da razmotri ciljno tržište, očekivani obim posete i brend organizacije na tržištu.

U toku faze definisanja vizije tim:

- identifikuje ciljeve projekta i ograničenja (inicijalni rizici, zahtevi, uska grla, budžetska sredstva i dr.);
- odgovara na pitanja o izvodljivosti projekta;
- definiše domen projekta (granice projekta, šta nije uključeno u projekat);
- procenjuje neophodne resurse (broj ljudi, hardverske resurse i dr.);
- identifikuje i utvrđuje vremenski raspored projekta i određuje finalnu tačku završetka projekta.

Iako, projektni tim zajednički radi na uspostavljanju vizije i domena projekta, svaka pojedinačna uloga ima određeni fokus u toku faza životnog ciklusa projekta:

- **Proizvodni menadžment** - osigurava da se tim usredsredi na zahteve klijenata. Analizira poslovne probleme, poslovne zahteve, viziju projekta, poslovne ciljeve i uloge korisnika. Razmišlja globalno, ne fokusira se samo na trenutne potrebe korisnika. Kreira dokument Vizija/Domen (*Vision/Scope*).
- **Program menadžment** - uspostavlja ciljeve projektovanja, infrastrukturu projekta, definiše faktore uspeha, jasno postavlja koncept rešenja i uloge korisnika.
- **Developeri** - razmatraju i procenjuju tehničku izvodljivost rešenja.

- **Testiranje** – oni koji vrše testiranje izveštavaju tim o kvalitetu ciljeva rešenja i predlažu akcije kako bi se osigurao kvalitet.
- **Menadžment puštanja rešenja u rad** - uloga sistem administratora koji identifikuje zahteve za uvođenjem rešenja, potrebnu infrastrukturu i određuje kako i kada će rešenje biti uvedeno.
- **Iskustveni korisnik -** analizira performanse rešenja, neophodnu podršku korisnicima i razmatra da li je rešenje izašlo u susret potrebama korisnika.

Tokom faze definisanja vizije rešenja, važan segment je i formiranje tima. Veoma je važno da su članovi tima kompetentni za obavljanje neophodnih zadataka. Formiranje tima je veština od koje jednim delom zavisi uspešnost projekta. Da bi identifikovali članove tima, neophodno je razmotriti sledeće:

- **Znanje** – informacije koje pojedinac mora da zna kako bi kompetentno obavio posao.
- **Veštine** – ponašanje ili sposobnosti koje čine kompetentnost, na primer matematička logika.
- **Nivo performansi** – sposobnost ispunjavanja rokova i dostizanje kvaliteta u zadatim rokovima.

Dokumenti koji se formiraju u toku faze definisanja vizije su:

- **Vizija/domen dokument (*Vision/scope document*)** – opisuje ciljeve projekta i ograničenja (zahtevi koji će se ispuniti, funkcionalnosti i inicijalni vremenski raspored).
- **Dokument strukture projekta (*Project structure document*)** – prikazuje organizacionu strukturu projekta, proces upravljanja projektom, odgovornost članova tima i identifikuje vođu tima za svaku rolu.
- **Dokument procene rizika (*Risk assessment document*)** – obezbeđuje inicijalnu identifikaciju, analizu rizika i upravljanje rizicima.

Projektni tim formira i interna dokumenta (prethodno objašnjena):

- katalog učesnika;
- katalog poslovnih pravila;

- rečnik termina;
- kandidatne tehnologije (npr., SQL Server, Oracle, DB2 i dr.).

U zavisnosti od veličine projekta, može se priložiti i sledeće:

- inicijalna lista testiranih funkcionalnosti;
- preliminarni zahtevi i slučajevi korišćenja;
- preliminarna arhitektura;
- grafički korisnički interfejsi.

Projektni tim se uglavnom oslanja na dokument vizija/domen kako bi odlučio da li dalje da nastavi sa projektom. Ukoliko se projekat odobri, tim koristi dokument vizija/domen kako bi isplanirao dalji tok projekta.

**Dokument vizija/domen (*Vision/Scope* – V/S)** sadrži ciljeve i ograničenja poslovnog rešenja. On predstavlja prvi zvaničan dogovor između svih učesnika u projektu i vodi i usmerava tim ka dostizanju postavljenih poslovnih ciljeva. Da bi se kreirao V/S dokument, tim prethodno mora da vodi niz intervjua sa korisnicima, da analizira globalne slučajeve korišćenja i na kraju identifikuje pretpostavke i ograničenja. Treba napomenuti da se proces prikupljanja i analize informacija odvija tokom svih faza razvoja projekta. V/S dokument mora da se fokusira na definisanje i razumevanje problema. Sadržaj V/S dokumenta uključuje sledeće:

- **Definisanje problema (*Problem statement*)** - kraći narativni prikaz poslovnih očekivanja. Podvlači poslovne probleme koje tim mora da reši. Razumevanje navedenih problema određuje dizajn rešenja. Što su problemi preciznije postavljeni, veća je mogućnost da se izmeri njihov uticaj na poslovne potrebe organizacije. Primeri iskaza problema su: „Treba da povećamo broj *on-line* registracija, tako što ćemo napraviti Web sajt koji će biti lak za navigaciju“ ili „Korisnicima su potrebne jasne smernice iz sistema kako bi rešavali greške kada se pojave“.
- **Viziju (*Vision statement*)** - Kreirati koncizan, jasan i motivirajući iskaz koji uspostavlja zajedničku viziju i omogućava timu da dostigne konsenzus. Vizija mora biti kratke sadržine da bi se lakše upamtila, mora biti jasna da bi se razumela i dovoljno jaka da bi bila motivirajuća za tim. Dobra vizija sadrži sledećih pet SMART karakteristika:

- **Specifična (S*pecific*)** – usredsređena na poslovni problem. Vizija mora da bude specifična i da uključi idealno stanje poslovnog problema tako da krajnji rezultat bude značajan.
- **Merljiva (M*easurable*)** – kreiranjem vizije koja je merljiva, projektni tim može da odredi uspešnost projekta sagledavanjem ispunjenosti poslovnih ciljeva.
- **Dostižna (A*chievable*)** – sa datim resursima, vremenskim okvirima i veštinama tima, vizija mora da bude dostižna. Takva vizija će motivisati članove tima da završe projekat.
- **Relevantna (R*elevant*)** – mora da bude usredsređena na određeni poslovni problem, ukoliko nije tako, onda projektni tim može da uvidi da pokušava da reši poslovni problem koji uopšte ne postoji.
- **Vremenski-zasnovana (T*ime-based*)** – treba jasno da ukaže na očekivano vreme isporuke rešenja.

Na primer: „Do kraja godine, postaćemo najproduktivnija proizvodna kompanija u industriji povećanjem naših *on-line* prodaja".

- **Korisnički profili (*User profiles*)** – Veoma je važno shvatiti ko su korisnici za koje se razvija rešenje. Da bi se dobio jasan opis svakog korisnika, tim kreira korisničke profile (*user profiles*). Korisnički profili identifikuju korisnike tako da tim može proceniti njihova očekivanja, rizike, ciljeve i ograničenja na projektu. Kada se identifikuju korisnici treba razmotriti sledeće:
  - **Ciljeve** – uključuje se lista stvari koje korisnik očekuje u toku interakcije sa proizvodom.
  - **Ograničenja** – koja utiču na sposobnost korisnika da koristi rešenje.
  - **Podrška** – korisnički profili sadrže informacije o problemima koje korisnici mogu da imaju sa sličnim proizvodima. Ove informacije će pomoći u planiranju podrške koja će koristiti korisniku tokom upotrebe novog proizvoda.
  - **Globalni korisnici i geografske granice** – Odrediti da li će rešenje podržati i različite kulture i potrebe lokalizacije.
  - **Tok informacija između korisnika** – Opisati komunikaciju koja se događa između korisnika, uključujući i tipove komunikacija, njihovu važnost i obim podataka.

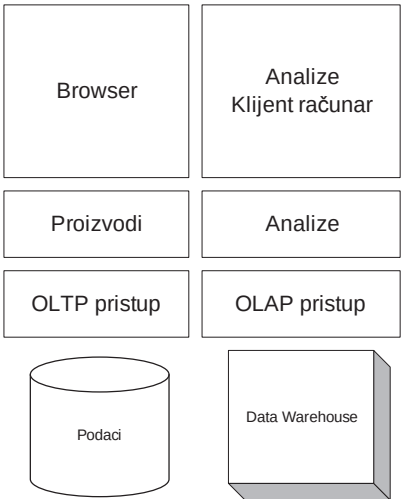
- **Funkcije korisnika** – Opisati zadatke koje korisnik izvršava, npr. "unosi porudžbinu klijenta i detalje o porudžbini".
- **Organizaciona komunikacija** – neke organizacije imaju rigidnu hijerarhijsku podelu koja ograničava načine, razloge i kada pojedinci komuniciraju kroz hijerarhijske nivoe. Dokumentovati sastav i granice organizacione hijerarhije.
- **Politike donošenja odluka** – koje direktno utiču na efektivnu implementaciju predloženog rešenja.
- **Dodatni faktori** – Neophodno je dokumentovati i raspoloživost i upotrebu resursa na korisničkim lokacijama i identifikovati bilo koje dodatne faktore kao što su nekompatibilni protokoli, mrežni operativni sistemi i dr.

- **Domen projekta (*Scope of the project*)** – Domen ili granice sistema opisuju šta će biti, a šta neće biti uključeno u projekat. Granice sistema se određuju na osnovu vizije projekta i obaveznih funkcionalnosti koje rešenje mora da ima. Važna aktivnost kod određivanja granice sistema je definisanje i prečišćavanje zahteva koje budući projekta mora da ispuni. Primer preliminarnog zahteva izvučenog iz intervjua sa korisnikom: "Sistem mora da podrži lokalizaciju kako bi odgovarala kulturama krajnjih korisnika.". Nakon prečišćavanja ovog preliminarnog zahteva, u toku faze sagledavanja vizije projekta, zahtev glasi: „Sistem mora da ima procedure za globalizaciju, lokalizaciju i dostupnost.". Usled ograničenih resursa, vremena i drugih limitirajućih faktora mora se navesti koje funkcionalnosti će zadovoljiti trenutno rešenje, a koje će se razvijati u sledećoj verziji. Na određivanje granica sistema takođe utiču i pretpostavke i ograničenja projekta. Na primer, korisnik iznosi sledeće pretpostavke: „Koristićemo Microsoft .NET Framework za razvoj rešenja. Koristićemo dva razvojna tima. Imaćemo i OLAP i OLTP sisteme." Ove pretpostavke znatno utiču na ciljeve projektovanja budućeg rešenja. Ograničenja su: budžet, karakteristike ranijeg sistema, mrežna arhitektura, operativni sistemi, nivoi znanja korisnika. Granice sistema se obično označavaju na dijagramu slučajeva korišćenja, kao što je prikazano na slici 5.12.
- **Koncept rešenja (*Solution concept*)** – Nakon identifikovanja poslovnih problema i definisanja vizije i granice sistema, tim kreira konceptno rešenje koje opisuje kako tim namerava da realizuje zahteve projekta. Fokus je na koncepte, a ne na detalje rešenja.

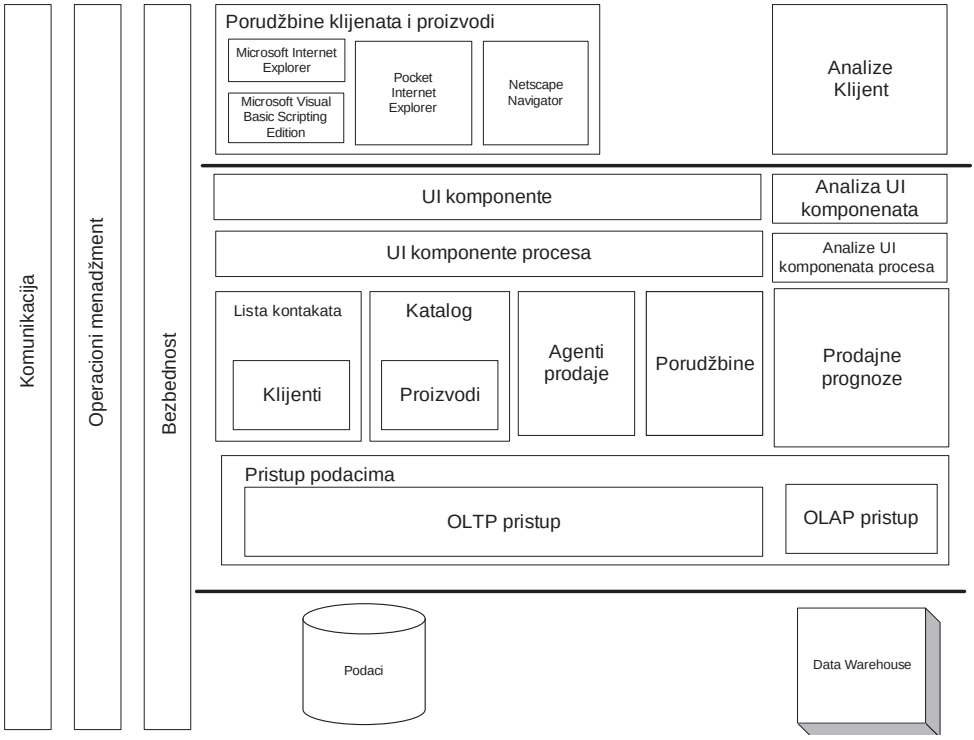
Konceptno rešenje uključuje konceptualni model arhitekture hardvera i softvera sistema. Na primer: u slučaju jednog Web sajta koji treba da podrži elektronsku trgovinu (*e-commerce*), treba doneti odluku da li da se *e-commerce* sajt radi od samog početka ili da se angažuje spoljna kuća koja će razvijati rešenje ili pak da se nabavi gotovo komercijalno rešenje. Slično, kod odlučivanja gde će se hostovati taj isti sajt, postavljaju se sledeća pitanja: Da li da se sajt hostuje na lokalnim serverima ili kompanija treba da koristi usluge provajdera? Nakon što je tim procenio opcije, on se odlučuje za konceptno rešenje koje najbolje odgovara njihovim potrebama, resursima i vremenskim okvirima. Konceptno rešenje uključuje sledeće elemente:

- **Faktore uspešnosti projekta i kriterijume prihvatljivosti** – uključuju zahteve koji moraju biti zadovoljeni pre nego što se krene u realizaciju rešenja.
- **Inicijalni pristup razvoju i isporuke rešenja** - uključuju scenarije načina implementacije rešenja, broj korisnika koji će koristiti novo rešenje i kompletnu listu faktora koji će omogućiti da novo rešenje bude funkiconalno.
- **Inicijalni opis funkcionalnosti rešenja** koje ukazuju na poslovni problem.

Na primeru proizvodne kompanije, na slici 5.17 se prikazuje proces razvoja od koncept rešenja do konceptualnog modela.




a) Primer koncept rešenja




b) Primer konceptualnog modela

Slika 5.17. Primer razvoja od koncept rešenja do konceptualnog modela

- **Ciljevi projekta** - Da bi projekat bio uspešan, neophodno je precizirati ciljeve projekta. Ciljevi projekta se mogu kategorizovati u:
  - **poslovne ciljeve (*business goals*)** – ono što klijent želi da postigne sa rešenjem, analiza matrice kompromisa i postavljanje prioriteta. Na primer, za projekat *e-commerce* Web sajta, poslovni ciljevi bi bili:
    - proširiti tržište kompanije i tamo gde nema prodavnica.
    - skratiti vremena prodaje proizvoda efikasnijim korišćenjem *on-line* sajta.
    - integrisati sve dobavljače, skratiti vreme porudžbine i isporuke.
  - **ciljeve projektovanja (*design goals*)** – fokusiraju se na atribute rešenja i postavljaju se prioriteti među ciljevima. Na primer, u slučaju *e-commerce* sajta, ciljevi projektovanja bi bili:
    - smanjiti vreme učitavanja (*download*) strane;

    - smanjiti zavisnost od konekcije sa serverom;
    - smanjiti vreme i nivo napora koji je nophodan da bi korisnik popunio *on-line* registraciju.
- **Kritične faktore uspeha (*Critical success factors*)**
- **Inicijalni raspored projekta (*Initial schedule*)**.

Nakon kreiranja preliminarnog ili draft V/S dokumenta, tim nanovo pregleda i modifikuje dokument. Zatim organizuje sastanak sa klijentima kako bi se sporazumeli o sledećim pitanjima:

- poslovne potrebe koje bi trebalo finalno rešenje da ispuni;
- vizija i domen rešenja;
- kompromisi koji su dogovoreni kako bi se definisale granice sistema;
- ciljevi projektovanja rešenja;
- rizici kojima se mogu izložiti preduzimanjem projekta;
- inicijalni koncept upravljanja projektom poslovnog rešenja;
- članovi projektnog tima;
- mehanizmi za upravljanje projektom.

Nakon usaglašavanja stavova po svim pitanjima, projektni tim zahteva od korisnika da verifikuju, a time i formalno odobre V/S dokument. Kraj faze definisanja vizije je odobren dokument vizija/domen koji predstavlja finalni dogovor između projektnog tima i klijenta o sveukupnim smernicama projekta.

Drugi dokument koji treba da se dostavi program menadžeru je **dokument strukture projekta (*project structure*)**. Dokument strukture projekta definiše način na koji će tim organizovati i upravljati projektom, uključuje:

- **Uloge i odgovornosti tima i klijenata** – lista imena ljudi koji su uključeni u projekat i njihove kontakt informacije. Opisuje odluke prema odgovornostima različitih uloga tokom narednih faza projekta. Tokom faze planiranja, tim treba da odluči o sledećim pitanjima:
    - Koji alati će se koristiti za razvoj i praćenje planova projekta?
    - Kako će se funkcionalnosti rasporediti po različitim verzijama rešenja?

- Ko će identifikovati rizike i definisati njihove izdatke?
- Da li će tim organizovati sastanke tokom faze planiranja i ko će im prisustvovati? i druga pitanja.

U toku faze razvoja rešenja, tim treba da donese odluke o sledećim pitanjima:

- U toku rada na projektu sa kojim grupama, organizacijama ili vendorima će tim biti u interakciji? Ko je odgovoran za ugovaranje sa spoljnim vendorima?
- Koje su uloge i odgovornosti vođe tima u toku razvoja rešenja?
- Ukoliko različiti timovi razvijaju različite komponente rešenja, koje su njihove pojedinačne uloge i odgovornosti?
- Da li će svaka komponenta biti dodeljena timu za podršku, iskustvenim korisnicima i timu za testiranje?
- Koja je uloga podugovarača (*subcontractor*) na projektu? Ko će birati podugovarače i pratiti njihov rad? i druga pitanja.

- **Komunikacione odluke** – prikazuje procese komunikacije unutar tima i sa klijentima koji će se pratiti tokom projekta. Uključuju se sledeći odgovori na pitanja:
  - Koga treba izveštavati o odlukama planiranja?
  - Kako će se klijenti, korisnici i tim informisati o odlukama u toku planiranja?
  - Koji tipovi sastanaka će se održavati, gde će se održavati i ko će im prisustvovati?
  - Ko će praviti agendu sastanaka i distribuirati je?
  - Ko mora da bude informisan o napretku projekta i kako će se informisati? i dr.

  U toku projekta održavaju se fajlovi koji sadrže informacije o ugovorima, rasporedima i planovima. Takođe, u okviru dokumenta strukture projekta tim treba da odluči koje informacije će se čuvati u fajlovima projekta. Neke od odluka koje treba doneti, vezanih za fajlove su:
  - Koje informacije će biti uključene u fajlove projekta, npr. specifikacije projekta, rasporedi, planovi, ugovori i dr.?

- Ko će kreirati i održavati fajlove projekta?
- Ko će im pristupati?
- Ko će održavati fajlove projekta, nakon završetka projekta i koliko dugo će se čuvati? i dr.

- **Logističke odluke** – dokumentuju odluke vezane za razvoj re-šenja:
  - Koje metode razvoja će se koristiti, npr. metode testiranja, provera nula defekata, specifikacija proizvoda, marketing metode i dr?
  - Ko će definisati sadržaj specifikacije proizvoda i ko će je i kako koristiti?
  - Koji alati će se koristiti za definisanje funkcionalnosti re-šenja?
  - Ko će izveštavati tim o izmenama na specifikaciji proizvoda?
  - Kako će se odrediti kriterijumi za završetak specifikacije proizvo-da?
  - Koje specifikacije će se dostavljati spoljnim vendorima koji rade na projektu i ko će ih kreirati?
  - Kako će se definisati i implementirati metoda nula defekata na projektu?
  - Kome je neophodna dodatna obuka, npr. kurs upravljanja projekti-ma? i dr.
- **Odluke upravljanja promenama** – dokumentuju se procesi koje tim treba da prati kako bi upravljao promenama na projektu. Odluke koje treba da se dokumentuju su:
  - Kako će se definisati promene na projektu (na primer, promene u rasporedu ili u povećanju troškova)?
  - Ko će i na koji način odrediti datum puštanja rešenja u rad?
  - Ko će i kako identifikovati i pratiti promene na projektu?
  - Kako će se meriti uticaj promena na projektu? Kolika je dozvo-ljena devijacija pre nego što dođe do promene rasporeda? i dr.
- **Odluke procene napretka** – dokumentuju se odgovori na sledeća pitanja, vezanih za praćenje i procenu napretka projekta:
  - Kako će se oceniti napredak projekta? Koje informacije će se koristiti za merenje napretka?

- Kako i koliko često će se prikupljati te informacije od članova tima?
- Koliko često će se ažurirati grupni raspored projekta?
- Ko će identifikovati i oceniti uticaj svake promene?
- Kako će se rešavati problemi između različitih timova?

Da rezimiramo, komponente dokumenta strukture projekta su ukratko:

- Tim i struktura;
- Ocene projekta (*Project estimates*) - model troškova po fazama (Slika 5.18);

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | | | | |
| 2 | | | Faza 0 (3 meseca) | Faza 1 (2 meseca) | Faza 2 (2 meseca) | Faza 3 (5 meseci) | | Ukupno | Ukupno po grupama | Komentari |
| 3 | | | | | | | | | | |
| 4 | *Ukupno* | | 40.000 | -127.090 | -389.150 | -587.210 | | -1.063.450 | Raspoloživ novac ili trošak na početku svake faze | |
| 5 | | | | | | | | | | |
| 6 | Developeri | | | | | | | | 600.000 | |
| 7 | | dev 1 | 30.000 | 60.000 | 60.000 | 150.000 | | 300.000 | | 30 časova nedeljno |
| 8 | | dev 2 | 30.000 | 60.000 | 60.000 | 150.000 | | 300.000 | | 30 časova nedeljno |
| 9 | SQL timovi | | | | | | | | 204.000 | |
| 10 | | rad | 32.000 | 96.000 | 32.000 | 32.000 | | 192.000 | | 2 ljudi 200 po času |
| 11 | | uređaji | 3.000 | 2.000 | 2.000 | 5.000 | | 12.000 | | struja, radni prostor itd. |
| 12 | Prodaja | | | | | | | | - | |
| 13 | | | - | - | | - | | - | | nije na teret projekta |
| 14 | Menadžment | | | | | | | | 68.460 | |
| 15 | | mgr 1 | - | - | | - | | - | | nije na teret projekta |
| 16 | | mgr 2 | 17.340 | 11.560 | 11.560 | 28.000 | | 68.460 | | 5 sati nedeljno, 282 po času |
| 17 | Putovanja | | | | | | | | 82.000 | |
| 18 | | dev 1 | 12.000 | 8.000 | 8.000 | - | | 28.000 | | 2 putovanja mesečno |
| 19 | | dev 2 | 12.000 | 8.000 | 8.000 | - | | 28.000 | | 2 putovanja mesečno |
| 20 | | mgr 1 | 2.000 | 2.000 | 2.000 | - | | 6.000 | | 1 putovanje po fazama |
| 21 | | mgr 2 | 2.000 | 2.000 | 2.000 | - | | 6.000 | | 1 putovanje po fazama |
| 22 | | QA 1 | 6.000 | 4.000 | 4.000 | - | | 14.000 | | 1 putovanje mesečno |
| 23 | Oprema | | | | | - | | | 7.000 | |
| 24 | | Razna | 2.000 | 4.000 | 1.000 | - | | 7.000 | | |
| 25 | Licence softvera | | | | | | | | - | |
| 26 | | SQL Server | | | | | | | | |
| 27 | | | | | | | | | | |
| 28 | Ukupni troškovi po fazama | | 148.340 | 257.560 | 190.560 | 365.000 | | 961.460 | 999.210 | |
| 29 | | | | | | | | | | |
| 30 | Deficit | | -108.340 | -384.650 | -579.710 | -952.210 | | -2.024.910 | Raspoloživ novac ili trošak na početku svake faze | |

Slika 5.18. Novčana procena projekta

- Dokument strukture projekta čiji sadržaj bi izgledao:
  - Svrha dokumenta (*Purpose of document*);
  - Raspored projekta (*Schedule*) – prikaz odgovarajućeg gantograma;
  - Model troškova (*Cost model*) – prikazan na slici 5.18;

- Tabelarni prikaz datuma kada se završavaju dokumenta vizije/domen, strukture projekta, glavni plan projekta, funkcionalna specifikacija, raspored glavnog projekta, procena rizika, aplikacija, obuka korisnika i priručnici za korisnike i tehnička dokumentacija;
- Struktura tima (*Team structure*) – sadrži tabelarni prikaz pregleda uloga u timu (uloga, opis, ime i prezime člana), tabelarni prikaz odgovornosti svake pojedinačne uloge (uloga, cilj, funkcionalne oblasti, odgovornosti) i tabelarni prikaz liste kontakata (ime i prezime člana, uloga u timu, telefon i email);
- Pristup upravljanju promenama (*Change Management Approach*) – prikazuje odgovorne za proces kontrole promena i dokumentacije, funkcionalnosti prema verzijama softvera i njihove procene, matricu kompromisa i dr.
- Ograničenja (*Constraints*) – opisuje poslovna ograničenja, ograničenja na projektu i tehnička ograničenja.

### 5.4.2. Analiza rizika

Jedan od bitnijih aspekata za razvoj uspešnog rešenja je upravljanje rizicima koji čine sastavni deo svakog projekta. **Dokument procene rizika (*Risk assessment document*)** je poslednji dokument u nizu koji treba da se kreira tokom faze definisanja vizije projekta pre nego što se pređe na narednu fazu – fazu planiranja.

Rizik je mogućnost gubitka. Rizik može biti i sve ono što umanjuje kvalitet rešenja, povećava troškove, premašuje rokove ili ukazuje na pad projekta. Osnovni aspekt razvoja uspešnog rešenja je da se kontrolišu i umanjuju rizici prisutni u projektu.

**Upravljanje rizicima (*Risk management*)** je proaktivan pristup kontrolisanja i umanjivanja rizika koji definiše šest koraka kroz koje tim upravlja tekućim rizicima, planira i izvršava strategije upravljanja rizicima i dokumentuje iskustva i znanja za buduće projekte. Većina pojedinaca koncept rizka poistovećuje sa gubitkom vrednosti, kontrole, funkcionalnosti, kvaliteta ili neispunjenja vremenskog roka završetka projekta, međutim rizici podrazumevaju i neuspeh u dostizanju maksimalnih mogućnosti ili nesigunosti pri donošenju odluka što može dovesti do

propuštanja poslovnih prilika. Upravljanje rizicima je kontinualni proces koji se odvija tokom svih faza projekta. Tim kontinualno procenjuje šta to može poći naopako i kako sprečiti ili minimizirali svaki mogući gubitak. Proces upravljanja rizicima prolazi kroz šest logičkih koraka, koji se ne moraju hronološki pratiti:

- **Identifikovanje rizika** – Identifikovati rizike i predočiti potencijalne probleme svim članovima tima.
- **Analiza rizika i sortiranje prema prioritetu** – Pretvoriti informacije i podatke o potencijalnim rizicima u korisne informacije za tim koje će oni dalje sortirati prema prioritetu rešavanja problema.
- **Planiranje rizika i pravljenje rasporeda** – Informacije kreirane u fazi analize rizika upotrebiti za formulisanje strategije umanjenja rizika, planova i akcija. Rasporediti vremena za planiranje rizika u projektnom planu.
- **Praćenje rizika i izveštavanje** – Nadgledati status određenog rizika i progres njegovog plana za umanjenje. Izvestiti tim i korisnike o tom napretku;
- **Kontrola rizika** – Izvršiti plan umanjenja rizika i izvestiti tim i korisnike o statusu rizika;
- **Učenje iz rizika** – Dokumentovati naučene lekcije iz projekta tako da ih tim i korisnici mogu opet upotrebiti.

U toku faze definisanja vizije, tim upravlja rizicima tako što kreira dokument procene rizika, identifikuje i dokumentuje sve moguće rizike i ocenjuje rizike prema verovatnoći pojave i uticaja na projekat. Dokument procene rizika može da sadrži sledeće delove (Tabela 5.19):

- **Iskazi o rizicima (*Risk statements*)** – opisuje prirodu svakog rizika;
- **Verovatnoća rizika (*Risk probability*)** – opisuje verovatnoću da se rizik desi (na primer, u skali od 1 do 10);
- **Mera rizika (*Risk severity*)** – opisuje uticaj rizika na sistem;
- **Izloženost riziku (*Risk exposure*)** – određuje sveukupnu pretnju rizika. Izračunava se množenjem verovatnoće i mere rizika:

$$\text{Verovatnoća } (P) * \text{Uticaj } (I) = \text{Izloženost riziku } (R);$$

- **Planovi umanjenja (*Mitigation plans*)** – opisuje planove sprečavanja ili umanjenja rizika;
- **Planovi kontigencije i trigeri (*Contingency plans and triggers*)** – specifira korake koji se preduzimaju kada se rizik dogodi i kada treba preduzeti date korake;
- **Odgovorni za rizik (*Risk ownership*)** – naznačava imena članova tima koji su zaduženi i odgovorni za praćenje i uopšte upravljanje određenim rizicima.

Program menadžer kreira dokument procene rizika. Pre nego što dokumentuje koncept rešenja u okviru dokumenta vizija/domen, tim može da proceni rizike za svako konceptno rešenje. Na primer, ukoliko je projekat kreiranje ERP poslovnog rešenja organizacije, onda se mogu razmotriti različite opcije. Jedna opcija bi bila da se rešenje kreira od samog početka, što dovodi do angažovanja dodatnih resursa, povećanja vremena razvoja i dr. Druga opcija bi bila nabavka gotovog rešenja, odnosno njegovog izvornog koda i da se postepenom kastomizacijom, rešenje prilagodi zahtevima korisnika. Očigledno je da će druga opcija umanjiti rizike koji bi se pojavili ukoliko bi se rešenje razvijalo od samog početka.

Nakon identifikovanja svih mogućih rizika koji se mogu pojaviti na projektu, tim dalje procenjuje svaki rizik pojedinačno, koristeći sledeću formulu:

$$R = p \bullet i$$

gde je:

R – izloženost riziku

$p$ – verovatnoća da će se gubitak dogoditi;

$i$ – uticaj rizika na sistem.

Ovaj proces vam omogućava da poredite rizike i odredite njihove uticaje i prioritete. Na primer, ukoliko koristite brojeve od 1 do 5 za označavanje verovatnoće i uticaja za svaki rizik, gde 5 označava najveću, a 1 najnižu verovatnoću i uticaj, tada se dobija faktor izloženosti rizika u rasponu od 1 do 25. Na ovaj način se mogu uočiti najkritičniji rizici na koje bi tim prvo trebao da se usredsredi. Na osnovu prethodnog proračuna, program menadžer će napraviti listu od, na primer, 10 najkritičnijih rizika, koju će s vremena na vreme ažurirati prema važnosti rizika (Slika 5.19).

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Lista ocene rizika | | | | | | | | | |
| 2 | | | | | | | | | | |
| 3 | | Opis rizika | | (Skala 1-100%) | (Skala 1-10) | | | | | |
| 4 | Rbr | Stanje | Posledice | Verovatnoća | Uticaj | Izloženost | Umanjenje | Kontigencija | Trigeri | Odgovorni |
| 5 | 1 | Korisnici neće biti dostupni 50% od zahtevanog vremena. | Može doći do odlaganja projekta. Mogu se doneti nekorektne odluke. | 80% | 9 | 7,2 | Nabaviti pismeni zahtev za angažovanje ugovorenog korisničkog vremena. | Sortirati pitanja prema prioritetu, koja će biti postavljena korisnicima. Odrediti druge izvore informacija. | Početak projekta | Program menadžer |
| 6 | 2 | Mogući napad radi blokiranja usluga (*denial of service attack* - DoS). | Sistem je nedostupan klijentima i krajnjim korisnicima. | 70% | 10 | 7 | Ispravljanje sigurnosnih propusta, korišćenje *firewall* tehnologije i procedura za ažuriranje (*update*) tehnologija. | Upotreba softvera za detekciju radi zaštite od DoS-a. Razvijati, testirati i redovno pregledati procedure oporavka. | Početak projekta | IT menadžer, menadžer mreže |
| 7 | 3 | Ukoliko prodaja opadne, budžet projekta će biti umanjen za 10,000e ili će biti otkazan. | Otkazivanje projekta, odlaganje ili redukcija granica projekta. | 50% | 10 | 5 | Nisu moguće akcije unutar granica projekta. | Razmotriti odlaganje nekih funkcionalnosti za narednu verziju rešenja. | Kada su poznate izmene budžeta projekta. | Proizvodni menadžer |
| 8 | 4 | Nepotpuni podaci o klijentima dovode do otežane analize. | Nemogućnost identifikovanja najboljih klijenata. | 80% | 6 | 4,8 | Utvrditi opseg problema i razmotriti načine prikupljanja podataka koji nedostaju. | Iščistiti podatke izvlačenjem iz živog sistema. | Kada je donešena odluka o procentu nekompletnih podataka i njihovog uticaja na sistem. | Proizvodni menadžer |
| 9 | 5 | Može se dogoditi da administrator baze podataka ne bude na raspolaganju 50% od ugovorenog vremena. | Projekat bi mogao da se odloži. Mogu biti donešene nekorektne odluke. | 30% | 6 | 1,8 | Obavezati administratora na ugovoreno vreme. | Sortirati pitanja prema prioritetu i ukoliko je moguće odrediti druge izvore informacija. | Početak projekta | Program menadžer |

Slika 5.19. Primer rangiranja rizika prema prioritetu

### 5.4.3. Kategorije izvodljivosti projekta

Generalno, postoje četiri kategorije izvodljivosti:

- **Operativna izvodljivost (*Operational feasibility*)** – meri se prihvatljivost rešenja u organizaciji. Posmatraju se dva aspekta:
  - **Vredi li rešavati problem?** – razmatra se efektivnost rešenja analizom PIECES okvira:
    - ⇨ **Performanse (P*erformance*)** – npr. da li sistem obezbeđuje adekvatno vreme odziva;
    - ⇨ **Informacije (I*nformation*)** – da li sistem pruža korisnicima i menadžerima pravovremene, važne, tačne i korisne informacije;
    - ⇨ **Ekonomija (E*conomy*)** – da li sistem smanjuje troškove poslovanja ili povećava profit;
    - ⇨ **Kontrola (C*ontrol*)** – da li sistem obezbeđuje adekvatnu kontrolu protiv krađa i pronevere i da li garantuje tačnost

i bezbednost podataka i informacija;

- ⇨ **Efikasnost (E*fficiency*)** – da li sistem maksimalno koristi raspoložive resurse uključujući i ljude, vremena, minimalna zakašnjenja i sl.;
- ⇨ **Usluge (S*ervices*)** – da li sistem pruža željene i pouzdane usluge onima kojima su potrebne i da li je sistem fleksibilan i proširljiv.

- ▫ **Kakav je stav korisnika prema rešenju?** – veoma je važno meriti ne samo kako sistem radi već i da li će biti prihvaćen u okruženju. Dešava se da rešenje ne uspe usled otpora koji pružaju korisnici ili menadžment. Treba razmotriti sledeća pitanja:
  - ⇨ Da li menadžment podržava sistem?
  - ⇨ Kako se korisnici osećaju prema novoj ulozi u sistemu?
  - ⇨ Prema čemu korisnici pružaju otpor i da li se može prevazići i kako?
  - ⇨ Kako će se promeniti radno okruženje korisnika? Da li se korisnici mogu prilagoditi promenama?

Analiza izvodljivosti se uglavnom izvodi u ranim fazama životnog ciklusa projekta. Najčešće se analizira na prototipu predloženog sistema i to uglavnom testiranjem korisničkog interfejsa. Meri se koliko korisnici lako mogu da nauče, korste ili kako se podržavaju željeni nivoi produktivnosti korisnika.

- ▪ **Tehnička izvodljivost (*Technical feasibility*)** – meri se primenjivost tehničkog rešenja i raspoloživost tehničkih resursa i ekspertize. Danas je malo toga što je tehnički nemoguće uraditi. Tehnička izvodljivost razmatra sledeće teme:
  - ▫ **Procena mogućih rešenja i alternativa** – procena stanja na tržištu opreme, postojećih rešenja u drugim organizacijama, primenjivosti različitih tehnologija i sl.;
  - ▫ **Primenjivost rešenja, tehnologije** – može li se tehnologija jednostavno primeniti. Pojedina okruženja zagovaraju upotrebu najsavremenije tehnologije, dok je većina sklona zreloj i dokazanoj tehnologiji zbog sigurnosti i bolje podrške;
  - ▫ **Raspoloživost tehnologije** – može li se nabaviti i u kojoj meri je treba prilagoditi ili doraditi;

- **Stručnost** – postoji li neophodna stručnost za primenu odgovarajuće tehnologije u organizaciji.

- **Izvodljivost rasporeda ili vremenska izvodljivost (*Schedule feasibility*)** – procenjuje se da li će projekat biti završen u dogovoreno vreme. Procenu vrše svi učesnici u razvoju rešenja, a naročito developeri koji direktno rade na funkcionalnosti rešenja. Proizvodni menadžer će, nakon razgovora sa svim učesnicima i analize mogućih rizika, brižljivo odrediti krajnji rok završetka projekta. Bolje je isporučiti ispravan sistem dva meseca kasnije, nego neispravan ili beskoristan na vreme!
- **Ekonomska izvodljivost (*Economic feasibility*)** – je troškovno-efektivna analiza projekta koja odgovara na pitanje da li se isplati razvijati rešenje. Troškovi podpadaju u dve kategorije. Postoje troškovi koji se javljaju tokom razvoja rešenja (troškovi razvoja sistema) i oni koji se odnose na upotrebu rešenja (operativni troškovi). Troškovi razvoja informacionog sistema se mogu klasifikovati prema fazama u kojima se javljaju. Mnoge organizacije imaju standardne kategorije troškova, kao na primer:
  - **Troškovi osoblja** – uključuje plate sistem analitičara, programera, konsultanata, sekretarica i svih onih koji rade na projektu. Njihove plate se mogu odrediti prema vremenu provedenog na projektu;
  - **Upotreba računara** – troškovi vezanih za održavanje računara, testiranje, hostovanje i sl. Ukoliko računski centar ili provajder naplaćuju posebno svoje usluge;
  - **Obučavanje** – ukoliko korisnici moraju da se obuče, onda i troškovi oko održavanju kurseva moraju da se uračunaju;
  - **Troškovi opreme i nabavke;**
  - **Troškovi softvera (licence).**

Operativni troškovi u toku životnog veka rešenja se mogu klasifikovati u fiksne i varijabilne troškove. Fiksni troškovi su troškovi koji ostaju nepromenjeni bez obzira na obim poslovanja. Na primer, plate i honorari osoblja, licence softvera, troškovi nabavke i zakupljivanja hardvera, uređaja, poslovnog prostora, obučavanje korisnika i sl. Varijabilni troškovi su troškovi koji se menjaju srazmerno obimu poslovanja. Na primer, troškovi nabavke DVD kaseta, papira za štampu, troškovi struje, telefona, održavanja i sl.

Popularne tehnike za ocenu ekonomske izvodljivosti su: analiza isplativosti, povratak investicija i neto sadašnja vrednost.

**Analiza isplativosti (*Payback analysis*)** je jednostavan metod za određivanje da li će i kada investicija isplatiti samu sebe. S obzirom da se troškovi razvoja dešavaju mnogo pre nego što dobit počne da se ostvaruje, treba sačekati izvesno vreme da dobit počne da prevazilazi troškove. Analiza isplativosti određuje koliko vremena treba da prođe da bi dobit sustigla troškove. Taj period se naziva period isplativosti.

Na slici 5.20 je prikazana analiza izplativosti za projekat razvoja informacionog sistema čiji troškovi razvoja iznose €418.040. Za narednih šest godina procenjeni su neto operativni troškovi i neto dobit. Da bi sagledali period isplativosti, prvo treba prilagoditi troškove i dobit prema trenutnoj vrednosti novca (u ovom primeru evra). Sadašnja vrednost evra u godini *n* će zavisiti od diskontne stope. Diskontna stopa je procentna vrednost slična kamati koja se dobija na račun ušteđevine. U našem primeru, recimo da je diskontna stopa 12%. Sadašnja vrednost evra, za bilo koji naredni period, izračunava se prema sledećoj formuli:

$$PVn = 1/(1+i)^n$$

gde je:

*PVn* – sadašnja vrednost (*present value*) od 1€ godišnje

*i* – diskontna stopa (*discount rate*)

Sadašnja vrednost evra za dve godine će biti:

$$PV_2 = 1/(1+0.12)^2 = 0.797$$

Da bi se izračunala sadašnja vrednost troškova ili dobiti u drugoj godini, treba samo pomnožiti sa 0.797. Na primer, procenjeni operativni troškovi u drugoj godini su €16.000 * 0,797 odnosno €12.752. Nakon izračunavanja svih diskontovanih troškova i koristi, analiza isplativosti je završena. Ukoliko se pogledaju ukupne vrednosti, može se uočiti da se troškovi, tokom šest godina, postepeno uvećavaju, dok se dobit uvećava mnogo brže. S obzirom da će dobit prevazići troškove između treće i četvrte godine, postavlja se pitanje da li je ulaganje u informacioni sistem dobra ili loša investicija. Mnoge kompanije imaju određeni period isplativosti za

sve investicije. Ukoliko pretpostavimo da sve investicije moraju da imaju period isplativosti manje ili jednako četiri godine, onda je investicija u našem primeru dobra.

| Opis cash flow (u evrima) | Godina 0 | Godina 1 | Godina 2 | Godina 3 | Godina 4 | Godina 5 | Godina 6 |
|---|---|---|---|---|---|---|---|
| Trošak razvoja: | 418.040 | | | | | | |
| Troškovi rukovanja i održavanja: | | 15000 | 16000 | 17000 | 18000 | 19000 | 20000 |
| Faktor diskontne stope za 12%: | 1,000 | 0,893 | 0,797 | 0,712 | 0,636 | 0,567 | 0,507 |
| Broj godina: | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Troškovi prema sadašnjoj vrednosti: | 418.040 | 13.393 | 12.755 | 12.100 | 11.439 | 10.781 | 10.133 |
| Kumulativni troškovi tokom životnog veka: | 418.040 | 431.433 | 444.188 | 456.288 | 467.728 | 478.509 | 488.641 |
| | | | | | | | |
| | | | | | | | |
| Dobit od upotrebe novog sistema: | 0 | 150.000 | 170.000 | 190.000 | 210.000 | 230.000 | 250.000 |
| Faktor diskontne stope za 12%: | 1,000 | 0,893 | 0,797 | 0,712 | 0,636 | 0,567 | 0,507 |
| Dobit prema sadašnjoj vrednosti: | 0 | 133.950 | 135.490 | 135.280 | 133.560 | 130.410 | 126.750 |
| Kumulativna dobit tokom životnog veka: | 0 | 133.950 | 269.440 | 404.720 | 538.280 | 668.690 | 795.440 |
| | | | | | | | |
| Ukupno (Troškovi - Dobit): | 418.040 | 297.483 | 174.748 | 51.568 | -70.552 | -190.181 | -306.799 |

Slika 5.20. Analiza isplativosti projekta

**Povratak investicija (*Return on investment* – ROI)** je tehnika koja poredi životni vek profitabilnosti alternativnih rešenja ili projekta. ROI za rešenje ili projekat je procentualna stopa koja meri odnos između iznosa koji posao dobija od investicije i investiranog iznosa. Kraće, ROI je procenat relativnog iznosa koristi projekta. Životni vek ROI za potencijalno rešenje ili projekat se izračunava prema sledećoj formuli:

$$ROI = \frac{(Procenjena\ \ ukupna\ \ dobit - Procenjeni\ \ ukupni\ \ troškovi)}{Procenjeni\ \ ukupni\ \ troškovi}$$

Svi troškovi i dobit treba da budu diskontovani za period od šest godina. Kumulativni troškovi i dobit su prikazani u redovima 6 i 12 na slici 5.20. Procenjena ukupna dobit manje procenjeni ukupni troškovi su jednaki:

€795.440 - €488.692 = €306.748

Stoga, ROI iznosi:

ROI = €306.748/488.692=0,628 = 63%

Treba napomenuti da ovo nije godišnji ROI već ROI tokom životnog veka rešenja ili projekta. Ukoliko podelimo 63% sa 6 godina dobićemo da

prosečan ROI iznosi 10,5% godišnje. Ono rešenje koje ima najveći ROI je najbolja alternativa. Kao i kod analize isplativosti, kompanija može da postavi minimalni prihvatljivi ROI za sve investicije. Nizak ROI (manji od 10% godišnje) može pokazivati da je dobit preniska da bi bila isplativa.

**Neto sadašnja vrednost (Net present value)** jedne investicije se smatra jednom od prioritetnijih tehnika analize troškova i dobiti. Kao i kod prethodnih analiza i ovde se određuju troškovi i dobit za svaku godinu životnog veka sistema, koji su prethodno usklađeni prema trenutnoj vrednosti evra. Na slici 5.21 se prikazuje tehnika neto sadašnje vrednosti. Troškovi su predstavljeni kao negativni tok novca (*cash flow*), dok je dobit prikazana pozitivnim tokom novca. Diskontna stopa za nultu godinu je 1 jer je sadašnja vrednost evra u nultoj godini tačno 1€. Nakon diskontovanja svih troškova i dobiti, treba odbiti sumu diskontovanih troškova od sume diskontovane dobiti da bi se odredila neto sadašnja vrednost. Ukoliko je pozitivna, investicija je dobra, a ukoliko je negativna, onda je loša. Kada se poredi više rešenja ili projekata, onda onaj sa najvećom pozitivnom neto sadašnjom vrednošću predstavlja najbolju investiciju. Na našem primeru, neto sadašnja vrednost iznosi €306.748, što znači da ukoliko investiramo €306.748 sa 12% na šest godina, imaćemo isti profit kao i kad bi implementirali konkretno rešenje IS.

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | **Opis cash flow (u evrima)** | **Godina 0** | **Godina 1** | **Godina 2** | **Godina 3** | **Godina 4** | **Godina 5** | **Godina 6** | **Ukupno** |
| 2 | **Troškak razvoja:** | (418.040) | | | | | | | |
| 3 | **Troškovi rukovanja i održavanja:** | | (15.045) | (16.000) | (17.000) | (18.000) | (19.000) | (20.000) | |
| 4 | **Faktor diskontne stope za 12%:** | 1,000 | 0,893 | 0,797 | 0,712 | 0,636 | 0,567 | 0,507 | |
| 5 | **Sadašnja vrednost godišnjih troškova:** | (418.040) | (13.435) | (12.752) | (12.104) | (11.448) | (10.773) | (10.140) | |
| 6 | **Ukupna sadašnja vrednost troškova u toku životnog veka projekta:** | | | | | | | | (488.692) |
| 7 | | | | | | | | | |
| 8 | | | | | | | | | |
| 9 | **Dobit od upotrebe novog sistema:** | 0 | 150.000 | 170.000 | 190.000 | 210.000 | 230.000 | 250.000 | |
| 10 | **Faktor diskontne stope za 12%:** | 1,000 | 0,893 | 0,797 | 0,712 | 0,636 | 0,567 | 0,507 | |
| 11 | **Sadašnja vrednost godišnje dobiti:** | 0 | 133.950 | 135.490 | 135.280 | 133.560 | 130.410 | 126.750 | |
| 12 | **Ukupna sadašnja vrednost dobiti u toku životnog veka projekta:** | | | | | | | | **795.440** |
| 13 | | | | | | | | | |
| 14 | **NETO SADAŠNJA VREDNOST OVE ALTERNATIVE:** | | | | | | | | **306.748** |

Slika 5.21. Analiza neto sadašnje vrednosti za jedan projekat

## 5.5. Kreiranje funkcionalne specifikacije poslovnog rešenja

Funkcionalna specifikacija je virtuelni repozitorijum projekta koji sadrži rezultate aktivnosti projektovanja dobijenih iz konceptnog, logičkog i fizičkog dizajna. Ti rezultati su uglavnom predstavljeni UML modelima kao što su dijagrami slučajeva korišćenja, scenarija upotrebe i drugi različiti informacioni modeli. Funkcionalna specifikacija je po prirodi virtuelna jer je uglavnom u elektronskoj formi, skladištena u bazama podataka različitih alata. Međutim, ona može biti i u papirnoj formi, u vidu tekstualnog dokumenta ili grafike ili predstavljena kao prezentacija. Funkcionalna specifikacija nije rezultat rada jedne osobe, već nastaje zajedničkim naporom različitih uloga u timu.

Funkcionalna specifikacija opisuje domen trenutne verzije rešenja listajući funkcionalnosti koje će biti deo rešenja. Isključeni delovi će ostati u dokumentu vizija/domen kao potencijalne karakteristike buduće verzije rešenja ili kao želje ili kao delovi koji će biti van granica sistema. Funkcionalna specifikacija se koristi radi evidentiranja donešenih odluka i dogovora vezanih za funkcionalnosti rešenja, njegovih interfejsa, ciljeva i prioriteta. Komunikacija sa razvojnim timom se upravo odvija preko funkcionalne specifikacije. Tim za testiranje takođe koristi funkcionalnu specifikaciju kako bi kreirao planove testiranja.

Ne postoje konačne tačke početka i završetka kreiranja funkcionalne specifikacije. Tokom faze planiranja, svaki od procesa projektovanja – konceptni, logički i fizički – doprinosi različitim elementima funkcionalne specifikacije. Sa završetkom svakog zadatka tokom faze planiranja, funkcionalna specifikacija postaje sve kompletnija.

Neki od ciljeva funkcionalne specifikacije su:

- **Konsolidovanje zajedničkog shvatanja poslovnih i korisničkih zahteva.** Funkcionalnost rešenja će zavisiti od poslovnih i korisničkih zahteva. Kod većih i složenijih projekata, broj i složenost zahteva se povećava. Funkcionalna specifikacija pomaže da se klijenti i projektni tim slože oko zahteva koje buduće rešenje mora da ispuni;
- **Razlaganje problema i logičko modelovanje rešenja**. Za složenije probleme je veoma važno da tim jasno identifikuje sve delove problema. Takođe tim treba da razbije rešenje na posebne, nedvosmislene delove. Funkcionalna specifikacija pomaže timu da

pojednostavi rešenje na logičke delove i da ih dokumentuje. Takođe, pomaže da tim izmeni dizajn rešenja u ranim fazama razvoja, čime se izbegavaju rizici i dodatni troškovi koji bi se pojavili u kasnijim fazama projekta.

- **Obezbeđuje strukturu za planiranje, pravljenje vremenskog rasporeda i razvijanje rešenja**. Funkcionalna specifikacija pomaže program menadžeru u identifikovanju timskih zadataka i kreiranju procena troškova i budžeta za čitav projekat. Program menadžer može da oceni resurse i vremena projekta i da kreira projektne planove i rasporede. Tim za testiranje koristi funkcionalnu specifikaciju da bi kreirao test scenarije u ranim fazama životnog ciklusa projekta. Menadžment puštanja rešenja u rad koristi funkcionalnu specifikaciju za uvođenje i implementiranje rešenja i za podršku okruženja za razvoj i testiranje.
- **Služi kao ugovor između tima i klijenta za ono što će biti isporučeno**. U većini organizacija i na većini projekata, funkcionalna specifikacija služi kao ugovor između tima i klijenta, to je pisani dokaz o tome šta će biti razvijeno i isporučeno. Funkcionalna specifikacija može poslužiti i kao zakonski dokument.

Ponekad ograničenja kao što su budžet i vreme mogu da odvrate tim od pisanja ili korišćenja funkcionalne specifikacije. Tim može direktno da ide na fazu razvoja bez pisanja funkcionalne specifikacije, ali tada se rizici projekta znatno povećavaju. Neki od rizika ukoliko se ne piše funkcionalna specifikacija su:

- tim može da razvije rešenje koje upotpunosti ne ispunjava korisničke zahteve;
- tim ne može biti potpuno siguran da je razvio željeno rešenje, jer je u nemogućnosti da jasno definiše očekivanja klijenata;
- tim neće imati dovoljno informacija o tome da li rešenje zadovoljava željeni kvalitet;
- projektni menadžer neće biti u mogućnosti da tačno proceni budžet i raspored projekta. Informacije unutar funkcionalne specifikacije pomažu timu da proceni neophodne veštine za razvoj rešenja.

Mogući elementi funkcionalne specifikacije su prikazani na tabeli 5.2. Svaki element može da predstavlja poseban dokument (Tabela 5.2):

- **Pregled konceptnog dizajna** – uključuje informacije kao što su pregled rešenja i arhitektura rešenja. Prikazuju se slučajevi korišćenja, scenarija upotrebe i kontekstualni modeli.
- **Pregled logičkog dizajna** – uključuje informacije kao što su korisnici, objekti i atributi. Prikazuju se modeli sekvence zadataka, logički modeli objekata i usluga, konceptni modeli predloženog rešenja, korisnički interfejsi, logički model baze podataka i arhitektura sistema.
- **Pregled fizičkog dizajna** – uključuje informacije o aplikaciji i infrastrukturi. Prikazuju se topologija distribuiranih komponenti, pakovanje komponenti, smernice za korišćenje tehnologije, arhitektura infrastrukture i dizajn, opis korisničkih ekrana i fizički model baze podataka.
- **Standardi i procesi** – uključuje informacije o standardima i procesima koje tim koristi za obavljanje različitih zadataka na projektu. Uključuje i detaljne informacije o kvalitetu i performansama rešenja.

Tabela 5.2. Elementi funkcionalne specifikacije

| Elementi | Sadržaj |
|---|---|
| Pregled konceptnog dizajna | Slučajevi korišćenja, scenarija upotrebe, kontekstni modeli npr., ekranski prikazi. |
| Pregled logičkog dizajna | Modeli sekvence zadataka, logički modeli objekata i usluga, konceptni modeli predloženog rešenja, UI ekrani, logički model baze podataka, sistemska arhitektura. |
| Pregled fizičkog dizajna | Topologija distribucije komponenti, infrastruktura arhitekture i dizajna, opis UI ekrana, fizički model baze podataka ... |
| Standardi i procesi | Sigurnost, pakovanje, održavanje, podrška, stabilizacija, uvođenje |

### 5.5.1. Faza planiranja prema MSF-u

Tokom faze definisanja vizije, projektni tim je prikupio dovoljno informacija za početak projekta. Verifikovanjem dokumenta vizija/domen, tim prelazi na fazu planiranja. Tokom ove faze se planira kako će rešenje dalje da se razvija i određuju se neophodni resursi za njihov razvoj. U okviru faze planiranja razvija se konceptni, logički i fizički dizajn rešenja. Ova

tri procesa nisu paralelna, već će logički dizajn zavisiti od konceptnog, a fizički od logičkog dizajna. Bilo koja promena na konceptnom rešenju uticaće na logički, a potom i na fizički dizajn.

U toku faze planiranja pravi se niz modela i dokumenata koji zajedno čine funkcionalnu specifikaciju rešenja. Postoje brojne tehnike modelovanja poslovnih procesa i ključnih aktivnosti u poslovanju. Svaka uloga (rola) projektnog tima ima različite odgovornosti tokom faze planiranja:

- **Proizvodni menadžment** – vodi računa da plan ispuni korisničke zahteve. Ova uloga je zadužena za analizu zahteva i tekućeg stanja poslovanja, optimizovanje koncept rešenja i kreiranje konceptnog (idejnog) rešenja.
- **Program menadžment** – osigurava da tim ima sve neophodne resurse za izvođenje projektnog plana. Ova rola je odgovorna za sveukupni dizajn sa naglaskom na logički dizajn i funkcionalnu specifikaciju. Program menadžer kreira projektne planove i vremenske rasporede i odgovoran je za završetak faze planiranja.
- **Razvojni tim** – obezbeđuje da plan bude tehnički izvodljiv. Developeri su odogovorni za kreiranje logičkog i fizičkog dizajna rešenja i određivanje vremena i resursa neophodnih za razvoj i stabilizaciju rešenja.
- **Testiranje** – odgovorni su za procenu da li se funkcionalnosti rešenja mogu tesitrati i ujedno obezbeđuju plan i raspred za njihovo testiranje.
- **Menadžment puštanja rešenja u rad** – procenjuju da li se rešenje može lako implementirati, upravljati i podržavati. Ova rola planira i pravi raspored uvođenja rešenja.
- **Iskustveni korisnik** – osigurava da će korisnici moći da koriste rešenje. Ova rola je odgovorna za analiziranje korisničkih zahteva, kreiranje strategije podrške performansi, lokalizaciju i ocenjivanje upotrebljivosti rešenja. Takođe, ocenjuju neophodno vreme i resurse potrebnih za razvoj sistema za podršku korisnicima (priručnici, obuka i dr.).

Faza planiranja se završava kada se odobri projektni plan. Kritične tačke faze planiranja su:

- **Funkcionalna specifikacija (*Functional specification*)** – definiše šta će se razvijati i kako i kada će rešenje biti razvijeno.

- **Glavni projektni plan (*Master project plan*)** – kolekcija planova koji ukazuju na zadatke koje tim treba da obavi kako bi dostigao funkcionalnost opisanu u funkcionalnoj specifikaciji.
- **Glavni raspored projekta (*Master project schedule*)** – određuju se vremenski okviri za glavni projektni plan. Ovaj dokument predstavlja prvi korak u određivanju tačnog datuma isporuke rešenja.
- **Ažurirani glavni dokument ocene rizika (*Updated master risk assessment document*)** – dokument ocene rizika koji je razvijen u toku faze definisanja vizije se ponovo pregleda i ažurira po kritičnim tačkama.

### 5.5.2. Konceptni dizajn

Konceptni dizajn (idejni projekat) je proces prikupljanja, analiziranja i postavljanja prioriteta na poslovne i korisničke poglede na probleme i rešenje. Kod konceptnog dizajna, projektni tim pokušava da shvati kontekst problema, zabeleži poslovne aktivnosti i odredi granice i međusobne veze na projektu. U okviru konceptnog dizajna se modeliraju zahtevi i kreira prezentacija rešenja visokog nivoa. Konceptni dizajn počinje da se kreira još u toku pisanja dokumenta vizija/domen. Bez konceptnog dizajna, može se desiti da se kreira veliko rešenje, ali na pogrešnom problemu.

Konceptni dizajn je jedan iterativan proces koji sledi sledeće korake:

- **Istraživanje** – identifikuju se ključni poslovni procesi i aktivnosti, vrednuju, prerađuju i proširuju zahtevi, slučajevi korišćenja i scenarija upotrebe, kreiranih tokom faze definisanja vizije.
- **Analiza** – dokumentuju se i modeliraju tokovi posla (*workflow*), sekvence zadataka i veze u okruženju.
- **Optimizacija** – optimizuje se konceptno rešenje i vrednuje i testiraju poboljšani poslovni procesi.

U toku **istraživanja**, veoma je važno da projektni tim razume razliku između različitih kategorija zahteva: korisnički, sistemski, operacioni i poslovni. U toku faze definisanja vizije, projektni tim je na bazi intervjua identifikovao grube zahteve (Tabela 5.3). U toku analize vrši se prečišćavanje tih zahteva (Tabela 5.4). Zahtevi moraju biti dobro definisani, koncizni, organizovani u hijerarhiju povezanih zahteva, pisani poslovnim jezikom da bi mogli biti testirani.

Tabela 5.3. Zahtevi izvučeni iz faze definisanja vizije

| ID | Opis zahteva |
|---|---|
| 1 | Identifikovanje najboljih klijenata prema proizvodu i lokaciji. |
| 2 | Identifikovati pad prodaje. |
| 3 | Identifikovati najbolje klijente. |

Tabela 5.4. Redefinisani zahtevi

| ID | Opis zahteva |
|---|---|
| 1.1 | Mogućnost analize podataka o klijentima po proizvodima, klijentima i regionima. |
| 1.1.1 | Mogućnost analize nivoa profita po proizvodima. |
| 1.1.2 | Mogućnost analize nivoa profita po klijentima. |
| 1.1.3 | Mogućnost analize nivoa profita po regionima. |
| 1.2 | Mogućnost sortiranja klijenata (u opadajućem ili rastućem redosledu). |
| 1.2.1 | Mogućnost sortiranja klijenata prema iznosu prodaje. |
| 1.2.2 | Mogućnost sortiranja klijenata prema iznosu prodaje određenog proizvoda. |
| 1.2.3 | Mogućnost sortiranja klijenata prema iznosu prodaje po regionima i iznosu prodaje za određeni period vremena. |
| 1.3 | Mogućnosti identifikovanja trendova prodaje. |
| 1.3.1 | Mogućnost identifikovanja pada prodaje. |
| 1.3.2 | Mogućnost identifikovanja opadanja prodaje kod klijenata. |

Tokom **analize** razmišlja se i o mogućnosti projektovanja želja korisnika. Zadaci koji se izvršavaju u toku analize su:

- **sintetizovanje informacija** – proces transformacije prikupljenih podataka u značajne informacije. Tim identifikuje informacije o tome šta su korisnici rekli i uradili, beleže detaljne tokove zadataka koje izvršavaju korisnici, identifikuju alate koji se koriste, izuzetke i alternativne sekvence zadataka, modeluju veze između poslovnih procesa, poslovnih sistema i korisnika, modeluju trenutno okruženje u kojem korisnici rade i eventualne promene koje se mogu desiti u postojećem okruženju;
- **poboljšavanje dijagrama slučajeva korišćenja** – slučajevi korišćenja koji su identifikovani u toku faze definisanja vizije prikazuju

ključne slučajeve korišćenja u sistemu, definišu domen rešenja i obezbeđuju osnovu za konceptno rešenje. Poboljšanje tih dijagrama podrazumeva kreiranje detaljnih slučajeva korišćenja i odgovarajućih scenarija upotrebe za svaki pojedinačan dijagram. Zatim, svaki dijagram slučajeva korišćenja i scenarija se upoređuju sa intervjuom i ostalom dokumentacijom i na kraju verifikuju sa korisnicima.

- **selektovanje odgovarajuće arhitekture aplikacije** - cilj konceptnog dizajna je konceptni model rešenja. Da bi mogli da kreiramo konceptni model, neophodno je razumeti servise koje rešenje mora da obezbedi. Servisi se definišu kao jedinice aplikacione logike koje uključuju metode za implementaciju operacija, funkcionisanje ili transformisanje. Servisi su pridruženi akcijama i koriste se za implementaciju poslovnih pravila, rukovanje podacima i omogućavaju akcije kao što su dodavanje, pregledanje, modifikovanje podataka i dr. Klijente ne zanima kako su servisi implementirani, već da li mogu da izvrše željenu akciju. Tipični servisi koje rešenje obezbeđuje su:
  - **korisnički servisi** (*user services*) – obezbeđuju korisnički interfejs u jednoj aplikaciji. Korisnički servisi jedne aplikacije omogućavaju interakciju između aplikacije i njegovih korisnika.
  - **poslovni servisi** (*business services*) – obezbeđuju da se poslovna pravila izvršavaju prema tačnoj određenoj sekvenci. Poslovni servisi kriju logiku implementacije poslovnih pravila i transformišu podatke iz korisničkih servisa, poslovnih i servisa podataka.
  - **servisi podataka** (*data services*) – obezbeđuju najmanje vidljiv nivo detalja za rukovanje sa podacima. Servisi podataka se koriste da bi se implementirala poslovna šema nad uskladištenim podacima koje koristi aplikacija.
  - **sistemski servisi** (*system services*) – obezbeđuju funkcionalnost van poslovne logike. Opšti sistemski servisi su *backup* servisi, servisi za rukovanje sa greškama, bezbednosni servisi, servisi za rad sa porukama.

Na tabeli 5.5. se prikazuju različiti servisi na primeru jedne aplikacije za obradu porudžbine. Treba istaći da se servisi klasifikuju prema funkcijama, a ne prema lokacijama.

Tabela 5.5. Servisi na primeru aplikacije za obradu porudžbine

| Servisi | Primeri |
|---|---|
| Korisnički servisi | ▫ prikazivanje porudžbine<br>▫ prikazivanje računa klijenta<br>▫ prikazivanje informacija o proizvodu |
| Poslovni servisi | ▫ popunjavanje porudžbine<br>▫ ažuriranje informacija na računu klijenta<br>▫ izvlačenje informacija o proizvodu<br>▫ provera zaliha proizvoda |
| Servisi podataka | ▫ informacije porudžbine<br>▫ račun klijenta<br>▫ informacije o proizvodu |

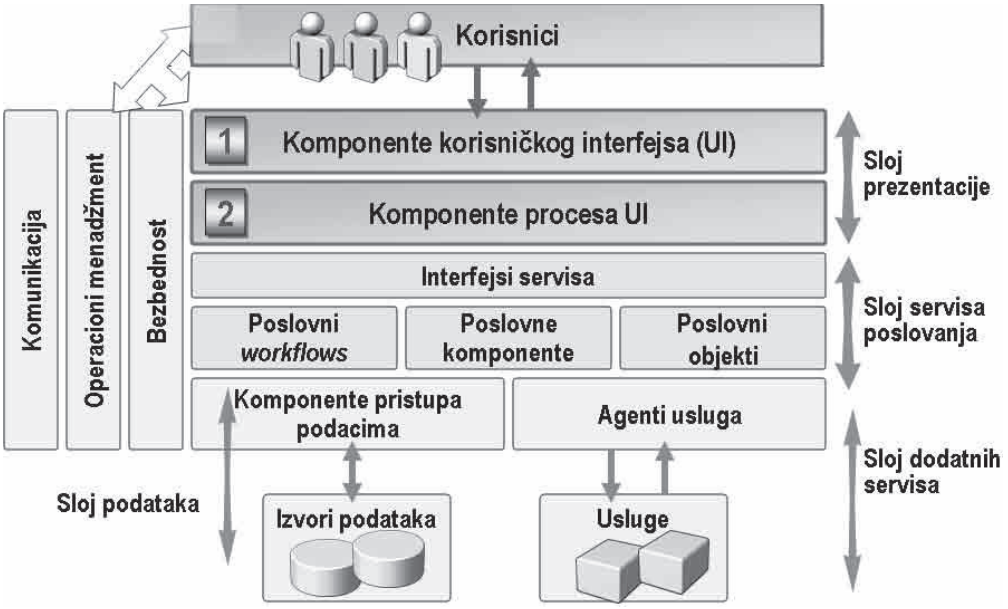
Servisi su organizovani prema arhitekturi aplikacije. Arhitektura aplikacije se sastoji od definicija, pravila i relacija koje formira struktura jedne aplikacije. Ona pokazuje kako je aplikacija struktuirana, ali ne i kako je implementirana. Fokusira se na rešenje, a ne na tehnologije koje će se koristiti za implementaciju rešenja. Da bi mogao da se razvije konceptualni model rešenja, treba izabrati aplikacioni model i kandidate aplikacione arhitekture koje će rešenje koristiti. Projektni tim bira arhitekturu zasnovanu na servisima koje rešenje mora da obezbedi i na korisničkim očekivanjima o performansama tog rešenja. Pretpostavke i ograničenja utiču na izbor arhitekture rešenja. Pretpostavke podrazumevaju, na primer, operativni sistem koji se koristi u sistemu. Ograničenja mogu biti raspoloživi budžet, vreme, veštine i resursi neophodni da bi se završio projekat. Arhitekture aplikacija koje se koriste su:

- **klijent/server arhitektura (*client/server*)** - klijent započinje sesiju sa serverom i kontroliše sesiju. Kada primi zahtev, server izvršava zahtevanu operaciju i vraća rezultat klijentu. Ograničenje ove arhitekture je velika zavisnost klijenta od servera.
- **slojevita arhitektura (*layered*)** – je proširena verzija klijent/server arhitekture i sastavljena je od hijerarhijskih slojeva. Različiti servisi u aplikaciji su jasno pozicionirani na određene

slojeve tako da servisi mogu međusobno da komuniciraju jedino ukoliko se nalaze na susednim slojevima. Slojevi enkapsuliraju servise i štite servise jedan od drugih obezbeđujući jednostavni set interfejsa za deljene resurse. Servisi prezentacije, poslovni i servis podataka su različiti slojevi u slojevitoj arhitekturi. Prednosti su poboljšani sistem skalabilnosti i visoka bezbednost.

- ***stateless* arhitektura** - je verzija klijent/server ili slojevite arhitekture gde svaki klijentski zahtev sadrži sve informacije koje zahteva server kako bi razumeo i obradio zahtev. Nijedna informacija se ne skladišti kod klijenta.
- ***cache* arhitektura** - kešing je drugi pristup u kojem aplikacija obezbeđuje način za obradu klijentskog zahteva bez prosleđivanja (*forward*) tog zahteva drugim mašinama. Da bi se implementirala ova arhitektura, treba identifikovati šta se može, a šta ne keširati, takođe treba definisati i načine upravljanja vremenom artikala u kešu.
- **hibridna arhitektura (*layered-client-cache-stateless-cache-server*)** – omogućava kreiranje fleksibilne i skalabilne aplikacije.

- **kreiranje konceptnog modela rešenja** – primer konceptnog modela je prikazan na slici 5.22.




Slika 5.22. Primer konceptnog modela rešenja [9]

Finalni korak konceptnog dizajna je **optimizacija**. Tokom optimizacije počinje da se projektuje rešenje. Projektovanje budućeg stanja sistema započinje tako što se ispituju scenarija koja odslikavaju trenutna stanja sistema i eliminišu neefikasnosti, uska grla i redundantnosti. Treba napomenuti da se reinženjering poslovnih procesa radi samo za one aktivnosti koje se nalaze u okviru domena projekta. Koraci kod opisivanja budućeg stanja sistema su:

- Definisanje vizije budućeg stanja – podrazumeva poboljšanje produktivnosti trenutnog stanja, razmatranje potreba u odnosu na želje, uravnoteženje poslovnih i korisničkih zahteva i razmotranje tehničke izvodljivosti rešenja.
- Redizajniranje trenutnih procesa - kako bi se optimalno podržale ključne poslovne aktivnosti i procesi.
  - Optimizovati čitav proces.
  - Integrisati procese gde je to moguće.
  - Eliminisati neefikasnosti, uska grla i redundatnosti.
  - Razviti scenarija upotrebe budućeg stanja koji odslikavaju redizajnirane procese.
  - Vrednovati scenarija budućeg stanja sa korisnicima.
  - Ponoviti reinženjering procesa ukoliko je neophodno.

Redizajniranje rešenja je moguće uraditi i iz perspektive korisnika. Naime, mogu se uočiti mogućnosti poboljšanja kod *workflow* aktivnosti, neproduktivne poslovne politike, birokratije, propuštenih komunikacija i uloga koje ne dodaju vrednost poslovanju i ometaju sveukupnu efikasnost poslovnih procesa. Treba imati na umu sledeće:

- identifikovati neželjene aktivnosti, uska grla i nepotrebne korake u procesu;
- otkloniti redundantnost u trenutnom okruženju;
- integrisati individualne funkcionalne informacione sisteme;
- identifikovati i otkloniti sav nepotreban paprini posao;
- odrediti minimalni nivo performansi koji se mora dostići u procesu;

- otkloniti vremena čekanja u sistemu identifikovanjem koraka koji se mogu simultano izvršavati umesto sekvencionalno;
- poboljšati sistem tako što će se obezbediti efikasan *feedback* korisnicima, npr. o performansama kako bi se osigurali da su problemi trenutno rešeni.

Cilj kod reinženjeringa poslovnih aktivnosti je da se rešenje projektuje sa najmanje napora. Ne postoji jedinstven metod za efikasno redizajniranje procesa. Tim može da obavlja *brainstorming* sesije. Često problem ima nekoliko alternativa njegovog rešavanja. Tokom reinženjeringa treba razmotriti sve elemente procesa – inpute, autpute, nivoe performansi, resurse, procedure kontrole, produktivnost i vreme. U toku optimizacije procesa treba obratiti pažnju na sledeće:

- Uskladiti procese prema ciljevima performansi.
- Dizajnirati procese oko proizvoda i usluga.
- Zameniti hijerarhiju sa samo-organizovanim timovima koji paralelno rade.
- Poboljšati produktivnost integracijom zadataka.
- Odrediti gde se može upotrebiti tehnologija koja će podržati redizajnirane procese.
- Analizirati komponente procesa i dr.

Razmotrimo sledeći primer. Ukoliko je u sistemu uočeno da magacioner popunjava optremnicu i prijemnicu koji su u trenutnom sistemu predstavljeni kao dva posebna objekta, tada se uočava problem ažuriranja kod dupliranih objekata. S obzirom da otpremnica i prijemnica imaju iste atribute, primenom optimizacije, pravi se jedan objekat Dokument sa dodatnim atributom na primer Tip=Ulazna/Izlazna.

U toku procesa optimizacije mogu se uočiti i novi slučajevi korišćenja koje treba inkorporirati u konceptualni model. Takođe, u ovom koraku se vrši i procena troškova i koristi redizajniranog rešenja. Mnogi projekti se u ovoj fazi otkazuju usled toga što evaluacija ukazuje da ROI (*Return On Investment*) novog rešenja ne ispunjava poslovne kriterijume. Validacija je neophodna jer umanjuje rizike, pomaže u pronalaženju propuštenih informacija, usaglašava obostrane interese i predstavlja osnovu za dalje logičko projektovanje rešenja.

### 5.5.3. Logički dizajn

Za razliku od konceptnog projekta koji opisuje rešenje sa aspekta poslovanja i korisnika, logički projekat opisuje rešenje sa aspekta projektnog tima. Dobar logički dizajn će zavisiti od dobrog konceptnog dizajna. Logički dizajn se može definisati kao proces opisivanja rešenja u terminima njegove organizacije, strukture i interakcije njegovih delova sa aspekta projektnog tima. Kada se kreira logički dizajn, tim uzima u obzir sve poslovne, korisničke, sistemske i operacione zahteve koji identifikuju potrebe za bezbednošću, praćenjem, skalabilnošću, rukovanjem greškama, licenciranjem, globalizacijom, arhitekturom aplikacije i integracijom sa drugim sistemima.

Da bi se razumela razlika između konceptnog i logičkog dizajna, razmotrićemo primer projektovanja kuće. Tokom konceptnog dizajna, klijent i arhitekta listaju zahteve klijenata, kao što su okruženje, spavanje, neophodne uređaje i sl. Tokom logičkog dizajna, arhitekta kreira podnu podlogu, kompletno okruženje kuće, vrata, prozore, sobe, krov itd.

Logički dizajn nije tehničko rešenje, već samo određuje poslovne potrebe koje tehnologija mora da podrži. Logički dizajn je nezavisan od fizičke implementacije. Primarni fokus je na identifikovanju onoga što rešenje treba da radi. Projektni tim mora u potpunosti da razume rešenje pre nego što odabere tehnologiju za implementaciju rešenja. Svakako da tim može da identifikuje pretpostavke i ograničenja koji mogu da utiču na izbor određene tehnologije. Logički dizajn ne sme da bude optimizovan prema fizičkom modelu.

Prednosti projektovanja glavnog (logičkog) projekta su:

- Pomaže timu da razume i upravlja kompleksnošću projekta. Mnogi projekti propadaju usled toga što su veoma složeni i što nisu prethodno dobro isprojektovani.
- Verifikuje da dizajn reflektuje poslovne probleme. Pomaže timu u otkrivanju grešaka, nekonzistentnosti kod konceptnog dizajna, eliminisanju redundantnosi u zahtevima i dr.
- Olakšava koordinaciju između višestrukih sistema u organizaciji i obezbeđuje dosledan pogled na čitav projekat.
- Koristi se kao osnova fizičkog dizajna.

U toku logičkog dizajna, svaki član tima ima različite odgovornosti:

- **Proizvodni menadžment** – osigurava da dizajn bude u skladu sa potrebama klijenata i poslovanja.
- **Program menadžment** – definiše projektni plan, što uključuje resurse, rasporede i procenu rizika i odgovoran je za izlaznu dokumentaciju logičkog dizajna.
- **Razvoj** – identifikuje servise i pridružene objekte.
- **Testiranje** – vrednuje modele logičkog dizajna u odnosu na konceptni i definiše preliminarni plan testiranja.
- **Menadžment puštanja rešenja u rad** – procenjuje izvodljivost implementacije rešenja, definiše infrastrukturu i uvođenje rešenja u sistem.
- **Iskustveni korisnik** – definiše korisničke ciljeve za performansama, preporučena rešenja i deriniše preliminarni plan za edukaciju korisnika.

Faza logičkog dizajna se može podeliti u sledeće zadatke:

- **Analiza logičkog dizajna** u okviru koga tim izvršava sledeće zadatke:
  - Preuređivanje liste kandidatnih tehnologija i alata;
  - Identifikovanje poslovnih objekata i servisa;
  - Identifikovanje važnih atributa i ključnih relacija.
- **Optimizacija logičkog dizajna**, u okviru koga tim izvršava sledeće:
  - Poboljšanje logičkog dizajna;
  - Validaciju logičkog dizajna.

### Analiza kandidatnih tehnologija

U toku **analize**, tim razbija problem na manje jedinice tzv. module. Za svaki modul, tim identifikuje objekte, servise, atribute, relacije i kandidatne tehnologije. Tokom logičkog dizajna, tim poboljšava listu kandidatnih tehnologija koje su prethodno identifikovane u fazi konceptnog dizajna.

Kod ocene kandidatnih tehnologija tim uzima u obzir poslovna razmatranja, arhitekturu preduzeća i tehnologiju. Neka od poslovnih razmatranja su:

- Izvodljivost – da li će tehnologija da ispuni poslovne zahteve;
- Proizvodni troškovi – razumevanje kompletnih proizvodnih troškova, koji uključuju troškove servera, licenciranja, *upgrade*-ovanja, developera, inicijalne troškove hardvera i softvera, podrške, infrastrukture, obučavanja i dr.;
- Iskustvo – veliki uticaj mogu imati i korisnička iskustva u korišćenju različitih tehnologija;
- Povratak investicija (*return on investment* - ROI) – svaka investicija mora da bude u skladu sa ROI. Ne treba birati tehnologiju samo zbog toga što je nova. Svako investiranje u nove tehnologije treba da obezbedi odgovarajuću vrednost poslovanju, kao na primer povećanje prihoda ili smanjivanje troškova i dr.;
- Zrelost – Zreli proizvodi su prihvaćeni na tržištu zbog toga što su dobro shvaćeni, stabilni i što se raspolaže sa resursima i znanjima;
- Podrška – u odabiru tehnologije, veoma je važno da postoji podrška u toku razvoja rešenja;
- *Legacy* (nasleđe, ostavština) problem - definiše da IT rešenja koja se koriste u sistemu traže više ulaganja od uvođenja novog IT rešenja. Održavanje više košta nego sam razvoj novog rešenja;
- Lakoća uvođenja rešenja u sistem;
- Konkurentna prednost;
- Sposobnost integracije sa postojećim sistemima i dr.

Projekat mora da bude u skladu sa ciljevima i principima opisanih arhitekturom preduzeća. Arhitektura preduzeća opisuje planove trenutnog i budućeg stanja. Na primer, ukoliko se planira upotreba *Microsoft Windows Server* 2003, onda i izbor tehnologija treba da bude u skladu sa ovim planom. Rešenje mora da se uklapa u ograničenja arhitekture preduzeća. Na primer, nova tehnologija kao što je *streaming* video može da zahteva povećanje prohodnosti mreže. Takve promene ne zahtevaju samo novac za kupovinu novih tehnologija, već i vreme da se nauči i implementira ta tehnologija. Tehnologija mora da funkcioniše i sa drugim sistemima unutar organizacije.

Neke od tehnoloških razmatranja koje treba uzeti u obzir su:

- **Bezbednost** – proceniti bezbednost novog rešenja razmatrajući autentifikaciju, kontrolu pristupa, enkripciju i praćenje. Autentifikacija

određuje da li je pojedinac onaj za koga se smatra, dok autorizacija dozvoljava identifikovanim pojedincima da izvršavaju dozvoljene akcije unutar sistema. Servisi bezbednosti takođe obezbeđuju enkripciju kako bi zaštitili informacije u toku njihovog prenosa ili skladištenja. Praćenje kreira zapise akcija koje pojedinci izvršavaju ili nameravaju da izvrše unutar sistema;

- **Standardi interakcije servisa** – razmatraju se standardi interakcije i platformi. Na primer, Microsoft .NET Fremework pojednostavljuje razvoj aplikacija u distribuiranom okruženju. Pored toga što obezbeđuje višejezične integrisane funkcionalnosti, smanjuje probleme performansi i obezbeđuje jednostavan model za interakciju komponenti.
- **Pristup podacima** – kod odabira usluga pristupa podacima, treba razmotriti performanse, standardizaciju, buduća usmerenja, upravljanje pristupom podacima i različita skladišta podataka.
- **Skladištenje podataka** – sistemi za skladištenje podataka se koriste za skladištenje svih poslovnih informacija tako da prvo treba razmotriti lokaciju skladištenja. Podaci se mogu skladištiti na jednom računaru, serverima ili distribuiranim klijent računarima. Lokacija direktno utiče na performanse aplikacije i način njenog razvoja.
- **Sistemski servisi** – projektni tim procenjuje zahtevane sistemske servise i identifikuje tehnologije koje obezbeđuju tražene servise. Primeri sistemskih servisa su:
  - transakcioni servisi - obezbeđuju mehanizme i okvir (*framework*) za transakcione aplikacije.
  - servisi asinhrone komunikacije – aplikacija prijema garantuje da je poruka stigla na udaljeni sistem.
  - servisi poruka – rutiraju i isporučuju informacije.
- **Alati razvoja** – obezbeđuju integrisana razvojna okruženja, kontrolu izvornog koda, *wizard*-e i biblioteke koda. Ovi alati znatno smanjuju vreme potrebno za razvoj jedne aplikacije. Projektni tim razmatra sledeće faktore:
  - veštine developera - novi alati zahtevaju dodatnu obuku i vremena učenja i direktno utiču na raspored i resurse projekta.
  - angažovanje sa strane tzv. autsorsing (*outsourcing*) spoljnih razvojnih timova.

- izbor različitih jezika za različite zadatke na projektu – na primer, .NET Framework programski model podržava različite programske jezike kako bi developer mogao da odabere one koje zna čime se ubrzava produktivnost developera.
- izbor razvojnih alata utiče na odluke vezane za druge tehnologije – na primer, Microsoft ASP.NET obezbeđuje platformu koja je nezavisna od jezika za kreiranje Web aplikacija, ali sam ASP.NET nije nezavisna platforma. Web aplikacije koje su izgrađene u ASP.NET-u moraju da se pokreću sa Windows operativnog sistema.

- **Operativni sistemi** – servisi koji obezbeđuju operativni sistemi utiču na potrebe skalabilnosti, bezbednosti, kodiranja i sl. Izbor operativnog sistema može da zavisi i od određenih tipova uređaja kao što su na primer: *Handheld* PC, *desktop* računari, serveri ili specijalizovani uređaji namenjeni određenoj industriji.

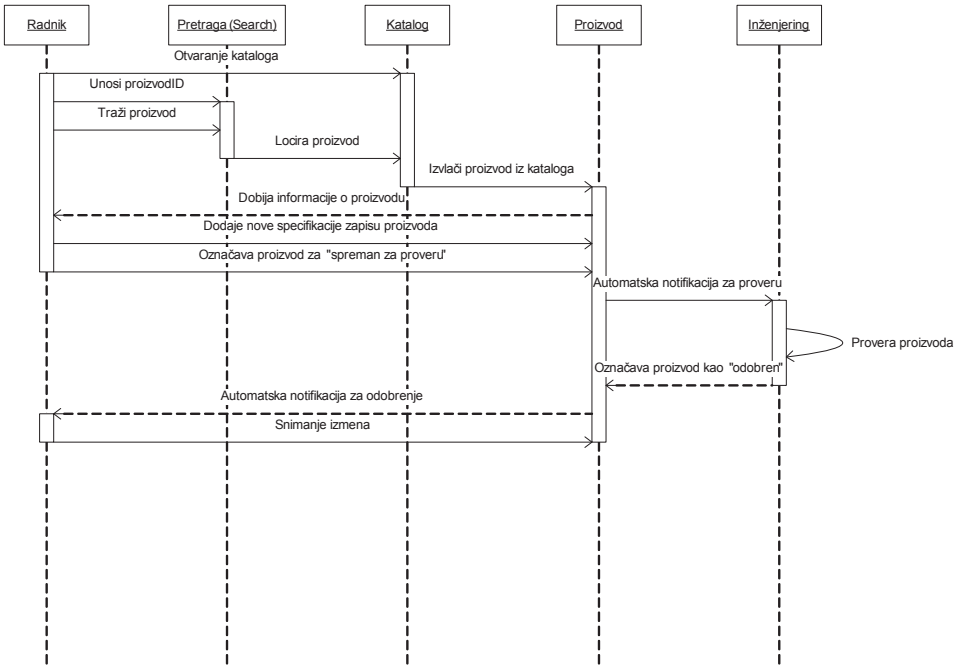
### Dokumentovanje logičkog dizajna

Postoji nekoliko metoda za modelovanje relacija, a samim tim i dokumentovanje izlaznih dijagrama. Jedan od njih je dijagram sekvenci. Dijagram sekvenci (*Sequence diagram*) prikazuje:

- hronološki uređene učesnike i objekte koji učestvuju u jednoj interakciji;
- događaje koje učesnici i objekti generišu;
- tok kontrole i sekvence ponašanja.

Vertikalna linija kod dijagrama sekvenci prikazuje životni vek objekta. Strelice između dva objekta predstavljaju poruke koje se prikazuju prema redosledu njihovog dešavanja. Dijagram sekvenci pomaže da se jasno sagledaju sekvence aktivnosti.

Dijagram na slici 5.23 opisuje scenario gde radnik dodaje nove specifikacije postojećem proizvodu u katalogu. Prvi korak je da radnik pronađe postojeći proizvod. Kada locira proizvod, onda dodaje nove specifikacije proizvodu. Pre nego što radnik doda nove informacije na Web sajt, inženjeri moraju da provere informacije. Ukoliko tim inženjera odobri informaciju, radnik se obaveštava da je informacija o proizvodu dodata na Web sajt.




Slika 5.23. Dijagram sekvenci

Dijagram sekvenci prikazuje redosled poslova (sekvence zadataka), reference na slučajeve korišćenja, učesnike (aktere), rezultate izvođenja, izuzetke (alternative događanja) i dr. Na osnovu dijagrama se može izvršiti optimizacija ukoliko se uoče sekvence zadataka koje se ponavljaju. Takođe se na osnovu dijagrama mogu iscrtati i ekranske mape. Problem kod dijagrama sekvenci je taj što mora da se ide dovoljno duboko da bi se uvidela sekvenca događaja.

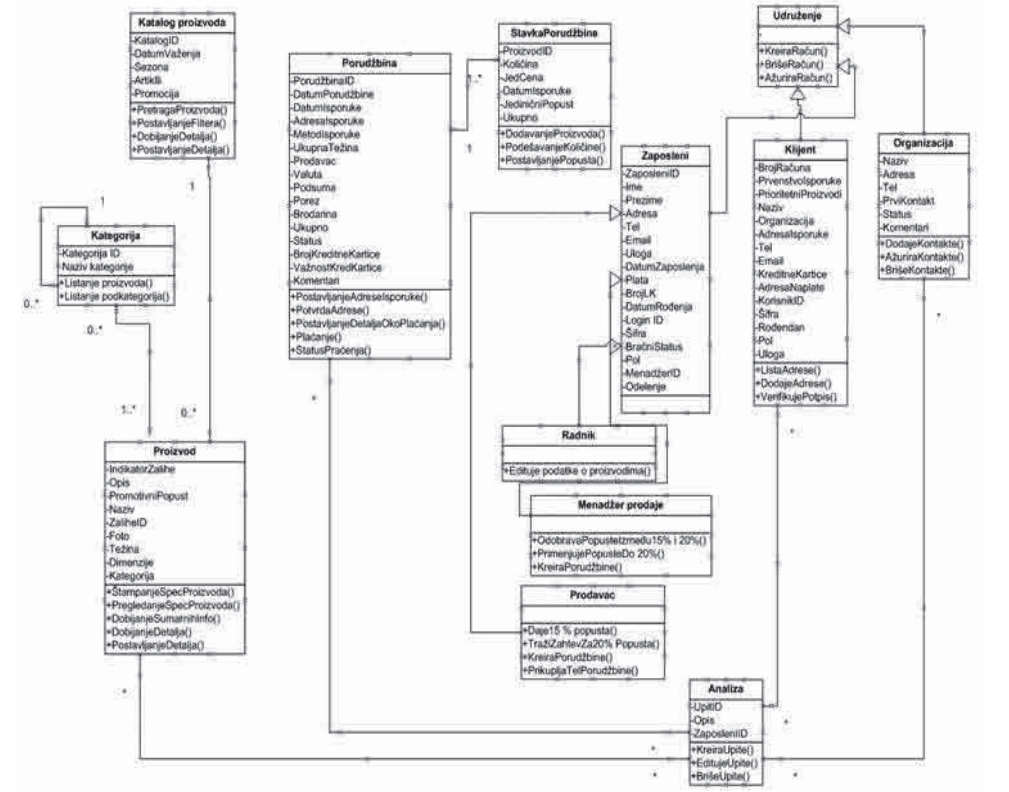
U toku logičkog dizajna, tim kreira izlaznu dokumentaciju:

- logički objektni model;
- preliminarni dizajn korisničkog interfejsa;
- logički model podataka.

Logički objektni model (*Logical Object Model*) se kreira na osnovu prethodno definisanih objekata, servisa, atributa i relacija tokom procesa logičkog dizajna. U toku kreiranja modela, veoma je važno da se razmotre svi poslovni i korisnički zahtevi za bezbednošću, globalizacijom, lokalizacijom, praćenjem i logovanjem, rukovanjem greškama, integracijom sa postojećim sistemima i menadžmentom stanja[17] (*state management*).

1 Menadžment stanja ukazuje na stanje kontrola korisničkog interfejsa (User Interface

Primer logičkog objektnog modela je dat na slici 5.24.

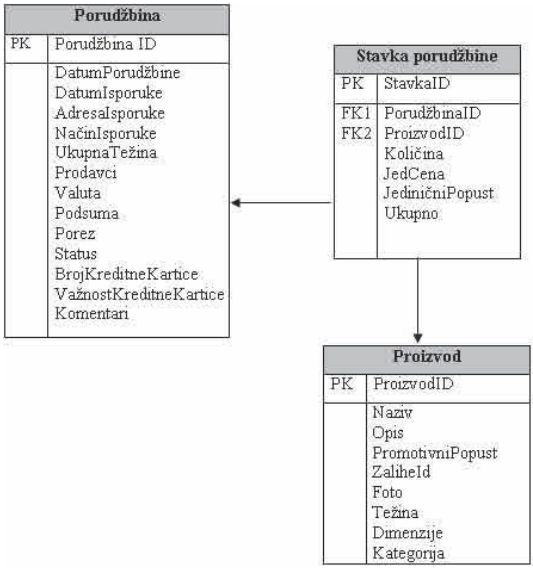



Slika 5.24. Primer logičkog objektnog modela

Logički objektni model i logički model podataka dokumentuju slične informacije za različite načine. Za većinu projekata, sasvim je dovoljno dokumentovati jedan model.

Potrebe za podacima identifikovane tokom konceptnog dizajna se u okviru logičkog dizajna pretvaraju u objekte i relacije. Prvi zadatak kod kreiranja logičkog modela podataka je da se identifikuju entiteti (objekti), atributi, relacije između objekata i kardinalnosti. Logički model podataka na primeru porudžbine je prikazan na slici 5.25.

– UI) kao što su tekstualna polja, radio dugmići i dr. Stanje UI kontrola će zavisiti od stanja drugih UI kontrola. Na primer, stanje UI kontrole dugme (button) će biti aktivno ukoliko su validno unešene vrednosti polja ili će biti neaktivno ukoliko su unešene pogrešne vrednosti ili su polja prazna.




Slika 5.25. Primer logičkog modela podataka

Lista objekata i servisa pruža timu početnu zamisao o funkcionalnosti koju korisnici očekuju. Na osnovu scenarija upotrebe tim projektuje elemente korisničkog interfejsa kao što su dugmići (*buttons*), tekstualna polja (*text fields*), meni (*menu items*) i sl.

Razmotrimo sledeći primer scenarija upotrebe: „Klijent selektuje katalog proizvoda. Na osnovu izabranog kataloga prikazuju se informacije o kategoriji proizvoda i sami proizvodi. Klijent može da bira proizvod kako bi video detalje ili da bira kategoriju kako bi pregledao podkategorije i njihove pridružene proizvode."

Na osnovu ovog scenarija, tim kreira sledeći preliminarni UI dizajn:

- Stranica „Info o proizvodima" prikazuje detaljan opis proizvoda, cene, raspoloživost i dr.
- Stranica „Info o klijentima" prikazuje informacije o registraciji klijenta i neke lične podatke.

Takođe, mogu biti uključene i stranice koje prikazuju informacije o porudžbini, o statusu porudžbine, istoriju prethodnih transakcija itd.

### Optimizacija logičkog dizajna

Pravi izazov predstavlja optimizacija logičkog dizajna prema postavljenim zahtevima i scenarijima upotrebe. Koraci optimizacije logičkog objektnog modela su:

- Validacija modela prema zahtevima
  - Logički model podataka je nekompletan ukoliko i jedan jedini zahtev nije pokriven.
- Verifikacija objekata
  - Identifikovati ulaze i izlaze jednog objekta i funkcionalnost koju objekat mora da obezbedi.
  - Pravovremeno predvideti izlaze i ponašanja svakog ulaza.
- Prolaženjem kroz scenarijo:
  - verifikovati nezavisnost i sekvencu objekata i servisa vođenjem komplentnog scenarija;
  - oceniti frekventnost jedne aktivnosti. Koliko često se ona događa i da li se odvija periodično ili uniformno?
  - odrediti da li servisi koji kontrolišu transakciju zavise i od servisa koji su deo drugih poslovnih objekata;
  - sagledati ona poslovna pravila koja podržavaju više poslovnih objekata;
  - odrediti da li se zahteva sekvencionalni redosled operacija u okviru cele ili dela transakcije;
  - identifikovati kritična vremena. Da li se aktivnost može obustaviti ili se mora odmah odgovoriti?
  - odrediti nezavisnost operacija kod poslovnih objekata.

## 5.5.4. Fizički dizajn

Fizički dizajn je poslednji korak kod faze planiranja prema MSF procesnom modelu. Nakon što se usaglase stavovi oko logičkog dizajna tim prelazi na fazu fizičkog dizajna. Tokom fizičkog dizajna se:

- prikazuje kako će rešenje biti implementirano;
- identifikuju odgovarajuće tehnologije za razvoj rešenja – tim procenjuje tehnologije i određuje koje tehnologije će se koristiti za razvoj rešenja;

- transformiše logički u fizički dizajn;
- obezbeđuje osnova za fazu razvoja (*developing phase*) – tim definiše uloge, odgovornosti i procese razvoja;
- razvija specifikacija komponenti i topologija uvođenja rešenja;
- definiše da li su ispunjene i odobrene kritične tačke – tim procenjuje rizike, ažurira prioritete i finalizira procenu resursa i rasporeda projekta.

Da bi uočili razliku između logičkog i fizičkog dizajna, razmotrimo analogiju za projektovanjem i izgradnjom kuće. Kod logičkog dizajna, određuju se zahtevi na primer za električnim kapacitetima, nivoima svetlosti, uređajima za vodovod i sl. Kod fizičkog dizajna biraju se uređaji koji će se koristiti i podržati prethodno definisane zahteve za strujom, odgovarajućim kablovima i sl.

Fizički dizajn opisuje komponente, servise i tehnologije rešenja sa aspekta developera (Slika 5.26). Fizički dizajn ne podrazumeva kodiranje rešenja, već omogućava da se kreiraju detaljne specifikacije komponenti za razvoj rešenja kao i da se identifikuju tehnologije koje se mogu koristiti za razvoj rešenja.

Slika 5.26. Od konceptnog do fizičkog dizajna

Tokom fizičkog dizajna, svaka rola u timu ima različite odgovornosti:

- **Proizvodni menadžer** – upravlja očekivanjima klijenta, kreira komunikacioni plan i priprema rešenje za uvođenje u okruženje;
- **Program menadžer** – upravlja procesom fizičkog dizajna, kreira funkcionalnu specifikaciju, definiše projektni plan uključujući resurse, rasporede i procenu rizika;
- **Razvoj** – pomaže pri formiranju izlazne dokumentacije fizičkog dizajna i to fizičkih modela, planova razvoja i rasporeda, procenjuje razvoj rešenja, tehnologije, izgrađuje prototipove i priprema okruženje za razvoj;

- **Testiranje –** procenjuju i vrednuju funkcionalnosti i konzistentnost fizičkog dizajna u odnosu na scenarije upotrebe, definišu detaljne planove tesitranja i priprema okruženje za testiranje i kontrolu kvaliteta;
- **Menadžment puštanja rešenja –** procenjuje uticaj fizičkog dizajna na infrastrukturu, definiše rešenja za uvođenje, dodatnu infrastrukturu i zahteve za operacijama;
- **Iskustveni korisnik –** ocenjuje fizički dizajn prema zahtevima korisnika, dizajnira *Help*, definiše plan obučavanja korisnika.

Na kraju fizičkog dizajna, projektni tim dostavlja timu za razvoj neophodnu dokumentaciju koja uključuje:

- preliminarne korisničke interfejse sa pseudokodom;
- dijagrame klasa;
- modele komponenti, dijagrame sekvenci ili dijagrame aktivnosti rešenja;
- šemu baze podataka;
- model uvođenja (*deployment)* koji prikazuje:
  - topologiju mreže - ukazuje na lokacije hardvera i njihove međusobne konekcije;
  - topologiju uvođenja - označava lokacije komponenata rešenja, servisa i skladišta podataka u relaciji sa topologijom mreže;
- specifikacije komponenti - uključuju internu strukturu komponenti i interfejse komponenti;
- strategije pakovanja i distribucije - identifikuju servise koji se pakuju u komponente i određuje kako će komponente biti distribuirane preko topologije mreže. Mogu da uključe i preliminarni plan uvođenja;
- model programiranja - identifikuje izbor implementacije, objekte stanja, načine povezivanja, smernice za višenitno programiranje, rukovanje greškama, izbor bezbednosti i dokumentovanje koda.

Fizički dizajn se završava kada se odobri projektni plan. Fizički dizajn prolazi kroz četiri koraka:

1. **Istraživanje** – na osnovu logičkog dizajna projektni tim kreira tehnička rešenja tako što razmatra ograničenja kod poslovnih

procesa, infrastrukture i arhitekture preduzeća, kao i zahteve za bezbednošću, skalabilnošću, upravljivosti i raspoloživosti rešenja. Na primer, rešenje treba da podrži određeni broj transakcija po sekundi. U toku istraživanja identifikuju se sva fizička ograničenja i zahtevi koji utiču na razvoj rešenja. Izlazna dokumentacija faze istraživanja su:

- trenutna topologija mreže;
- trenutna topologija podataka;
- trenutna topologija komponenti;
- fizički zahtevi aplikacije;
- ažurirani planovi procene i umanjenja rizika.

Fizički zahtevi se prikupljaju iz poslovnog okruženja i arhitekture preduzeća. Neki tipični fizički zahtevi su: performanse, troškovi i koristi, lakoća korišćenja, podrživost, pouzdanost, ponovna upotrebljivost i dr. Neka tipična fizička ograničenja su budžet, raspored, topologija mreže, podataka, komonenti, bezbednost i sl.

U toku faze istraživanja razrešavaju se i mogući konflikti između zahteva i ograničenja. Na primer, aplikacija može da zahteva 100 megabita po sekundi (Mbps) prohodnosti mreže, dok postojeća infrastruktura mreže može da podrži samo 10 Mbps.

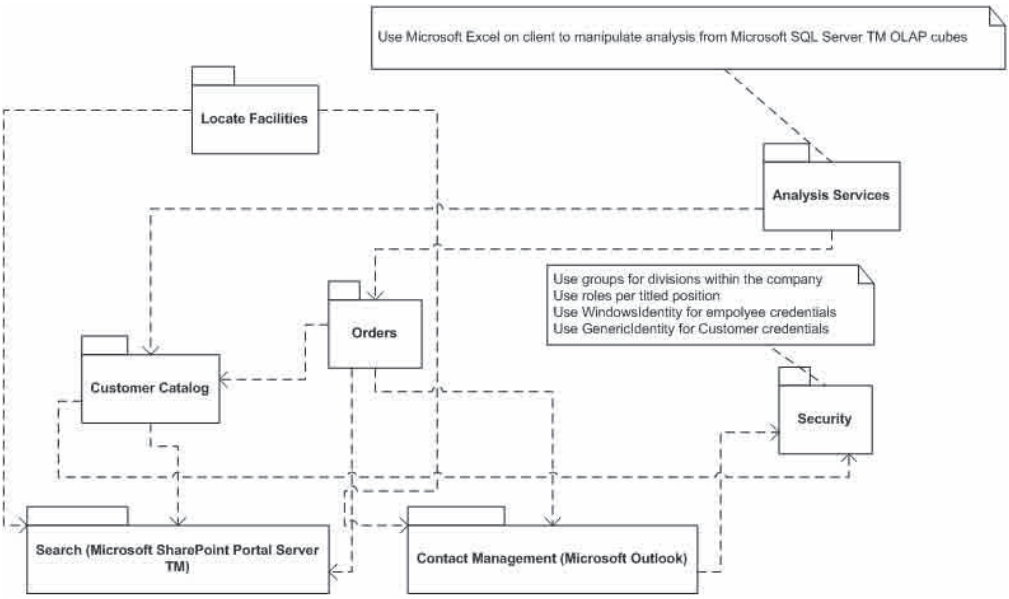
Razrešavanje konflikta se odvija na sledeći način:

- identifikovati zahteve koji su apsolutno neophodni za projekat. Analizirati uticaj na ograničenja i moguće kompromise.
- identifikovati oblasti infrastrutkure gde može da dođe do konflikta;
- analizirati razlike između zahteva i ograničenja i odrediti da li je potrebno doneti odluku.
- sa svim učesnicima na projektu uraditi *brainstorming* rešenja.

2. **Analiza** – tim kreira i poboljšava sledeće modele fizičkog dizajna:
   - **Dijagram klasa (*Class diagrams*)** – predstavljaju statičku strukturu aplikacije.
   - **Dijagram sekvenci (*Sequence diagrams*)** – predstavljaju interakcije između objekata i dinamički aspekt modela objekata.

Obično se koristi kako bi se razjasnile složene relacije između klasa. Tim ažurira klase, poboljšava dijagram sekvenci tako što uključuje relacije između klasa ili servisa zasnovanih na fizičkim ograničenjima ili tehnološkim zahtevima.

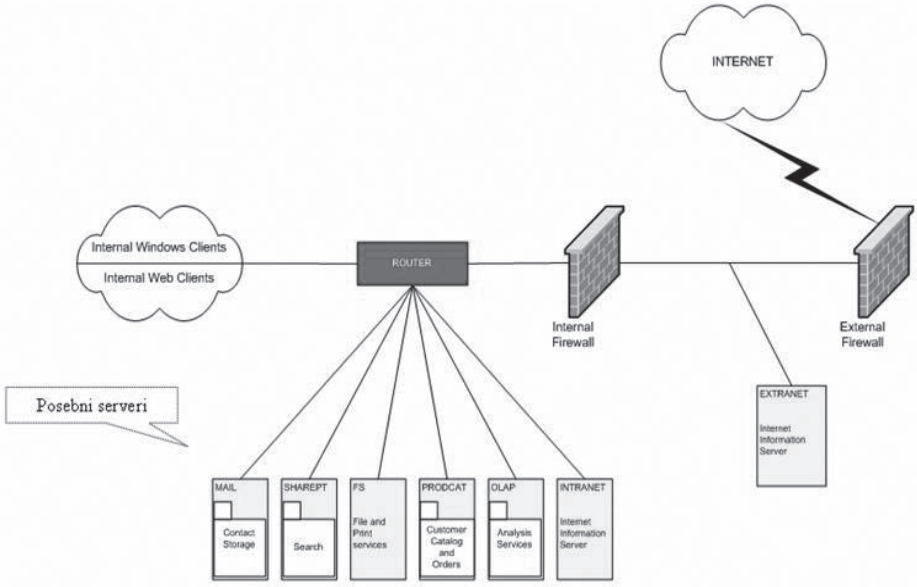
- **Dijagram aktivnosti (*Activity diagrams*)** – uključuje zahteve za platformom i tehnologijom i identifikuje potencijalne *workflow* procese. Može se koristiti dijagram aktivnosti umesto dijagrama sekvenci i obrnuto.
- **Dijagram komponenti (*Component diagrams*)** – prikazuje zavisnosti između komponenti ili paketa komponenti. Projektni tim kreira dijagram komponenti da bi jasno sagledao zavisnosti između komponenata i definisao dalje odluke vezane za pakovanje komponenti. Primer dijagrama komponenti za proces poručivanja je prikazan na slici 5.27.




Slika 5.27. Primer dijagrama komponenti [9]

- **Preliminarni model uvođenja (*Deployment model*)** – Na osnovu arhitekture aplikacije, strukture tima, rasporeda projekta, zahteva, dokumenta ocene rizika i kandidatnih tehnologija, tim kreira preliminarni model uvođenja. Preliminarni model uključuje topologije mreže, podataka i komponenti koje predlaže projektni tim. **Topologija mreže** je mapa infrastrukture koja

prikazuje radne stanice i servere, opisuje njihove funkcije i prikazuje infrastrukturu mreže koja povezuje računare. **Topologija komponenti i podataka** je mapa koja ukazuje na lokacije paketa, komponenti i njihovih servisa u relaciji sa topologijom mreže i lokacijama baze podataka. **Topologija uvođenja** prikazuje fizičku distribuciju komponenti i njihovih lokacija preko različitih slojeva servisa. Primer topologije uvođenja je prikazan na slici 5.28.




Slika 5.28. Primer topologije uvođenja [9]

3. **Racionalizacija** – prikazuju se tačno definisana rešenja, odnosno opisuju se tehnologije, strategije i topologije dizajnirane za konkretno rešenje. Racionalizacija je jedan iterativni proces tokom koga projektni tim pokušava da dizajnira jedno optimalno rešenje. Jedan od ciljeva racionalizacije je distribucija servisa i pakovanje ovih servisa u komponente. Da bi se odredila odgovarajuća strategija distribucije i pakovanja, neophodno je prethodno razmotriti menadžment stanja i performanse rešenja.

   **Menadžment stanja (*state management*)** je proces u okviru koga rešenje održava informacije o stanju i stranicama tokom višestrukih zahteva za istom ili različitim stranicama. Microsoft ASP.NET podržava različite opcije za upravljanje stanjem na klijent i server

strani. Poznato je da su Web *Forms* stranice *stateless*, odnosno ne mogu automatski da ukazuju na to da li su svi sekvencni zahtevi od istog klijenta ili da li pretraživač (*browser*) klijenta još uvek aktivno prikazuje stranicu ili sajt. Opcije menadžmenta stanja na klijent strani u jednom Web zasnovanom rešenju su:

- Svojstva stanja (*ViewState property*) – Web *Forms* (obrasci) stranice obezbeđuju osobine stanja koje automatski pamte vrednosti između višestrukih zahteva za istom stranicom. Stanje je sadržano u strukturi unutar koda stranice. Međutim, što se više vrednosti čuva, povećava se korišćenje resursa servera, bezbednost i implementacija, što dovodi do usporavanja odziva i prikazivanja stranice.
- Skrivena polja (*Hidden fields*) – ukoliko se odlučite da pamtite određene informacije o stranici u skrivenim poljima, tada morate da šaljete stranice serveru putem HTTP Post metoda. Najbolje je da se skladište manje količine frekventnih i promenljivih podataka, na klijent strani.
- Kolačići (*Cookies)* – predstavljaju manje količine često menjanih informacija koji su uskladišteni na klijent fajl sistemu. Oni umanjuju potražnju za resursima servera. Kolačići se koriste za autentifikaciju, praćenje i održavanje određenih informacija o korisnicima kao što su omiljeni sajtovi, sadržaji njihovih elektronskih korpi za kupovinu i sl.
- Stringovi upita (*Query strings*) – predstavlja manju količinu informacija priključenih na kraj URL stranice. Obezbeđuju jednostavan metod za održavanje stanja. Mogu se koristiti samo onda kada se zahteva stranica, nisu bezbedni i imaju limitirane kapacitete.

Opcije za upravljanje stanjem na serverskoj strani u jednom Web zasnovanom rešenju su:

- Stanje aplikacije (*Application state*) – globalni mehanizam za skladištenje kome mogu pristupati sve stranice jedne Web aplikacije. ASP.NET obezbeđuje klasu *HttpApplicationState* za skladištenje vrednosti aplikacije koje će deliti sesije i koje se neće često menjati. Omogućava laku implementaciju i utiče na performanse.

- Stanje sesije (*Session state*) – slično je stanju aplikacija samo što je limitirano trenutnom sesijom pretraživača (*browser*). ASP.NET klasa *HttpSessionState* čuva vrednosti i objekte koji će biti vidljivi samo toj sesiji. Ukoliko različiti korisnici koriste rešenje, svaki korisnik će imati različito stanje sesije. Ukoliko korisnik napusti rešenje i vrati se kasnije, imaće različito stanje sesije.
- Baza podataka (*Database*) – čuvanje stanja u bazama podataka. Ova tehnika se koristi kod većine Web sajtova za elektronsku trgovinu kojima su bitni bezbednost, personalizacija, konzistentnost i *data mining*.

Kod projektovanja, projektni tim razmatra:

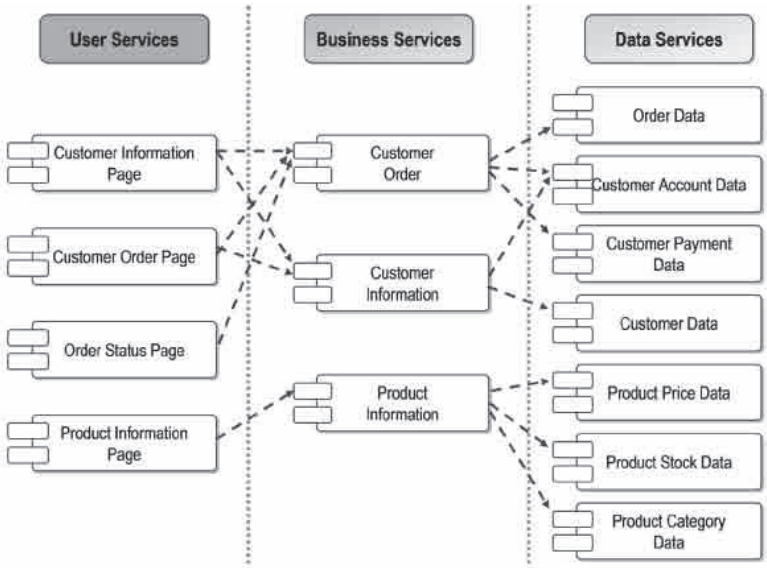
- **Skalabilnost** – sposobnost brzog i lakog proširivanja rešenja kako bi podržao veliki broj korisnika i transakcija;
- **Performanse** – vreme odziva sistema i brzina izvršavanja zadataka aplikacije;
- **Upravljivost** – lakoća upravljanja sistemom na svim nivoima;
- **Upotrebljivost** – lakoća ponovne upotrebljivosti komponenti od strane drugih aplikacija;
- **Granularnost** – odnosi se na broj servisa grupisanih u jedan objekat i na veličinu i broj objekata grupisanih u jednu komponentu.

Jedna od karakteristika dobrog plana komponenti je visoka kohezija i slabo sprezanje. Kohezija (*Cohesion*) je veza između različitih internih elemenata komponente. Komponenta ima visoku koheziju kada su joj servisi usko povezani. Sprezanje (*Coupling*) je odnos komponente prema drugim komponentama. Kada komponenta ima slabu spregu sa ostalim komponentama, to znači da može da obavlja svoje funkcije nezavisno od drugih komponenti.

Prvi korak kod racionalizacije fizičkog dizajna je pakovanje servisa u slojeve: korisničke, poslovne i slojeve podataka. Da bi započeli proces distribucije, treba prvo identifikovati glavne kategorije servisa kod fizičkog modela objekta, a zatim ih razbiti na njihove individualne slojeve servisa. Za svaki poslovni objekat, treba grupi-

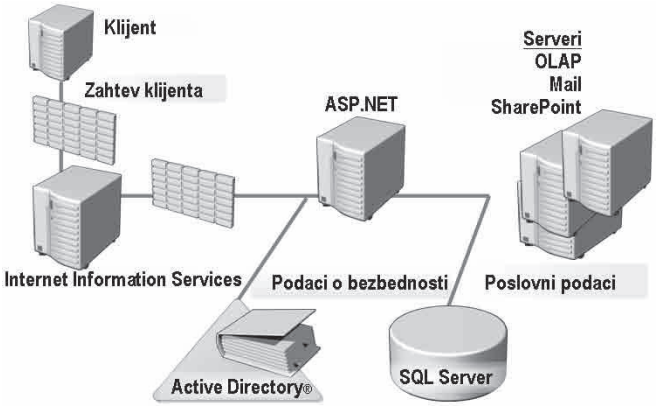
sati servise u tri sloja: sloj servisa korisnika, sloj servisa poslovanja i sloj servisa podataka. Nakon pakovanja servisa u komponente treba distribuirati komponente preko mreže i kreirati topologiju komponenti.

Za svaki čvor u topologiji mreže, tim treba da identifikuje kategorije servisa – korisnik, poslovanje i podaci. Treba odlučiti koje korisničke servise treba distribuirati na Web servere, a koje na klijent računare. Poslovne servise treba distribuirati na servere aplikacije ili Web servere. Servise podataka treba distribuirati na servere baze podataka, odnosno prema lokacijama podataka naznačenih na topologiji podataka.




Slika 5.29. Primer dijagrama komponenti [9]

Model uvođenja (*deployment model*) je dijagram koji pridružuje aplikacije i njihove servise na topologiji servera (Slika 5.30). Svrha modela je da omogući razvojnom timu i timu puštanja rešenja u rad da projektuju i planiraju topologiju servera i njihovu konfiguraciju. Model koristi topologiju mreže kako bi se prikazalo na kom serveru će se nalaziti servisi aplikacija.




Slika 5.30. Model uvođenja [9]

4. **Implementacija** – tim određuje model programiranja koji će razvojni tim koristiti, interfejse za svaku komponentu i internu strukturu svake komponente. Model programiranja opisuje kako će komponente biti struktuirane i kako će razvojni tim koristiti odabrane tehnologije. Model programiranja razmatra tehnologije implementacije, koheziju i sprezanje, konekcije, višenitni model, rukovanje greškama, bezbednost, distribuciju i dr. Nakon opisivanja modela programiranja, projektni tim opisuje kako će komponente biti u vezi. Ova interakcija je dokumentovana interfejsima komponenti, koji opisuju načine pristupa servisima i atributima. Sloj prezentacije obezbeđuje mehanizme komunikacije između korisnika i slojeva servisa. Postoje dva tipa korisnika i to računarski sistemi i korisnici. Računarski sistemi ne zahtevaju korisničke interfejse, već posrednika preko koga će biti u interakciji. Obično su to XML Web servisi koji olakšavaju komunikaciju između dva sloja. Tokom ovog koraka, razvojni tim mora da razmotri i sledeće:
   - fizička ograničenja baze podataka, kao što su memorija ili veličina diska;
   - performanse, detekcija *deadlock*-a, indeksiranje i sl.;
   - primarne i spoljne ključeve;
   - dizajn trigera;
   - migraciju podataka iz prethodnih sistema baza podataka i dr.

Kada se informacioni sistem razvija od samog početka tzv. razvoj u kući (*in-house*) tada su zadaci projektovanja sistema sledeći:

- **Projektovanje arhitekture aplikacija** – definišu se tehnologije koje će informacioni sistem koristiti. Projektovanje arhitekture aplikacija razmatra mrežnu tehnologiju i načine kako će se podaci, procesi i interfejsi distribuirati prema poslovnim lokacijama. Ovom zadatku prethodi analiziranje logičkih modela podataka i procesa koji su inicijalno bili kreirani tokom analize zahteva.
- **Projektovanje baze podataka** – šema fizičke baze podataka.
- **Projektovanje interfejsa sistema** – grafičko korisnički interfejs.
- **Ažuriranje plana projekta** – u odnosu na postavljene zahteve sistema.

Koraci projektovanja sistema kod integrisanja komercijalnog softvera tj. kupljenog rešenja su:

- Identifikovanje i sagledavanje proizvoda koji bi mogli da podrže predloženo rešenje;
- Prikupljanje, procena i rangiranje ponuda vendora;
- Selekcija najboljih predloga;
- Sklapanje ugovora sa izabranim vendorom.

### Projektovanje sloja prezentacije

Sloj prezentacije (*presentation layer*) je deo poslovne aplikacije koji obezbeđuje mehanizme komunikacije između korisnika i slojeva poslovnih servisa sistema. Jednostavan sloj prezentacije sadrži komponente korisničkog interfejsa koji se nalaze na grafičko korisničkim interfejsima kao što su *Microsoft Windows* forme ili *Microsoft* ASP.NET *Web* forme

Informacije koje su analizirane i optimizovane tokom faze definisanja vizije i faze planiranja predstavljaju ulaze u fazu projektovanja sloja prezentacije. Informacije uključuju: zahteve i ograničenja rešenja, scenarija upotrebe, *workflow* modele, korisničke profile, opise zadataka i dr.

Za većinu korisnika najvažniji deo je upravo korisnički interfejs koji za njih predstavlja samu aplikaciju. Dobro dizajniran korisnički interfejs u najvećem broju slučajeva, obezbeđuje uspeh i prihvatljivost poslovne aplikacije. Komponente korisničkog interfejsa upravljaju interakcijom sa korisnicima, prikazuju i prihvataju podatke od korisnika, interpretiraju događaje koji su posledica akcija korisnika, menjaju stanje korisničkog

interfejsa i prate tok zadataka. Interfejs aplikacije se može podeliti na tri dela: model (objekat aplikacije), pogled (prezentacija korisnicima) i kontroler (korisnička kontrola).

Dizajn korisničkog interfejsa bi trebao da implementira zadatke korisnika na način koji je intuitivan korisnicima. Ovaj cilj se može postići jedino uključivanjem korisnika u svim fazama projektovanja korisničkog interfejsa i to za vreme prototipovanja, beta testiranja i ranog usvajanja programa. Neka od pitanja koja bi trebalo razmotriti u toku projektovanja korisničkog interfejsa su:

- Kako će korisnici biti interakciji sa sistemom?
- Da li korisnički interfejs koristi odgovarajuću terminologiju, koncepte i metafore?
- Da li korisnici mogu lako da pronađu neophodne naredbe?
- Da li je *workflow* korektan i kompletan?
- Da li interfejs optimizuje *workflow*?
- Da li korisnici mogu brzo i lako da pristupe *Help*-u?
- Da li korisnici mogu da prilagode korisnički interfejs svojim potrebama?
- Da li postoje alternativni načini za izvršavanje određenih zadataka u slučaju da se pojavi problem?

Da bi interfejs bio efektivan treba razmotriti sledeće karakteristike:

- Intuitivan dizajn – dizajnirati interfejs tako da korisnici intuitivno mogu da razumeju kako da ga koriste.
- Optimalno korišćenje prostora ekrana – planirati količinu informacija koja će biti prikazana na jednom ekranu.
- Odgovarajući izgled - obratiti pažnju na boje, na primer za poruke greške koristiti crvenu i sl.
- Lakoća navigacije – na primer, za prelazak na polja koristiti Tab, strelice i druge prečice sa tastature.
- Kontrolisana navigacija – održati redosled pristupanja komponentama.
- Difoltne vrednosti – neki put je bolje obezbediti difoltne vrednosti.

- Provera ulaznih podataka – dobro je proveriti unete podatke pre nego što ih aplikacija obradi.
- Meni, *toolbars* i *Help* – dizajnirati interfejs tako da obezbedi pristup svim funkcionalnostima aplikacije.
- Efikasno rukovanje događajima – bitno je da korisnik ne mora da čeka dugo na odziv aplikacije.

Kod projektovanja korisničkog interfejsa veoma je bitno odabrati najbolji interfejs model. Neki opšti modeli i tehnologije implementacije korisničkog interfejsa su:

- **Windows zasnovan korisnički interfejs** – obezbeđuje bogatu korisničku interakciju, interakciju sa drugim aplikacijama, omogućava diskonektovan rad.
- **Web zasnovani korisnički interfejs** – kada je potrebno da rešenje radi u Web okruženju. Omogućava prenosivost na mnoge druge uređaje i platforme.
- **Korisnički interfejs mobilnih uređaja** – u današnje vreme sve je popularnije koristiti WAP telefone ili *pocket* PC-jeve. Korisnički interfejs za mobilne uređaje treba da prikaže informacije na manjim ekranima i da ponudi prihvatljivu funkcionalnost.
- **Dokument zasnovani korisnički interfejs** – neke aplikacije omogućavaju korisnicima unošenje i pregledanje podataka u dokument formatu. Na primer, korisnik može da pregleda i unosi podatke u Word dokumente ili Excel radne listove.

Izbor okruženja klijenta (*client environment*) za korisnički interfejs poslovnog rešenja je određena načinom korišćenja aplikacije i povezanosti sa drugim sistemima. Kada su korisnici na lokalnoj mreži ili veoma brzoj WAN mreži, najverovatnije će izbor klijenta biti okruženje bogato funkcionalnostima, poznatije kao bogati klijent (*rich client*). Brzina komunikacije između računara i sistema koji im pristupaju, ovde nije primarna tema. Ovakav tip klijenta ima veliki broj opcija za razvoj i implementaciju.

Drugi izbor klijenta je tanak klijent (*thin client*). Tanki klijenti, kao što su Web pretraživači (*browsers*) i udaljene desktop komunikacije se najčešće koriste za udaljene ili distribuirane korisnike ili za korisnike sa slabim konekcijama kao što su *dial-up* modem. Ovakvi korisnici su zainteresovani za brzinu komunikacije i zahtevaju korisnički interfejs koji šalje minimum informacija putem mreže.

Pre nego što se odlučite za klijent okruženje, prethodno je potrebno razmotriti nekoliko faktora:

- Klijent uređaji – aplikacija koja mora da se koristi na različitim klijent uređajima, PDA (*Personal Digital Assistants*), telefonima, interaktivnim televizijama i sl., bi trebala da bude aplikacija tankog klijenta.
- Grafički crteži – ukoliko aplikacija treba da podrži visoko dinamičke grafičke crteže, kao što su CAD ili video animacije, onda bi se trebalo odlučiti za bogatog klijenta.
- Interakcija – ukoliko aplikacija treba da podrži visok nivo interakcije, odnosno ukoliko sadrži dinamičke korisničke interfejse koji se često menjaju, obezbeđuju brze odzive, provere i druge dinamičke operacije, onda bi takva aplikacija trebala da bude bogat klijent.
- Širina mreže – aplikacija tankog klijenta zahteva veću prohodnost mreže za razliku od bogatog klijenta.
- Diskonektovani klijent – ukoliko klijent mora da radi u diskonektnovanoj korporativnoj mreži, onda treba koristiti bogatog klijenta.
- CPU operacije – ukoliko korisnik izvršava operacije koje zahtevaju intenzivan rad CPU-a, onda bi takve operacije trebalo da se odvijaju na računarima krajnjih korisnika. Aplikacije koje zahtevaju intenzivno korišćenje CPU-a je najbolje da rade u okruženju bogatog klijenta.
- Zaključavanje baze podataka – ukoliko aplikacija zahteva visok stepen konkurentnosti kontrola onda je bogat klijent neminovan.
- Lokalni resursi – ukoliko aplikacija treba da pristupi lokalnim resursima, kao što su baze podatka, štampači, skladišta, registri ili bilo koji drugi lokalni uređaji, onda je pogodan bogat klijent.
- Bezbednost – u okruženju visoke bezbednosti, prednosti aplikacije bogatog klijenta su: autorizacija, autentifikacija, bezbedna komunikacija, menadžment stanja i praćenje.

Nakon što se odlučite za dizajn korisničkog interfejsa, počinjete da kreirate prototipe korisničkog interfejsa. Poslednji korak je njihova validacija u odnosu na zahteve, slučajeve korišćenja, scenarija upotrebe i logički dizajn.

### Projektovanje sloja aplikacije

Arhitektura aplikacije ukazuje na tehnologije koje će se koristiti kod implementacije informacionog sistema. Teme koje treba razmotriti su:

- stepen distribuiranosti informacionog sistema;
- distribuiranje baze podataka;
- informacione tehnologije za sve softvere koje se razvijaju u kući;
- integracija i kastomizacija kupljenog softvera;
- tehnologija koja će se koristiti za interfejs sa drugim sistemima.

Današnji informacioni sistemi su uglavnom distribuirani. Distribuirani sistemi su sistemi gde se komponente informacionog sistema distribuiraju na više lokacija preko računarske mreže. Suprotno distribuiranim sistemima su centralizovani sistemi. Distribuirane sisteme je daleko komplikovanije implementirati nego centralizovana rešenja.

Klijent/server sistem je distribuirano rešenje gde su prezentacija, logika prezentacije, aplikaciona logika, manipulacija podacima i slojevi podataka, distribuirani između klijent PC-ja i jednog ili više servera. Logika aplikacije je podeljena između klijenta i servera tako da obezbedi optimalno korišćenje resursa. Na primer, prezentacija podataka i provera ulaznih podataka su sastavni deo klijent-aplikacije, dok se rukovanje podacima, u smislu njihovog fizičkog smeštaja i kontrole pristupa, vrši na serveru.

Tanak klijent (*thin client*) je PC računar koji predstavlja samo interfejse korisniku, dok se aplikaciona logika izvršava na udaljenom aplikacionom serveru. Serveri kod klijent/server modela moraju da budu veoma jake mašine, koji uglavnom imaju instalirane klijent/server operativne sisteme, kao što su *Unix*, *Windows* 2000 *Server* ili *Enterprise Edition* ili *Linux*. Mogu biti:

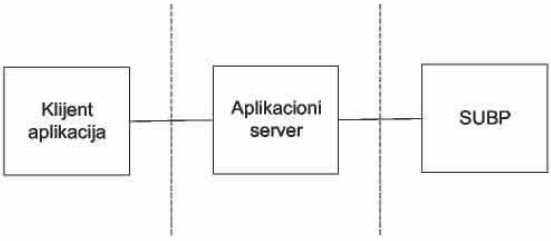
- **sever baze podataka** – hostuje jednu ili više deljenih baza podataka, izvršava sve naredbe i servise baze podataka. Na njima mogu biti instalirani Oracle, *Microsoft* SQL *Server*, IBM DB2 ili drugi.
- **transakcioni serveri** – osiguravaju da su sve promene na bazi podataka za jednu poslovnu transakciju uspešno obavljene ili su otkazane. Primeri su *Microsoft Transaction Server*, BEA *Tuxedo* ili IBM CICS i dr.

- **aplikacioni serveri** – izvršavaju aplikacionu logiku i servise. Komunicira kao *front end* sa klijentima zbog prezentacije i *back end* sa serverima baze podataka radi pristupa i ažuranja podacima. Aplikacioni server je uglavnom integrisan sa transakcionim serverom. Većina aplikacionih servera su bazirani na CORBA standardu za deljenje objekata ili Microsoft COM+ ili DNA standardu.
- **Groupware server** – hostuje servise za email i druge funkcionalnosti radnih grupa. Na primer, *Microsoft Exchange Server.*
- **Web server** – hostuje Internet ili intranet Web sajtove. Komunicira i sa debelim i tankim klijentima kako bi im prosledio dokumenta (HTML formata) ili podatke (XML formata). Pojedini Web serveri su posebno projektovani da hostuju e-*commerce* aplikacije koje podržavaju elektronsku trgovinu (npr., *Netscape Commerce Server*).

Klijent/server sistemi sa troslojnom arhitekturom (*three-tier architecture*) predstavljaju sisteme sa tri, u velikoj meri, nezavisna podsistema:

- podsistem za interakciju sa korisnikom (implementira funkcije korisničkog interfejsa);
- podsistem za implementaciju osnovnih funkcija sistema (implementira tzv. "poslovnu logiku");
- podsistem za rukovanje podacima, gde se pre svega misli na sistem za upravljanje bazama podataka.

Na slici 5.31 je prikazan odnos ova tri podsistema gde se vidi da ne postoji direktna veza između podsistema za interakciju sa korisnikom i podsistema za rukovanje podacima. Zbog ovakvog međusobnog odnosa, ovi podsistemi se nazivaju i slojevi.




Slika 5.31. Elementi troslojne arhitekture sistema

Za razliku od dvoslojnog modela obrade podataka, u okviru koga je logika aplikacije bila podeljena između klijenta i servera, u modelu troslojne arhitekture sistema ona se nalazi koncentrisana u tzv. aplikacionom serveru čija je namena da izvršava programski kôd koji implementira logiku aplikacije. Klijent aplikacija je namenjena samo za implementaciju korisničkog interfejsa, a funkcija sistema za upravljanje bazom podataka je isključivo fizičko rukovanje podacima. Tako organizovan sistem je jednostavniji za održavanje, jer je moguće nezavisno razvijati korisnički interfejs i logiku aplikacije.

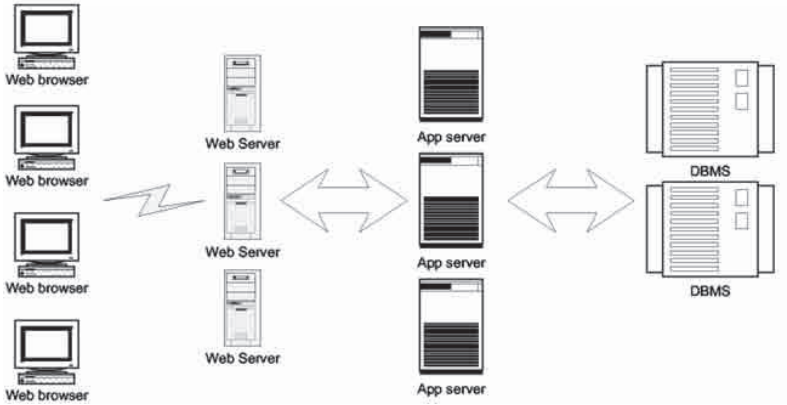
Jedna od važnijih karakteristika troslojnih sistema je skalabilnost. Naime, povećavanje broja klijenata u tom slučaju je veoma jednostavno. Povećavanje propusne moći servera srednjeg sloja postiže se dodavanjem novih serverskih mašina. Analogno tome moguće je povećati i propusnu moć zadnjeg sloja. Ovde se može uočiti da se povećanje brzine odziva serverskog sloja može postići dodavanjem novih serverskih mašina uz korišćenje postojećih, te se na taj način može iskoristiti i oprema koja ne mora imati vrhunske performanse. Sistem sa više servera karakteriše i povećana pouzdanost i fleksibilnost. Pored toga, moguće je efikasno vršiti balansiranje opterećenja serverskog podsistema.

Široka prihvaćenost *World Wide Web*-a uticala je na pojavu Web čitača, čiji je izgled i način korišćenja postao poznat većini korisnika. Informacioni sistemi kod kojih se komunikacija sa korisnikom odvija kroz Web čitač u velikoj meri eliminiše potrebu za dugotrajnom i skupom obukom korisnika. Sa druge strane, npr., Java programi se, u formi apleta, mogu ugrađivati u Web stranice i na taj način distribuirati korisnicima. Nezavisno od konkretne fizičke platforme klijenta (dovoljan je Web čitač sa podrškom za Javu), takve klijent aplikacije se mogu na mreži automatski distribuirati i instalirati.

Troslojne arhitekture informacionih sistema podrazumevaju oslanjanje na standarde gde su najčešće u pitanju sistemi zasnovani na Internet tehnologijama. Oslanjanje na standarde omogućava integraciju informacionih sistema heterogenih u pogledu korišćene hardverske i softverske opreme. Na primer, računarska mreža ovakvog sistema može biti zasnovana na TCP/IP familiji protokola. Serveri u mreži mogu biti od različitih proizvođača, sve dok obezbeđuju standardne servise predviđene protokolom.

Proširivanjem koncepta troslojnih sistema dolazi se do pojma višeslojnih sistema (*multitier architecture*), gde se vrši dalja podela na komponente

u okviru srednjeg sloja sa ciljem još većeg povećanja skalabilnosti, odnosno performansi sistema. Jedna moguća arhitektura višeslojnih sistema je prikazana na slici 5.32. Srednji sloj je podeljen na dva sloja: jedan je namenjen za opsluživanje Web klijenata, a drugi sadrži komponente koje implementiraju poslovnu logiku sistema.

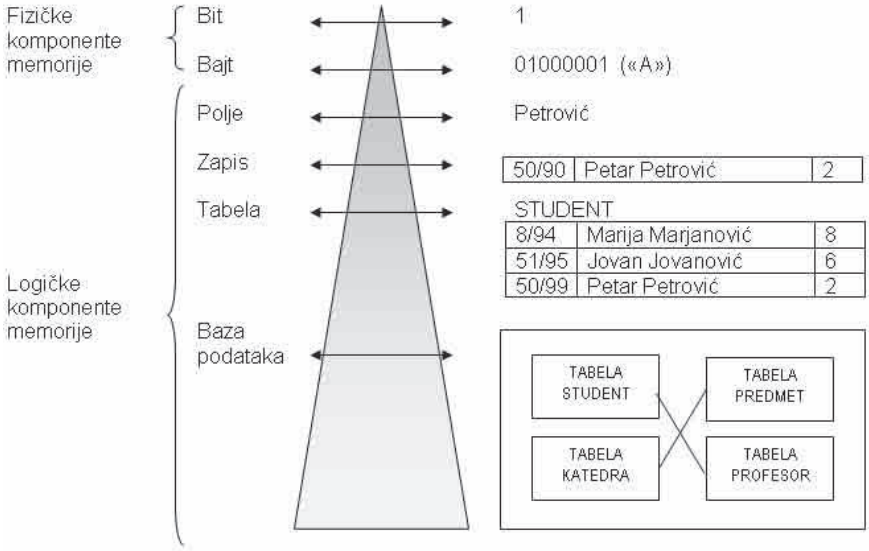



Slika 5.32. Primer arhitekture višeslojnog sistema

### Projektovanje sloja podataka

**Baza podataka** (BP) se najopštije može definisati kao dobro struktuirana kolekcija podataka, uskladištenih sa minimumom redundanse, koju, zajednički, koriste i održava više korisnika i aplikacija. Baza podataka se projektuje tako da što efikasnije zadovolji zahteve za informacijama. Baza podataka mora da omogući pravovremno dobijanje kvalitetnih informacija na svim nivoima odlučivanja kako bi na taj način povećala efikasnost organizacije. Da bi se ovaj zadatak izvršio, potrebno je detaljno poznavati poslovanje cele organizacije, a ne samo njenih izolovanih delova.

Podaci u bazi podataka su organizovani sledećim redom: bit, bajt, polje, zapis, tabela, baza podataka (Slika 5.33).




Slika 5.33. Hijerarhija organizacije podataka

Bit je najmanji element prikaza podataka u računaru i uzima vrednosti 0 ili 1. Bajt je najmanja adresibilna jedinica koja sadrži 8 bita i predstavlja jedan karakter. Polje ili atribut predstavlja se sa jednim ili više bajtova. Zapis sadrži skup atributa o jednom primerku (instanci), na primer atributi jednog studenta su: broj indeksa: 50/90, ime i prezime: Petar Petrović, semestar: 2. Tabela je kolekcija različitih zapisa, a koje pripadaju jednom objektu. Baza podataka je kolekcija povezanih tabela.

Proces projektovanja baza podataka je deo procesa projektovanja informacionih sistema i sastoji se od sledećih faza:

- **Analiza sistema i zahteva korisnika** – cilj je izrada poslovnog modela sistema (*Business Model*) koji predstavlja model procesa. Postoji više različitih metoda za analizu sistema i specifikaciju zahteva korisnika koje se mogu podeliti na konvencionalne i objektne. Kod objektno-orijentisanog pristupa izrađuju se poslovni slučajevi korišćenja (*Business Use Cases*).
- **Specifikacija aplikacija** – u ovoj fazi se izrađuju detaljne specifikacije aplikacija. Aplikacija se obično predstavlja kao skup ulaza i izlaza koji su najčešće opisani preko ekranskih formi. Kod objektno-orijentisanog razvoja koriste se slučajevi korišćenja, sistemski dijagrami sekvenci i dijagrami aktivnosti.

- **Izrada konceptualnog modela** – konceptualni model je model realnog sistema koji predstavlja celokupan informacioni sadržaj baze podataka. Izgradnja konceptualnog modela i specifikacija aplikacija su međusobno uslovljeni. U nekim pristupima se konceptualni model izgrađuje direktno, na bazi poslovnog modela sistema, a objekti konceptualnog modela služe za specifikaciju aplikacija. U drugim pristupima se konceptualni model razvija na osnovu specifikacije aplikacija, odnosno strukture ulaza i izlaza aplikacije. Kao alat za konceptualno modelovanje u konvencionalnim pristupima se najčešće koristi model objekti-veze. Kod objektno-orijentisanog pristupa, za prikaz konceptualnog modela koristi se dijagram klasa.
- **Logičko projektovanje baze podataka** – predstavlja transformaciju konceptualnog modela u opis baze podataka u jeziku za opis (DDL) konkretnog DBMS-a. Pojedini CASE alati sadrže ovu mogućnost transformacije modela objekti veze ili dijagrama klasa u relacioni model.
- **Fizičko projektovanje baze podataka** – podrazumeva projektovanje interne, fizičke strukture baze podataka, na osnovu logičkog modela i specifikacija svih aplikacija. Fizička struktura baze podataka treba da omogući ostvarivanje zadovoljavajućih performansi sistema.
- **Implementacija i podešavanje** – Baza podataka se imlementira u konkretnom DBMS-u, a zatim se prate performanse sistema, izmene u aplikacijama i logičkoj strukturi baze i vrši odgovarajuće podešavanje fizičke strukuture.
- **Projektovanje aplikacija** – na osnovu specifikacije aplikacija i logičkog modela baze podataka u konkretnom DBMS-u obavlja se projektovanje aplikacija. U konvencionalnim pristupima za ovu fazu se koristi metoda Strukturnog projektovanja, dok se za objektne sisteme standardno primenjuje jedinstveni proces razvoja softvera (*The Unified Software Development Process*).
- **Implementacija i održavanje** – podrazumeva se kodiranje, testiranje i uvođenje aplikacija. Održavanje aplikacija podrazumeva promene u aplikacijama, a ponekad i u logičkom modelu baze podataka do kojih se dolazi usled otkrivenih grešaka, promena u implementacionom okruženju ili promena zahteva korisnika.

Izučavanju baze podataka može se pristupiti sa dva aspekta:

- **modeli podataka** – teorije pomoću kojih se specifikuje i projektuje neka konkretna baza podataka ili informacioni sistem. U prethodnom delu knjige smo se upoznali sa nekim modelima podataka.
- **sistemi za upravljanje bazom podataka (*Database Management Systems* – DBMS)** – softverski sistem koji kreira, pristupa, upravlja, kontroliše, čuva i pretražuje podatke. Osnovne komponente DBMS-a su (Slika 5.34):

  1. **Fizička baza podataka** – baza podataka je smeštena na diskovima. Sistem za upravljanje skladištenjem podataka mora da obezbedi jednoobrazan pristup svim formatima podataka na svim vrstama memorije. Pored podataka, baza sadrži i metapodatke, tzv. rečnik podataka (*Data Dictionary*, *Catalog* ili *Data Directory*). Rečnik baze podataka opisuje strukturu posmatrane baze podataka, pravila očuvanja integriteta podataka, prava korišćenja, indekse posmatrane baze i sl. DBMS održava i bazu indeksa koji omogućavaju brz pristup „indeksiranim“ podacima baze. Najčešća struktura indeksa je B-stablo.

2. **Sistem za upravljanje skladištenjem podataka** – sadrži dve osnovne komponente:

   - **Upravljanje datotekama (*File Manager*)** – upravlja lokacijama datoteka preko kojih se realizuje baza podataka i pristupima blokova podataka na zahtev upravljanja baferima. Diskovi su obično podeljeni u blokove koji predstavljaju kontinualne segmente memorije veličine od 4000 do 16000 bajta.
   - **Upravljanje baferima (*Buffer Manager*)** – prihvata blok podataka sa diska, dodeljuje mu izabranu stranicu centralne memorije, zadržava ga u skladu sa ugrađenim algoritmom upravljanja baferima, a zatim vraća na disk oslobađajući dodeljenu mu stranicu.

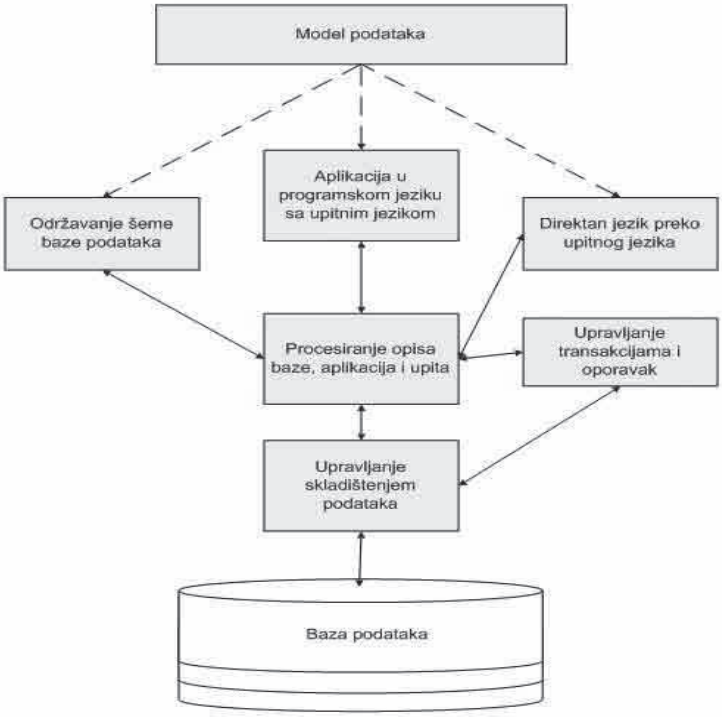
3. **Upravljanje transakcijama i oporavkom** – obezbeđuje konzistentnost baze podataka u konkurentnoj obradi podataka i pri veoma ozbiljnim otkazima sistema.

4. **Održavanje šeme baze podataka** – dizajn baze podataka se opisuje modelom šeme baze podataka. Šema baze podataka je

fizički model baze podataka i predstavlja tehničku implementaciju logičkog modela podataka. Šema baze podataka opisuje strukturu baze, pravila integriteta i prava korišćenja. Održavanje šeme baze podataka podrazumeva kreiranje i modifikovanje ovog opisa koji se čuva u rečniku podataka.

5. **Podšema** – ulazi u DBMS su upiti i aplikacije i šema baze podatka. Sva tri navedena ulaza se realizuju preko jezika baze podatka:
   - **Jezik za opis podataka (*Data Definition Language* - DDL)** se koristi za izradu rečnika podataka, inicijalizaciju ili kreiranje baze podataka, opisivanje logičkog pogleda za svakog individualnog korisnika ili programera i specifiranje bilo kog ograničenja nad poljima ili tabelama baze podataka.
   - **Jezik za manipulaciju podataka (*Data Manipulation Language* - DML)** se koristi za održavanje podataka, koja uključuje takve operacije koje ažuriraju, ubacuju i brišu delove baza podataka i realizuju upite.

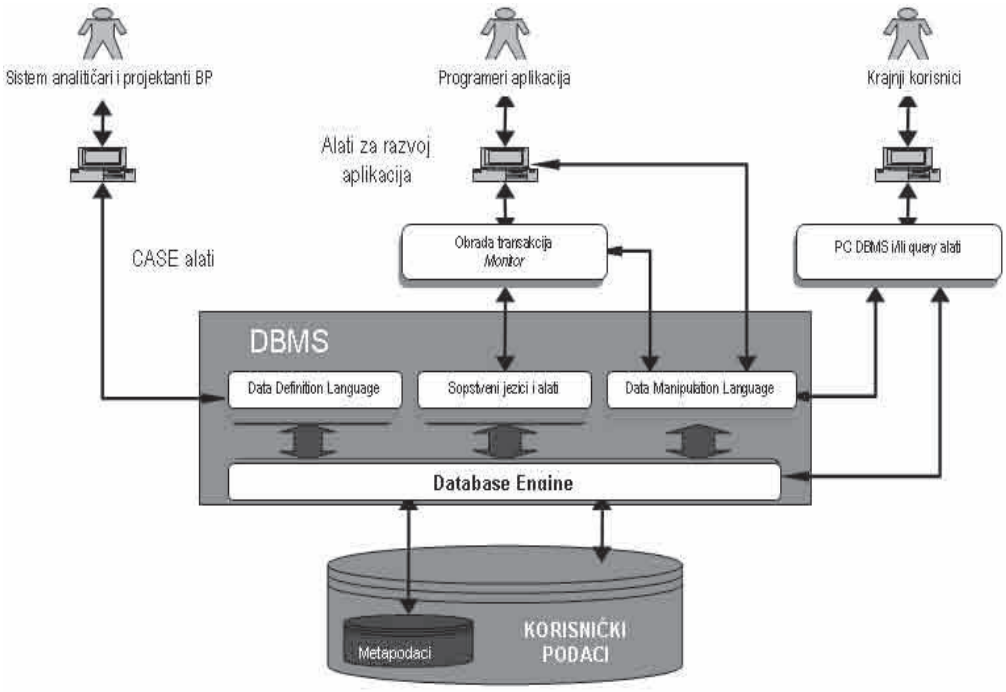
Jezici baze podataka su ili potpuno neproceduralni (relacione baze) ili u sebi sadrže proceduralne delove (objektne baze). Neproceduralni jezici sadrže konstrukcije preko kojih se samo specifikuju uslovi koje treba da zadovolji željeni rezultat, a ne i procedura pomoću koje se dobija taj rezultat. Neproceduralni jezici se često nazivaju i upitni jezici jer im je osnovna namena specifikacija upita. Osnovni zadatak procesora upita (*Query Processor*) je da transformiše neproceduralni iskaz u sekvencu akcija koje treba da obavi sistem za upravljanje skladištenjem podataka. Najsloženiji deo je optimizacija upita, tj. nalaženje najpogodnije procedure (sekvence akcija) za realizaciju neproceduralnog iskaza. Najpoznatiji upitni jezik je SQL (*Structured Query Language*), međutim sve veća primena objektnih baza dovela je do OQL (*Object Query Language*) standardnog upitnog jezika za ovaj tip baze.




Slika 5.34. Osnovne komponente DBMS-a [8]

DBMS je tronivovske arhitekture koja ima sledeće nivoe (Slika 5.35):

- **Eksterni dizajn (podšema)** je takvo predstavljanje podataka, koje odgovara zahtevima nekog korisnika ili programima. To je apstraktan prikaz određenog dela baze podataka, koji je relevantan za određenog korisnika.
- **Konceptualni nivo ili šema baze podataka** je apstraktni globalni logički pogled na bazu podataka. U njemu su obuhvaćene sve vrste objekata, njihovi atributi i međusobne veze. Šema baze podataka je važnija od programskog jezika za korisnika, jer predstavlja temelj celokupne obrade podataka u organizaciji. Taj temelj raste i postaje s vremenom sve opsežniji i kompleksniji.
- **Fizički ili interni pogled** na podatke je opis fizičke organizacije podataka, koja je, naravno, zavisna od medijuma za memorisanje. Ovaj pogled pokazuje u kakvom su obliku podaci stvarno memorisani, kakvi su pojedini zapisi, kako su raspoređeni, kakvi se mehanizmi za adresiranje koriste i dr.

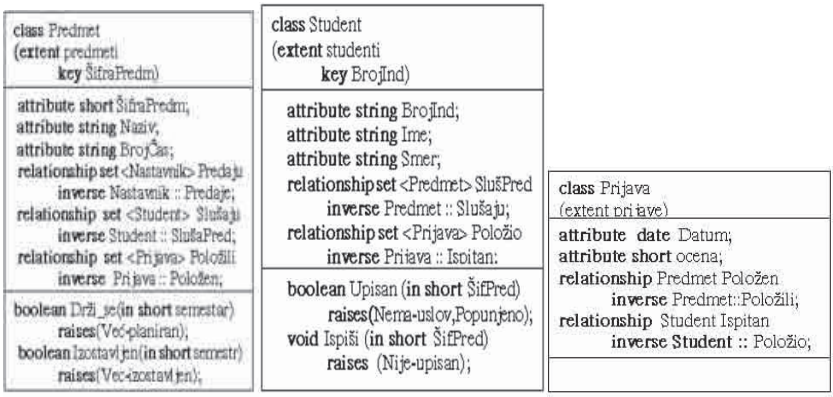



Slika 5.35. Tipična DBMS arhitektura

Objektna orjentacija postepeno preuzima primat i u sistemima za upravljanje bazama podataka (*Database Management Systems* – DBMS). Cilj razvoja objektnih DBMS-a je da se preuzmu i druge bitne karakteristike objektno-orijentisanog razvoja i objektno-orijentisanih programskih jezika, kao što su: učaurenje (enkapsuliranje), odvajanje implementacije softvera od njegove specifikacije tj. načina korišćenja; modularizacija, grupisanje pojedinih delova softvera u koherentne i relativno izolovane celine; ponovno korišćenje ranije razvijenog softvera i sl. Kod konvencionalnih DBMS sistema, tronivovska arhitektura je tretirala bazu podataka kao samostalnu komponentu informacionog sistema koja je u najvećoj meri bila nezavisna od aplikacija koje je koriste. To je razlog i što su jezici DBMS-a (DDL i DML) potpuno nezavisni od programskih jezika u kojima se razvijaju aplikacije. Nasuprot ovome, objektni DBMS teže da integrišu funkcije DBMS-a u programski jezik kako bi se poboljšale performanse sistema. Osnovna karakteristika objektnih DBMS-a je integracija objektno-orijentisanih jezika i funkcija DBMS-a. Oni omogućavaju da se objekti baze podataka pojavljuju kao objekti programa.

Osnovni elementi arhitekture objektnih DBMS-a su:

- **objektni model** – treba da bude definisan kao zajednička osnova za objektne programske jezike, komunikaciju objekata u nekoj klijent/server arhitekturi i objektne baze podataka. U objektnom modelu objekat se definiše kao entitet koji je sposoban da čuva svoja stanja i koji okolini stavlja na raspolaganje skup operacija preko kojih se tim stanjima pristupa. Stanje objekata se predstavlja vrednostima njegovih atributa i veza sa drugim objektima u sistemu. Ponašanje sistema se opisuje preko skupa operacija koje objekat izvršava ili se nad njim izvršavaju. Primer objektnog modela je prikazan na slici 5.36.
- **objektni specifikacioni jezici** – pod specifikacijom se podrazumevaju eksterne karakteristike, vidljive korisniku kao što su: operacije koje mogu biti pozvane, osobine kojima se može pristupiti i izuzeci koji se mogu dogoditi pri izvođenju operacija. Specifikacija tipa se daje preko koncepta interfejsa ili preko koncepta klase. Interfejs se ne može direktno instancirati i služi da se preko njega definišu operacije koje će naslediti neki korisnički definisani objekti.




Slika 5.36. Primer objektnog modela [8]

- **objektni upitni jezik** – OQL (*Object Query Language*) je namenjen objektnom modelu. Upiti napisani u OQL-u mogu da vrate objekte čiji tipovi odgovaraju skupu tipova u objektnim programskim jezicima.
- **povezivanje jezika** (*language binding*) – povezivanje jezika baze podataka i objektnih programskih jezika. Radeći sa objektnim

jezikom nad objektnom bazom podataka, programer treba da stekne utisak da radi sa jednim, a ne sa dva posebna jezika i nekim vidljivim načinom njihovog povezivanja. Da bi se ostvario ovaj cilj definišu se tzv. *language binding*.

**Aktivni sistemi baza podataka** automatski izvršavaju određene operacije kao odgovor na pojavu određenih događaja i/ili za zadovoljenje određenih uslova. Aktivni sistemi imaju mehanizme za prepoznavanje situacija i za automatsko pokretanje odgovarajućih procedura kao reakcija na nastale situacije. Pasivni sistemi, za razliku od aktivnih, imaju manje mogućnosti za predstavljanje dinamike realnog sistema. Oni pamte logičku šemu i podatke, dok složeniji uslovi integriteta i pravila poslovanja se realizuju aplikativnim programima.

Za ilustraciju načina rada pasivnih sistema prikazaće se primer magacinskog poslovanja. Na skladištu jedne parfimerije nalazi se određeni broj proizvoda. Na osnovu informacija o broju prodatih proizvoda smanjuje se broj na zalihama. Kada broj proizvoda na skladištu postane manji od 50, tada se vrši naručivanje novih 150 komada.

Kod pasivnih baza podataka čuvaju se samo podaci o trenutnom broju proizvoda kojima se raspolaže. Sva ostala semantika se nalazi u programima. Neki programi služe za ažuriranje stanja na skladištu, dok poseban program periodično ispituje stanje i generiše porudžbinu kada broj proizvoda padne na 50. Takvi programi se pokreću na korisnički zahtev.

Kod aktivnih sistema prestaje potreba za programom za periodično ispitivanje stanja na skladištu, jer ga menja pravilo u jedinstvenoj bazi pravila i podataka. To pravilo se automatski pokreće i omogućuje generisanje porudžbenice svaki put kada broj proizvoda na skladištu postane manji od 50. Svako pravilo definiše situaciju i akcije koje se preduzimaju sa nastankom situacije. Pravila u aktivnim bazama podataka su poznata kao ECA (*Event-Condition-Action*) produkciona pravila. Situacija se određuje u terminima događaja i uslova: „Kada se događaj pojavi, ukoliko je uslov zadovoljen (događaj i uslov određuju situaciju), izvršti akciju“. Događaji i akcije prethodno moraju biti definisani. Sva specifirana pravila čine model znanja.

Trigeri su specifična vrsta ECA pravila. Događaj je operacija ažuriranja baze podataka, uslov je proizvoljni SQL predikat, a akcija je sekvenca SQL naredbi. Sekvenca SQL naredbi se izvršava samo ako je detektovan

događaj (pokrenuta odgovarajuća operacija ažuriranja baze) i ako je uslov zadovoljen (*true*). SQL:1999 standard definiše trigere kao objekte šeme baze podataka, koji su vezani za tačno jednu baznu tabelu i koji se pozivaju svaki put:

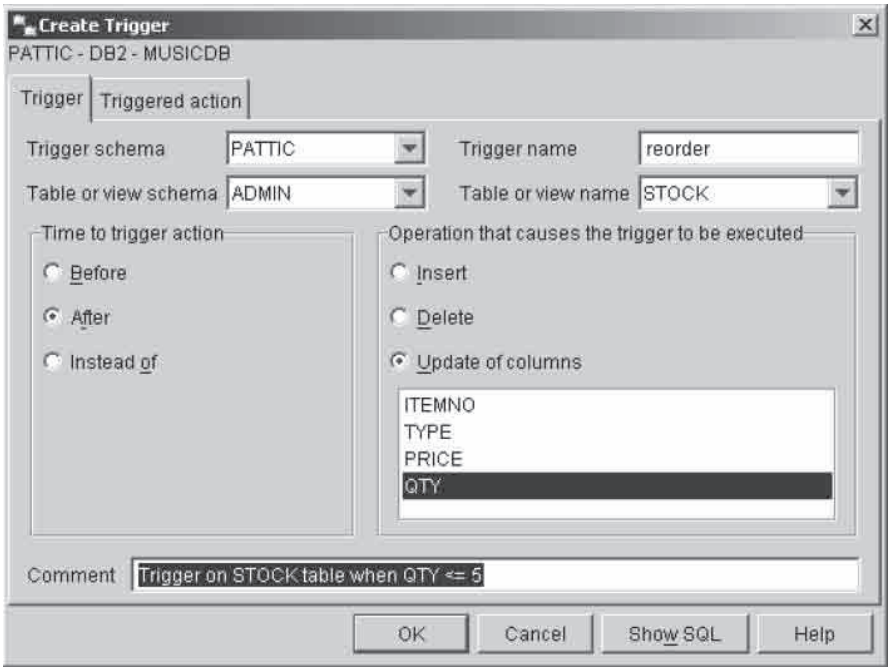
- kada se ubaci jedan ili više redova u tabelu za koju je definisan triger (INSERT naredba);
- kada se promeni jedan ili više redova u tabeli za koju je definisan triger (UPDATE naredba) ili
- kada se obriše jedan ili više redova iz tabele za koju je definisan triger (DELETE naredba).

Trigeri mogu biti kategorizovani na više načina:

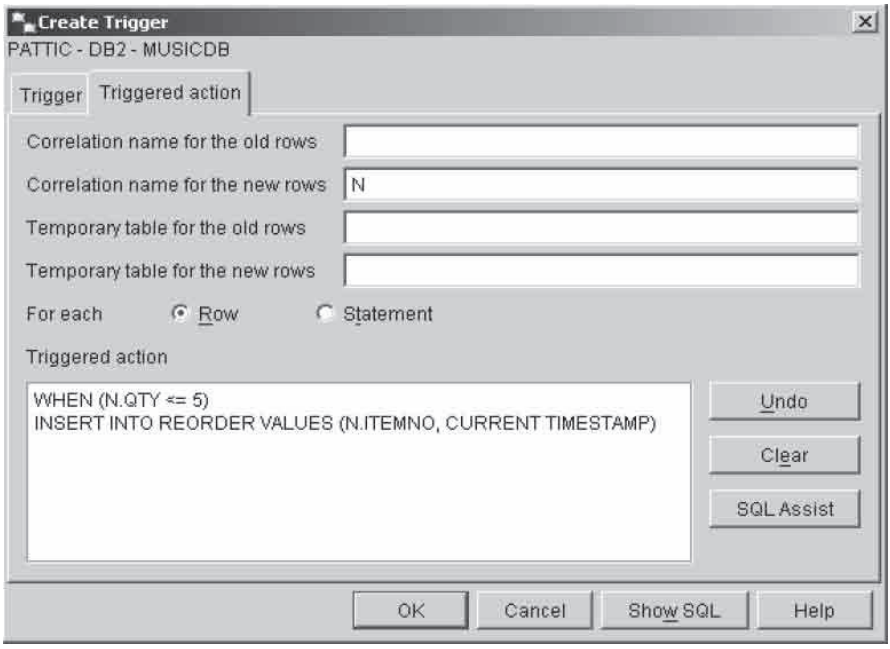
- trigeri mogu da se pozivaju neposredno pre (BEFORE) ili neposredno posle (AFTER) izvršavanja SQL naredbe za koju je definisan triger;
- u zavisnosti od toga koji od tri tipa operacija ažuriranja baze predstavlja događaj koji izaziva pokretanje trigera, mogu biti INSERT, UPDATE ili DELETE trigeri;
- triger se može izvršiti jednom za operaciju ažuriranja koja ga pokreće ili jednom za svaki red koji je ažuriran operacijom koja pokreće triger. U skladu sa time razlikuju se trigeri koji se pokreću na nivou naredbe i oni koji se pokreću na nivou reda.

Trigeri mogu da podrže sledeća poslovna pravila:

- validacija ulaznih podataka;
- automatsko generisanje vrednosti za ubačeni red;
- čitanje iz drugih tabela;
- pisanje u druge tabele;
- održavanje poslovnih pravila itd.




Slika 5.37. Kreiranje trigera kod DB2 baze [6]




Slika 5.38. Kreiranje akcije trigera [6]

**Sloj podataka (*data layer*)** jednog rešenja se sastoji od skladišta i servisa podataka. Skladište podataka je obična baza podataka u kojoj su podaci organizovani i skladišteni. Servisi podataka izvršavaju osnovne zadatke kao što su kreiranje, izvlačenje, ažuriranje i brisanje podataka. Tokom logičkog dizajna, projektni tim je odredio objekte i atribute koji će biti uskladišteni u bazu podataka, načine pristupa, obrade i pretraživanja. U toku fizičkog dizajna, tim kreira šemu baze podataka koja prikazuje tabele, relacije, tipove polja podataka, indekse, procedure i planove za migraciju podataka, *backup*, oporavak baze podataka i odgovarajuće podrške. Pitanje performansi bi trebalo razmatrati kroz čitav ciklus razvoja.

Tehnike koje se najčešće koriste za optimizovanje pristupa podacima su:

- **Indeksiranje podataka** – indeks (*index*) je uređena lista redova u tabeli koju DBMS može da koristi kako bi ubrzao operacije pretraživanja. Indeks je struktuiran kao stablo i održava sortiranu listu određenog skupa podataka. Upiti (*query*) koji se izvršavaju nad indeskiranim podacima su mnogo brži i efikasniji. DBMS koristi indekse kako bi brzo vodio upit do direktne lokacije zahtevanih podataka. Mogu se koristiti dva tipa indeksa i to klasterovani i neklasterovani indeksi.

  Klasterovani indeks fizički pamti redove podataka u tabeli kako bi složio redosled indeksa. Klasterovani indeks je obično definisan kao primarni ključ tabele. Jedino ograničenje klasterovanih indeksa je to što usporavaju operacije upisivanja jer redovi moraju biti češće preuređivani. Neklasterovani indeks održava malu tabelu informacija o indeksima o koloni ili grupi kolona. Jedina razlika između klasterovanog indeksa (*primary key*) i neklasterovanog (*unique constraint*) je ta što primarni ključ ne dozvoljava *null* vrednost (prazno polje), dok *unique constraint* dozvoljava jednu *null* vrednost, dok su sve vrednosti različite.

- **Particionisanje podataka** – kada je broj zapisa u tabeli toliko velik da usporava pristup podacima, onda je neophodno izvršiti particionisanje tabele. Mogu se implementirati horizontalno ili vertikalno particionisanje velikih tabela kako bi se ubrzala brzina obrade podataka. Kod horizontalnog particionisanja (*horizontal partitioning*) tabela koja sadrži mnogo redova se deli u više tabela koje sadrže isti broj kolona. Međutim, svaka tabela sadrži određeni

podskup podataka. Na primer, jedna tabela može da sadrži sve nazive klijenata koji počinju od slova A do M, dok druga tabela može da sadrži nazive klijenta čiji nazivi počinju od N do Ž. Vertikalno particionisanje deli tabelu koja sadrži mnogo kolona u više tabela.

- **Normalizacija podataka** – podrazumeva poboljšanje logičkog modela baze podataka kako bi se eliminisali duplicirani podaci u bazi. Normalizacija obično podrazumeva deljenje baze podataka u dve ili više tabele i definisanje relacija između njih. Normalizacija baze podataka minimizira dupliciranje informacija, smanjuje nekonzistentnost podataka i ubrzava menjanje podataka preko operacija ubacivanja, ažuriranja i brisanja.

Važan korak u planiranju baze podataka je **integritet podataka** koji obezbeđuje tačnost i konzistentnost podataka. Mogu se implementirati tri tipova integriteta podataka u bazama podataka:

- **integritet domena (*domain integrity*)** – određuje skup validnih vrednosti podataka za kolonu i određuje da li su dozvoljene *null* vrednosti.
- **integritet objekta (*entity integrity*)** – zahteva da svaki red u tabeli ima jedinstveni identifikator, tj. vrednost primarnog ključa.
- **referencijalni integritet (*referential integrity*)** – osigurava da se veze između primarnog i spoljnjeg ključa uvek održavaju. Razmatra se šta treba da se uradi ukoliko primarni ključ mora da bude izmenjen ili obrisan.
- **validacija podataka (*data validation*)** – osiguravaju korektnost i validnost podataka. Mogu se koristiti sledeće metode za proveru podataka:
  - provera opsega (*range checking*) – proverava da li su vrednosti podataka unutar granica koji su određeni funkcionalnom specifikacijom rešenja.
  - provera formata podataka (*data format checking*) – proverava da li su podaci postavljeni na odgovarajući format, na primer format telefonskog broja, format valute (*currency*) i sl.
  - provera tipa podataka (*data type checking*) – proverava da su podaci odgovarajućeg tipa (*string*, *date*, *decimal*, *integer* i sl.).

Validacija podataka se može vršiti na klijent ili server strani. Provera podataka na klijent strani (skriptovima, kontrolama korisničkog interfejsa, programima i sl.) sprečavaju upisivanje nevalidnih podataka u bazu.

Integritet podataka se može postići sledećim funkcionalnostima baze podataka:

- tipovi podataka (*data types*) – postavljanje odgovarajućih tipova podataka na polja u bazi podataka obezbeđuje da se nekorektni tipovi podataka ne mogu dodati u tabelu.
- difoltne vrednosti (*default values*) – određuju vrednosti polja pre nego što je korisnik izmeni.
- pravila validacije podataka (*data validation rules*) – određuju se prihvatljive vrednosti podataka, kao što su dužina polja, ulazne maske ili opseg vrednosti.
- ključevi (*keys*) – mnoge baze podataka automatski nadgledaju referencijalni integritet između tabela.
- okidači (*triggers*) – triger je skup programskih iskaza koji se definišu za određenu tabelu. Kada se određena akcija (na primer, ubaci (*insert)*, promeni (*update)* ili briši (*delete*)) dogodi u toj tabeli, trigeri automatski aktiviraju te iskaze. Trigeri mogu i verifikovati podatke u drugim tabelama ili izvršavati automatsko ažuriranje različitih tabela.
- uskladištene procedure (*stored procedures*) – predstavlja imenovanu kolekciju SQL iskaza koje su uskladištene u DBMS-u. Moguće je napisati jedan deo koda koji izvršava jednu akciju, zatim je skladištiti i pozivati je onoliko puta koliko je potrebno.
- skriptovi (*scripts*) – skript je manji programski kod u tekstualnom formatu koji može biti interpretiran u raznim programima. Svaki skript se aktivira van memorijskog prostora DBMS-a.

Servisi podataka se takođe mogu koristiti za implementaciju integriteta podataka. Komponente implementiraju servise podataka unutar jedne aplikacije. Komponente su izvršne kolekcije koda koje se nalaze na serveru ili unutar nekog drugog programa. Komponente su slične skriptovima samo što omogućavaju užu integraciju okruženja razvoja sa bazom podataka i programskim kodom.

Tokom identifikovanja zahteva, projektni tim je definisao i mnoga poslovna pravila koji predstavljaju logiku rešenja. Nakon postavljanja pravila integriteta podataka, tim mora da preispita zahteve u odnosu na date kriterijume.

### Projektovanje bezbednosti

Sistem mora biti projektovan za bezbednost, a ne dorađivan za bezbednost! Tokom svih faza razvoja sistema treba preduzimati odgovarajuće mere bezbednosti (Tabela 5.6).

Veoma je važno da organizacija bude svesna svih bezbednosnih slabosti kako bi se bolje zaštitila. Tipične slabe tačke sistema su:

- **slabe šifre (*weak passwords*)** – korisnici obično biraju šifre koje se lako pamte, kao što su nazivi njihovih voljenih ili bilo šta vezano za njihove lične podatke. Zlonamerni napadači lako mogu da pogode tačnu šifru i pristupe ne samo računaru, već i čitavoj mreži na koju je računar povezan.

Tabela 5.6. Mere bezbednosti po fazama MSF procesnog modela

| Faza | Mere bezbednosti |
|---|---|
| Definisanja vizije | Prikupiti zahteve za bezbednošću. Razmatraju se poverljivi podaci i operacije, privilegije svakog korisnika i način kako aplikacija trenutno upravlja tim bezbednosnim zahtevima. Zahevi za bezbednošću su izlistani u dokumentu zahteva. |
| Planiranje | Pravi se model pretnje. U funkcionalnoj specifikaciji se predlažu mere bezbednosti koje umanjuju identifikovane rizike. |
| Razvoj | Implementira se bezbednost koja je identifikovana u funkcionalnoj specifikaciji. |
| Stabilizacija | Upravlja se testiranjem bezbednosti. |
| Uvođenje | Prate se bezbednosne pretnje aplikaciji. |

- **nepravilno konfigurisan softver (*misconfigured software*)** – ukoliko su servisi softvera konfigurisani tako da koriste lokalni nalog ili dopuštaju više nego što se zahteva, napadači mogu lako da iskoriste servise kako bi pristupili sistemu ili izvršili zlonamerne akcije. Nepravilno konfigurisan softver se može podeliti u tri kategorije:

- mreža – Internet veze, *firewalls*;
- aplikacija – konfigurisani fajlovi;
- host – servisi, portovi, protokoli.

- **prenos nekriptovanih podataka (*unencrypted data transfer*)** – većina Web aplikacija prihvataju korisnički input, obrađuju korisničke zahteve i vraćaju odgovor korisniku. Ukoliko se podaci šalju na Web server i vraćaju korisniku u čistom tekstualnom formatu, onda postoji mogućnost da podaci, tokom prenosa, budu uhvaćeni, pročitani i izmenjeni od strane napadača. Ukoliko su podaci koji sadrže poverljive informacije, na primer poverljivi brojevi prodaje, informacije o autentifikacji, poslati bez enkripcije, napadač lako može da im pristupi i da iskoristi za obavljanje zlonamerne akcije na Web aplikaciji ili organizaciji.

- ***buffer overflow*** – tip greške koji se dešava kada se više podataka upisuje u bafer koji nije programiran da sadrži toliki broj. Na primer, ukoliko korisnik pozove funkciju sa stringom od 260 karaktera, koja treba da se kopira u bafer u koga može da stane 256 karaktera tada se šalje tip greške „*buffer overflow*“. Zlonamerni korisnici ispituju aplikacije ne bi li pronašli način da prouzrokuju „*buffer overflow*“ jer to mogu iskoristiti za:
  - obaranje aplikacije ili operativnog sistema;
  - pronalaženje još bezbednosnih slabosti čitajući poruke o greškama (developeri obično pišu poruke o greškama u procesu debagovanja aplikacije koje ukazuju na trag o grešci, na primer može da oda lokaciju fajla o šiframa i sl.);
  - dobijanje pristupa memoriji sistema koji se zahteva da bi se pokrenuo kod programa.
- **Ubacivanje koda (*code injection*)** – podrazumeva ubacivanje zlonamernog koda koji u zavisnosti od prirode napada mogu da se pokrenu na Web serveru, serveru baze podataka ili na klijentu. Tipični tipovi napada ubacivanjem koda su ubacivanjem SQL iskaza, unakrsno skriptovanje i ubacivanjem koda preko *buffer overflows*. Ukoliko developeri dinamički prave SQL iskaze preko korisničkog inputa, tada napadači mogu da modifikuju SQL iskaze. Kod unakrsnog skriptovanja, napadači ne napadaju Web server već druge

korisnike na Web serveru tako što izvršavaju skript na drugim korisničkim računarima koji mogu dalje pristupati kolačićima (*cookies*) ili aktivirati druge skriptove.

**Kreiranje modela pretnje** uključuje sledeće zadatke:

- Organizovati *brainstorming* sastanke kako bi se identifikovale potencijalne pretnje;
- Izlistati sve moguće pretnje;
- Primeniti STRIDE kategorije bezbednosti (nakon što se kreira inicijalna lista mogućih pretnji bezbednosti, pomoću STRIDE modela treba kategorizovati pretnje). STRIDE model je tehnika za identifikovanje i kategorisanje pretnji aplikacije:
  - Prevara identiteta (**S***poofing identity*) – napadač krade ili pogađa šifru korisnika; šalje e-mail poruku u tuđe ime ili IP prevara gde napadač koristi IP adresu poverljivog izvora.
  - Podmićivanje (**T***ampering)* – kada korisnik neautorizovano pristupa računaru i zatim menja operacije, konfiguraciju ili podatke.
  - Poricanje (**R***epudiation)* – kada sistem administrator ne može da dokaže da je korisnik zlonameran ili da je izvršio neku akciju.
  - Obelodanjena informacija (**I***nformation disclosure*) – kada neautorizovani korisnik može da pregleda privatne podatke, kao što su fajlovi koji sadrže brojeve kreditnih kartica i njihove datume isteka.
  - Napad uskraćivanjem usluga (**D***enial-of-service* - DoS) – može da prouzrokuje da aplikacija ili operativni sistem prestane da radi, da je CPU angažovan predugo, sistemska memorija se troši što dovodi do usporavanja funkcionalnosti aplikacija i operativnog sistema, smanjen je opseg mreže ili je potpuno ugušen saobraćaj na mreži.
  - Visina privilegije (**E***levation of privilege*) – kada korisnik dobija prava pristupa koja su veća i od privilegija koje administrator ima.
- Pisati komentare za svaku pretnju (tip pretnje, uticaj napada na organizaciju u odnosu na troškove i uložen napor, moguće tehnike

koje će napadač koristiti, mesto dešavanja napada i moguće tehnike za umanjenje napada)

- Voditi istraživanje;
- Rangirati rizike pretnji.

Nakon što se identifikuju pretnje, sledeći korak je da se pronađe način kako da se te pretnje umanje. Na pretnje se može odgovoriti na nekoliko načina:

- informisati korisnike o potencijalnim pretnjama;
- ukloniti funkcionalnosti rešenja koje dovode do većih rizika;
- identifikovati tehnike umanjenja:
  - autentifikacija i autorizacija – autentifikacija je proces verifikovanja korisnika, dok je autorizacija proces određivanja pristupa resursima;
  - bezbedna komunikacija – koristiti *Secure Sockets Layer* – SSL (koji uspostavlja kriptovani komunikacioni kanal između klijenta i servera); *Internet Protocol Security* – IPSec (obezbeđuje podatke koji se šalju između dva računara, kao na primer servera aplikacija i servera baze podataka); *Virtual private networks* – VPNs (omogućava da se uspostavi *point-to-point* IP prenos preko Interneta);
  - kvalitet usluga (*Quality of Service* - QoS) – mehanizam koji omogućava sistem administratorima da dodele veće prioritete određenim tipovima mrežnog saobraćaja;
  - gušenje (*throttling*) – ograničava broj poruka koji se šalje ka vašem sistemu;
  - praćenje (*auditing*) – proces prikuplja informacije o korisničkim aktivnostima i važnim događajima i skladišti te informacije radi dalje analize. Praćenje je poznatije kao *logging*. *Windows event logs* omogućava vašoj aplikaciji da snima informacije o važnim događajima.
  - filtriranje (*filtering*) – proces hvatanja i ispitivanja podataka se šalje do vašeg sistema gde vi možete da odlučite dali da prihvatite ili odbijete podatak.
  - najniže privilegije (*least privilege*) – korisnicima se daje minimalni nivo privilegije koji im omogućava samo da završe svoje poslove.

**Politika bezbednosti (*Security policy*)** definiše organizacione zahteve za bezbednim korišćenjem računara i mreže. Uključuje procedure za detektovanje, prevenciju i odzive na napade i obezbeđuje okvir (*framework*) za implementaciju plana i procedura bezbednosti aplikacije. Politika bezbednosti određuje šta vaša aplikacija i njeni korisnici smeju da urade.

Prvi korak kod projektovanja bezbedne aplikacije je pisanje sigurnog koda. Kôd koji pristupa samo memoriji za koju je autorizovan da pristupa se zove **bezbedan kôd (*type-safe code*).** Na primer, bezbedan kôd ne pristupa vrednostima privatnih polja drugih objekata.

Drugi način da se zaštitimo je **potpisivanje koda (*code signing*)** čime se osigurava autentičnost i integritet koda. .NET *Framework* podržava autentične digitalne sertifikate (a*uthenticode digital certificates)* i potpise sa jakim imenom (*strong name signatures*).

**Potpisivanje podataka (*data signing*) i enkripcija (*encryption*)** su procesi koji se koriste u zaštiti sadržaja podataka. Enkripcija je proces prikrivanja podataka pre nego što se pošalju ili uskladište. Pre nego što se podaci šifruju, njihov sadržaj je u punom tekstu (*plaintext*), a nakon enkripcije tekst se naziva šifrirani tekst (*ciphertext*). Šifrirani tekst je totalno nečitljiv. Dekripcija je proces prevođenja šifriranog teksta u čitljiv (*plaintext*). Procesi enkripcije i dekripcije podataka se oslanjaju na tehnike hešinga i potpisivanja podataka. Hešing (*hashing*) je proces uparivanja podataka bilo koje dužine na fiksnu dužinu sekvence bajtova tzv. heš (*hash*). Heš je dobijen primenom matematičke funkcije koje se zove hešing algoritam na određenu količinu podataka. Potpisani podatak (*signed data*) je standardni tip podataka koji se sastoji od bilo kog tipa sadržaja kombinovanog sa enkriptovanim heševima za nula ili više potpisnika. Heševi se koriste za potvrđivanje identiteta potpisivaoca podatka i da poruka nije bila menjana tokom njenog potpisivanja.

**Bezbednost uloga (*role-based security*)** pomaže kod rešavanja prevare identiteta tako što omogućava verifikovanje identiteta i uloge (role) korisnika.

### Kompletiranje faze planiranja

Faza planiranja obuhvata veći deo arhitekture i dizajna rešenja. Tokom faze planiranja nekoliko elemenata utiču na projektovanje rešenja:

- **Skalabilnost** – sposobnost proširivanja resursa kako bi se povećao kapacitet rešenja. Na primer, povećanje memorije, premeštanje

aplikacije na moćniji računar ili distribuiranje aplikacije na više računara tako da se održi transparetnost lokacije.

- **Raspoloživost** – mera koliko često je aplikacija raspoloživa da rukuje zahtevima. Ukoliko rešenje treba da radi 24 časa dnevno, 7 dana u nedelji, onda treba da bude projektovano za visoku raspoloživost.
- **Pouzdanost** – sposobnost aplikacije da obezbedi tačne rezultate.
- **Performanse** – može se definisati i kao mera odziva sistema ili kao odnos broja transakcija i upotrebe resursa.
- **Interoperativnost** – sposobnost sistema da uspešno radi u heterogenim računarskim okruženjima.
- **Globalizacija i lokalizacija** – koriste se za razvoj aplikacija koje će se koristiti širom sveta. Globalizacija podrazumeva proces projektovanja i razvoja aplikacija koje mogu da rade u različitim kulturama i lokacijama. Globalizacija uključuje:
  - identifikovanje kultura i jezika koje rešenje mora da podrži;
  - dizajniranje dodatnih funkcionalnosti koji će podržati te kulture;
  - pisanje koda koji će adekvatno da se aktivira u svim kulturama i na svim jezicima.

  Lokalizacija je proces prilagođavanja globalizovanih aplikacija određenoj kulturi.

Važan deo rešenja su i administrativne funkcionalnosti:

- praćenje (*monitoring*) – obezbeđuje da aplikacija funkcioniše korektno i da se izvršava na optimalnom nivou. Elementi plana praćenja su resursi, detekcija greške, praćenje performansi, alata i dr.
- migriranje podataka (*data migration*) – kako će podaci migrirati sa postojećeg na novo rešenje. Elementi plana migracije su strategije migracije, smernice migracije, testiranje okruženja, alati, proces migracije i plan povratka na prethodnu konfiguraciju (*rollback plan*).
- specifikacije licenciranja (*licensing specifications*) – neophodno je obezbediti licence za sve korisnike koji će instalirati i aktivirati aplikacije.

U toku faze planiranja, neophodno je napisati planove aktivnosti koji će se izvršiti u narednim fazama projekta:

- **plan za fazu razvoja** – pre nego što razvojni tim počne da razvija rešenje, veoma je važno da se verifikuje da je infrastruktura spremna za razvoj, a okruženje za testiranje rešenja. Elementi plana razvoja uključuju: ciljeve razvoja, matrice kompromisa, proces izgradnje, kontrolu verzija i izvornog koda, strategiju isporuke, ciljeve projektovanja, obučavanje razvojnog tima i standarde i najbolje prakse;
- **plan za fazu stabilizacije** – plan testiranja opisuje strategiju planiranja, organizovanja i upravljanja aktivnostima testiranja projekta. Elementi plana testiranja uključuju: testiranje koda, baze podataka, infrastrukture, bezbednosti, integracije, prihvatljivosti i upotrebljivosti rešenja, performanse i testiranje kapaciteta.
- **plan faze uvođenja** – u okviru plana uvođenja diskutuju se faktori koji mogu da utiču na uspešno uvođenje rešenja. Elementi plana uvođenja uključuju: cilj uvođenja, arhitektura, resursi uvođenja, koordinacija obučavanja, raspored uvođenja, podrška rešenja, proces instalacije sajta i dr.

Da bi započeo kreiranje rešenja, razvojni tim koristi dokument tehničke specifikacije. Tehnička specifikacija je skup dokumenata koji uključuju fizički dizajn, kao što su specifikacije klasa, modela komponenti, topologije mreže i komponenti i sl. Elementi dokumenta tehničke specifikacije su:

- pregled arhitekture – opisuje se arhitektura koja će biti implementirana rešenjem;
- objektni model – opisuje model objekata rešenja. Uključuje opise svih objekata i njihovih funkcionalnosti;
- interfejsi – sadrži kôd i detalje svakog interfejsa u rešenju;
- kôd – opisuje operacije svakog metoda u rešenju;
- kodovi greške – opisuje kodove greški koji se koriste za rukovanje greškama u rešenju;
- praćenje grešaka – opisuje kako će se različite greške pratiti i upravljati;
- konfiguracija – opisuje kako će rešenje biti registrovano na destinacionim računarima.
- dokumentacija podrške – npr., funkcionalna specifikacija i dr.

## 5.6. Stabilizacija i uvođenje rešenja

Nakon što je završena faza razvoja rešenja, projekat ulazi u faze stabilizacije i uvođenja (*deploying*). Cilj faze stabilizacije je da se poboljša kvalitet rešenja i da ga stabilizuje pre nego što se pusti u proizvodno okruženje. Faza uvođenja uvodi rešenje u proizvodno okruženje.

Dva glavna zadatka u fazi stabilizacije su:

- Testiranje rešenja – tim implementira planove testiranja koji su kreirani tokom faze planiranja i koji su testirani tokom faze razvoja;
- Vođenje pilot rešenja – tim testira pilot rešenje sa aktuelnim korisnicima i na realnim scenarijima.

Izlazna dokumenta iz faze stabilizacije su:

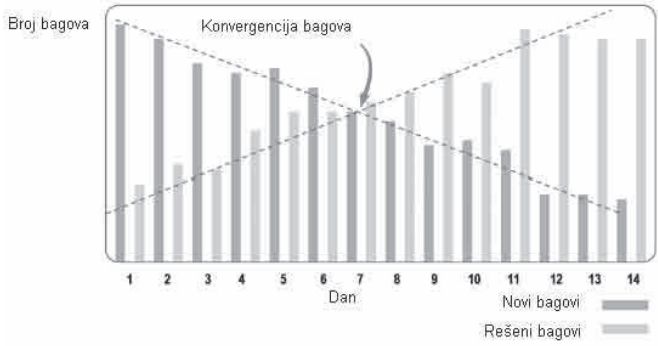
- Pregled pilot rešenja
- Gotove verzije:
  - izvornog koda
  - skriptova i instalacione dokumentacije
  - korisničkog *help-a* i priručnika za obuku
  - dokumentacije za rukovanje
- Testiranje i izveštaj o bagovima
- Dokumenti projekta

U toku faze stabilizacije, svaka rola u projektnom timu ima određene odgovornosti:

- proizvodni menadžer – prati plan izvršenja i lansiranja proizvoda;
- program menadžer – prati projekat i uspostavlja prioritete među bagovima;
- razvojni tim – razrešava bagove i optimizuje kôd;
- tim za testiranje – testira i izveštava o statusu bagova;
- menadžment puštanja u rad – postavlja pilot rešenje, planira uvođenje rešenja i vodi obučavanje;
- iskustveni korisnici – stabilizuju materijale obučavanja, asistencije i performanse korisnika.

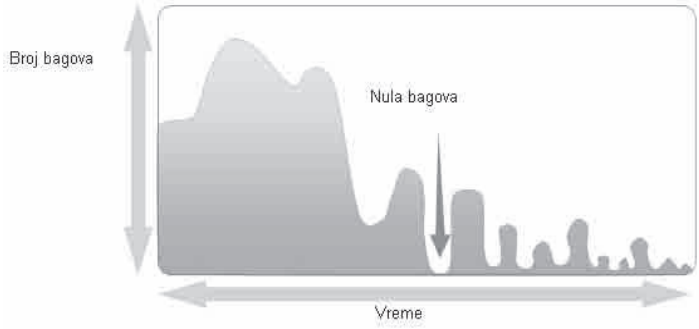
Koraci faze stabilizacije su:

- **konvergencija bagova (*bug convergence*)** – je tačka u kojoj broj bagova koji se pronalazi je isti broju bagova koji se razrešava. To je tačka u kojoj je vidljiv napredak rešenja u odnosu na broj bagova. Nakon konvergencije bagova, broj bagova treba da nastavi da se smanjuje sve do nula bagova.




Slika 5.39. Konvergencija bagova [9]

- **nula bagova (*zero-bug*)** – je tačka u projektu kada razvojni tim razreši sve bagove koji su se pojavili u toku testiranja.




Slika 5.40. Nula bagova [9]

- **faza pred puštanja u rad** – na osnovu povratne informacije od korisnika, projektni tim nastavlja da poboljšava proizvod i razrešava bagove koji su se pojavili u toku pilot projekta.
- **zlatno puštanje** – je puštanje rešenja u rad. Zlatno puštanje je određeno kombinacijom nula defekata i kriterijuma uspešnosti.

Bag (*bug*) ne podrazumeva obavezno grešku, već se može definisati kao bilo koja pojava nastala korišćenjem proizvoda (Slika 5.41). Bagovi treba

da se sagledaju i obrade saglasno prioritetima isporuke proizvoda. Upravljanje bagovima (*bug management*) je proces tretiranja bagova kako bi se uspostavio željeni odnos između definisanog nivoa kvaliteta proizvoda u projektu i kvaliteta proizvoda u realizaciji. Proces praćenja bagova (*bug tracking process*) izveštava, klasifikuje bagove, određuje prioritete i rešava bagove (Slika 5.42).

Slika 5.41. Pojam baga [2]

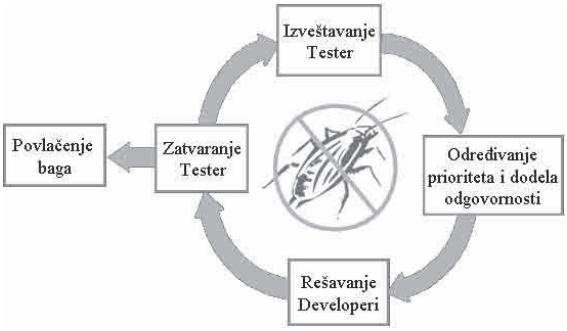
Bagovi se mogu klasifkovati na dva standardna načina:

- klasifikacija prema uticaju (*by severity*) - meri uticaj baga na proizvod:
  - Uticaj 1: Pad sistema (*system crash*) – ova vrsta bagova prouzrokuje pad sistema ili gubitak podataka;
  - Uticaj 2: Glavni problem (*major problem*) – ova vrsta bagova dovodi do ozbiljnih oštećenja u programu, ali ne narušava sistem;
  - Uticaj 3: Manji problem (*minor problem*) – ova vrsta bagova dovodi do narušavanja funkcionalnosti programa;
  - Uticaj 4: Trivijalni (*trivial*) – beznačajne greške, trivijalne.
- klasifikacija prema prioritetu (*by priority*) – sortira bagove prema prioritetu rešavanja:
  - Prioritet 1: Najveći prioritet (*highest priority*) – bagovi klasifikovani pod najvećim prioritetom podrazumeva da rešenje nije isporučivo;
  - Prioritet 2: Visoki prioritet (*high priority*) – bagovi klasifikovani pod visokim prioritetom takođe podrazumevaju da rešenje nije spremno za isporuku;

- Prioritet 3: Srednji prioritet (*medium priority)* – srednji prioritet bagova ukazuje da je rešenje isporučivo;
- Prioritet 4: Nizak prioritet (*low priority*) – proizvod je isporučiv jer su bagovi najnižeg prioriteta.

Nivo kvaliteta svake verzije rešenja se opisuje kroz klasifikaciju bagova.

Proces testiranja je projektovan na način da identifikuje i ukaže na potencijalne teme koje su od prioriteta za razvoj rešenja. Testiranjem se potvrđuje da su komponente rešenja ispunile planirane ciljeve projekta u predviđenom roku i sa odgovarajućim kvalitetom. Merenje da li je projekat uspešan je gotovo nemoguće bez nečega sa čime bi se merili rezultati projekta. Kreiranje kriterijuma uspešnosti uključuje definisanje uslova pod kojim će predloženo rešenje dostići ciljeve. Na primer, „Sveukupno zadovoljstvo klijenta sa novim modulom za obradu porudžbina na Web sajtu, biće najmanje 82%“. Ponekad se kriterijumi uspešnosti nazivaju indikatori ključnih performansi (*Key Performance Indicators* - KPI).




Slika 5.42. Proces praćenja bagova

Bez sistema za praćenje bagova, nemoguće je proceniti da li su kriterujmi nula bagova ispunjeni. Promenljive koje se prate su:

- Ponovljivost, repetivnost (*repeatability*) – meri koliko se često ponavlja jedan bag. Predstavlja procenat vremena u kome se bag pojavljuje.
- Vidljivost (*visibility*) – meri situaciju koja mora da se dogodi kako bi se bag pojavio. Na primer, ukoliko se bag događa jedino onda kada korisnik drži *Shift* taster dok pritiska desnik klik miša i pregleda *File* meni, onda je procenat nevidljivosti visok.

- Uticaj (*severity*) – meri koliki je uticaj baga na rešenje, kôd ili korisnike. Na primer, kada se bag dogodi, da li to utiče da se operativni sistem ili aplikacija blokiraju ili utiče na to da se promeni piksel boje na meni baru? Da bi mogla da se odredi uticaj bagova, projektni tim treba da definiše listu problematičnih situacija. Na primer, ukoliko sistem prestane da se odaziva i postoji gubitak podataka onda je promenljiva uticaja 10.

Ove tri promenljive, prethodno spomenute, se procenjuju i pridružuju svakom pojedinačnom bagu kako bi se utvrdila prioritetnost. Prioritet baga se može izračunati prema sledećoj formuli:

(PONOVLJIVOST + VIDLJIVOST) * UTICAJ = PRIORITET

Tipični načini rešavanja bagova su:

- ispravljen (*fixed*) – podrazumeva da je developer popravio bag, testirao prepravku, proverio kôd, dodelio prepravci verziju koda gde je prepravka izvršena i vratio bag testeru koji ga je prijavio;
- dupliciran (*duplicated)* – znači da je prijavljeni bag duplikat nekog drugog baga koji je već prijavljen u bazi. Novi bag je rešen, zatvoren i pokazuje na originalni bag;
- odložen (*postponed)* – znači da se bag odlaže za narednu verziju programa. Ova odluka se donosi ukoliko je tim ograničen vremenom, a bag nije visokog prioriteta;
- prema projektu (*by design)* – podrazumeva da je prijavljeno ponašanje namerno postavljeno jer je traženo u funkcionalnoj specifikaciji;
- ne reprodukovanje *(can't reproduce)* – developeri ne mogu da nađu postojanje baga ni na jednom nivou;
- ne ispravljanje *(won't fix)* – znači da developeri neće popraviti bag u trenutnoj verziji, jer smatraju da nije vredan truda.

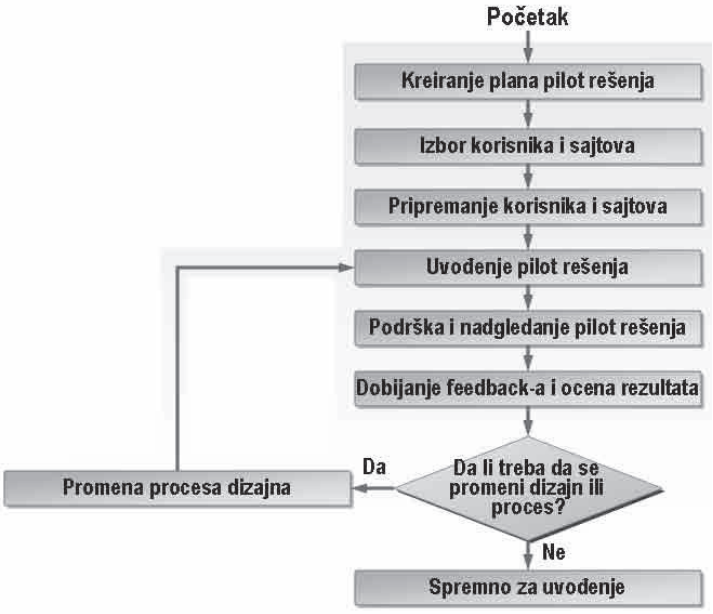
Danas, jedan od popularnijih softvera za praćenje defekata je BUGZILLA *defect tracking software*.

### 5.6.1. Vođenje pilot rešenja

Pilot podrazumeva testiranje rešenja u proizvodnom okruženju i njegovo isprobavanje od strane krajnjih korisnika, osoblja za podršku i instalaciju rešenja. Primarna svrha pilot rešenja je da pokaže da projekat radi u proizvodnom okruženju kao što se i očekivalo i da ispunjava organizacione zahteve. Sekundarna svrha je da se pruži šansa razvojnom timu da poboljšaju proces razvoja. Pilot nije završen sve dok tim nije siguran da je svaka komponenta spremna za uvođenje u sistem.

Pilot je prilika korisnicima da pruže povratnu informaciju o tome kako rade funkcionalnosti rešenja što ujedno pomaže timu u određivanju nivoa podrške koji je neophodan nakon potpunog uvođenja rešenja u sistem.

Pilot proces je iterativan. Započinje sa zadacima planiranja, kao što su kreiranje plana pilot rešenja i pripremanje korisnika i sajtova, a zatim nastavlja sa implementacijom pilot rešenja, procenom rezultata, rešavanjem problema i uvođenjem druge verzije, sve dok se ne postigne cilj i kvalitet koji ukazuje na spremnost potpunog uvođenja rešenja u sistem. Na slici 5.43 se prikazuju koraci planiranja i vođenja pilot rešenja. Faza stabilizacije se završava onda kada menadžment odobri dokument gotove verzije (*release readiness*) koji sadrži rezultate prethodnih zadataka rađenih u toku ove faze.




Slika 5.43. Koraci vođenja pilot rešenja

### 5.6.2. Faza uvođenja rešenja

Tokom faze uvođenja, tim uvodi tehnologiju rešenja, komponente, stabilizuje uvođenje, usmerava projekat na operacije i podršku i dobija finalno odobrenje projekta od klijenta. Glavni zadatak ove faze je pripremanje i uvođenje rešenja. Izlazna dokumentacija u ovoj fazi su:

- informacioni sistemi operacija i podrške;
- procedure i procesi;
- baza znanja i izveštaji;
- repozitorijum svih verzija dokumenata, konfiguracija, skriptova i koda;
- završni izveštaj projekta
  - dokumenta projekta;
  - nadzor klijenta;
  - naredni koraci.

Nakon faze stabilizacije, projektni tim može da migrira na fazu uvođenja na razne načine. Jedna od opcija je da iskoristi organizacioni operativni tim na uvođenju rešenja. Ukoliko operativni tim uspe da upravlja celokupnim procesom uvođenja, onda obično predstavnici iz razvojnog tima ostaju na projektu kako bi umanjili potencijalne probleme tokom transfera. Druga opcija je da se kombinuju članovi iz svake grupe i da se kreira odvojeni razvojni tim. Tokom faze uvođenja, odgovornosti pojedinačnih rola su sledeće:

- proizvodni menadžment – povratne informacije od klijenta; procena; odobravanje;
- program menadžment – poređenje rešenja i cilja; upravljanje stabilizacijom;
- razvoj – rešavanje problema; podrška;
- testiranje – performanse testiranja; identifikovanje, definisanje, rešavanje i izveštavanje o problemima;
- menadžment puštanja rešenja – upravljanje uvođenjem;
- iskustveni korisnik – obučavanje; upravljanje rasporedom obuke.

Kompleksnost i dužina faze uvođenja će zavisiti od samog projekta. U toku faze uvođenja neophodno je razmotriti zahteve za hardverom i operativnim sistemom koji uključuju servere preduzeća, centre podataka, XML Web servise, klijente (desktop, mobilni).

Uvođenje rešenja u proizvodno okruženje prolazi kroz nekoliko koraka:

- **planiranje uvođenja** – testiraju se, instaliraju i konfigurišu fizička infrastruktura, sistemski i aplikacioni softveri. Projektni tim poboljšava i ažurira sledeću dokumentaciju:
  - dijagrami uvođenja (*deployment diagrams*);
  - plan testiranja (*test plan*);
  - plan bezbednosti (*security plan*) - tim osigurava da je svo osoblje upoznato sa pravilima i standardima bezbednosti;
  - *backup* plan;
  - plan za analiziranje performansi i upotrebe sistema;
  - plan za rukovanje logovima i drugi administrativni zadaci;
  - plan oporavka od štete (*disaster recovery plan*) – šta se očekuje da će se desiti rešenju, opremi, osoblju i podacima ukoliko se šteta dogodi;
  - plan poslovne kontigencije (*business contingency plan*) – tim utvrđuje šta će se desiti ukoliko dođe do zastoja poslovnih aktivnosti;
  - informacije o obučavanju – tim mora da osigura da je osoblje adekvatno obučeno za održavanje i ažuriranje rešenja;
- **uvođenje komponenti** – primeri komponenti su: ruteri, serveri baza podataka, *mail* ruteri, serveri za udaljeni pristup (*remote access servers*), lokalni ruteri, fajl serveri, klijentske aplikacije i dr. Uvođenje komponenti podrazumeva izbor odgovarajuće strategije uvođenja (serijsko ili paralelno uvođenje), a zatim i izvršavanje uvođenja. Uvođenje podrazumeva i korišćenje sistema od strane korisnika.
- **period zatišja** – započinje kada se završi faza uvođenja. Period zatišja od 15 do 30 dana je neophodan da bi se generisali statistički podaci.
- **transfer projekta** – podrazumeva prenošenje operacija i funkcija podrške na osoblje.
- **završne aktivnosti** – podrazumeva praćenje zadovoljstva klijenta, pripremanje završnog izveštaja, završni pregled projekta i dobijanje odobrenja od klijenta.

# 6. Metodologija implementacije ERP rešenja

## 6.1. Proces razvoja ERP sistema

Proces projektovanja gotovog ERP rešenja se razlikuje od tradicionalnog procesa razvoja sistema. Proces razvoja ERP sistema obuhvata planiranje, analizu zahteva, projektovanje, detaljan dizajn, implementaciju i održavanje.

Planiranje započinje sa procenom potreba koje predstavljaju poslovnu opravdanost kupovine softvera. Procena potreba je veoma bitna usled investicije u ERP sistem i njegovog uticaja na celokupno poslovanje. Faza analize zahteva jednog ERP projekta obuhvata određivanje poslovnih procesa koji će biti podržani ERP paketom. U fazi projektovanja se uvode najbolje prakse koje podržava ERP sistem, što povlači reinženjering poslovnih procesa. Ovde se uočava osnovna razlika u odnosu na tradicionalni pristup razvoja sistema, gde projektanti definišu nove poslovne zahteve i implementiraju softver prema tim zahtevima. Jedna od osnovnih odluka u implementaciji ERP paketa je da li da se redizajniraju organizacioni poslovni procesi kako bi odgovarali softveru ili kastomizirati softver prema organizacionom poslovnim praksama.

#### Planiranje

Poslovna opravdanost uvođenja ERP rešenja obuhvata merljive i nemerljive koristi, kao što su smanjenje zaliha, smanjenje operativnih troškova, poboljšanje procesa, smanjeno vreme ciklusa i dr. Neka tehnološka i poslovna obrazloženja uvođenja ERP rešenja su prikazana u tabeli 6.1.

#### Analiza zahteva

Aktivnosti analize zahteva uključuju analiziranje poslovnih procesa i određivanje procesa koji će biti podržani ERP paketom. S obzirom da kompanija kupuje vendorov pogled najbolje prakse, veoma je važno da se izabere sistem koji najbolje odgovara organizacionim ciljevima i konkurentnoj strategiji.

Tabela 6.1. Obrazloženja za ERP

| Tehnološka obrazloženja | Poslovna obrazloženja |
|---|---|
| ▪ Mogućnost korišćenja pravovremenih podataka | ▪ Smanjenje troškova zaliha |
| ▪ Mogućnost integrisanja sistema umesto održavanja više posebnih sistema | ▪ Smanjenje IT troškova |
| ▪ Mogućnost povećanja sistema bez izlaganja troškovima i vremenu za kastimiziran razvoj i modifikaciju | ▪ Smanjenje troškova osoblja |
| ▪ Mogućnost implementacije novih funkcionalnosti, kao npr. CRM | ▪ Povećana profitabilnost |
| ▪ Pristup podacima *online* i u realnom vremenu | ▪ Poboljšanje produktivnosti |
| ▪ Redukcija u vremenu i troškovima razvoja i održavanja sistema | ▪ Bolje upravljanje gotovim novcem |

Većina vendora nudi najbolje prakse za određene industrije, kao što su na primer, farmaceutska industrija ili naftna industrija i sl. Proces odabira najboljeg ERP sistema se vrši čekiranjem (selektovanjem) aktivnosti.

Tabela 6.2. Selekcija jednog ERP sistema

| Postupak | Aktivnosti |
|---|---|
| Kreiranje vizije | Razviti korporativne ciljeve za ERP |
| Kreiranje liste funkcionalnosti | Angažovati tim koji je upoznat sa procesima u kompaniji; mapirati trenutne procese prema novoj najboljoj praksi. |
| Kreirati listu kandidatnih softvera | Ograničiti se na veličinu kompanije, tip industrije |
| Ograničiti se na 4 do 6 ozbiljnih kandidata | Sagledati preliminarnu analizu snage i slabosti |
| Kreirati zahtev za ponudama | Napraviti listu karakteristika i funkcionalnosti |
| Pregledati ponude | Razmotriti slabosti i jake strane svake ponude |
| Izabrati 2 ili 3 finalne | Neka vendori sa finalne liste demonstriraju njihove pakete |
| Izabrati pobednika | Razmotriti mnogobrojne faktore (npr., podrška, tehnološki rizik itd.) |
| Opravdati investiciju | Razmotriti analizu troškova i koristi; merljive i nemerljive koristi |
| Pregovaranje | Učestvovati u sastavljanju ugovora |
| Pokrenuti pilot projekat | Unakrsno funkcionalni tim treba da isproba pilot projekat |
| Potvrditi opravdanost | Uključiti unakrsno funkcionalni tim u odlučivanje tipa nastaviti ili ne sa projektom |

### Projektovanje

Ključno pitanje kod projektovanja ERP sistema je da li izvršiti reinženjering ili kastomizaciju. Kod reinženjering pristupa, tim odabira gotovo komercijalno ERP rešenje i redizajnira poslovne procese kako bi bili u saglasnosti sa paketom. Kod pristupa kastomizacije ili prilagođavanja, tim odabira komercijalni ERP kako bi ga kastomizirala prema jedinstvenim potrebama.

Tabela 6.3. Reinženjering vs. kastomizacija

| | Za | Protiv |
|---|---|---|
| Kastimizacija | Podržava jedinstvene poslovne procese; održavaju se strateški procesi | Neki ERP sistem možda neće podržati ove jedinstvene poslovne procese; prilagođavanje je teško, pošto su moduli integrisani; nadogradnja softvera sa novijom verzijom je veoma teško. |
| Reinženjering | Prednost generičkih procesa u industriji; najbolje prakse predstavljaju poboljšane promene procesa. | Ne podržava strateške ili jedinstvene poslovne procese; otpor se javlja kada nastupe važne organizacione promene. |

Najlakše je instalirati kompletno ERP rešenje, jer organizacije mogu da slede metodologiju koju propisuju vendori i koriste konsultante vendora, koji poseduju specijalizovanu ekspertizu. Ukoliko se ERP rešenje prilagođava organizaciji, tada se vreme i troškovi investirani u projekat, povećavaju srazmerno rizicima uspešne implementacije. Razlog tome je da se kastomiziran softver ne može lako integrisati sa novijim verzijama ERP paketa, koje s vremena na vreme isporučuje vendor.

Pojedine organizacije se opredeljuju za održavanje postojećih sistema i pridodavanje ERP modula koje podržavaju određene funkcije kao što su finansijsko računovodstvo ili CRM i sl. Ovakav vid implementacije u poređenju sa kompletnom implementacijom ERP rešenja je isplativ, ali je lišen funkcionalnosti koje nudi kompletna ERP implementacija. Jedina prednost ovakvog pristupa je da ne izaziva velike potrese u organizaciji, jer korisnici sistema ne moraju da usvajaju nove poslovne procese.

Drugi pristup u realizaciji ERP sistema je autsorsing, odnosno da sistem održava spoljni vendor. Odluke koje se donose zavise od pouzdanosti i stabilnosti vendora što čini organizaciju ranjivom.

### Detaljni dizajn

U fazi detaljnog dizajna projekta, tim odabira modele, procese i informacije koje će biti podržane sistemom. Metodologija „najbolje prakse“ obuhvata sledeće korake:

- Izbor primenjivih poslovnih procesa.
- Odbacivanje neprimenjivih poslovnih procesa.
- Ukoliko se poslovni procesi ne poklapaju sa sistemom, oni služe kao osnova za reinženjering.
- Identifikovati one oblasti koje nisu pokrivene od strane „najbolje prakse“ i koje će zahtevati razvoj kastomizovanih modela.

Detaljni dizajn uključuje interaktivno prototipovanje i uključenost korisnika u određivanju elemenata sistemskog dizajna.

### Implementacija

ERP implementacija obuhvata pitanja konfiguracije, migracije podataka sa starog sistema na novi sistem, izgradnju interfejsa, implementaciju izveštaja i pilot testiranje. Konfigursanje ERP sistema zahteva da projektni tim razmotri određeni broj faktora (tabela 6.4).

Tabela 6.4. Faktore za razmatranje tokom konfigurisanja ERP sistema

| Faktori | Stvari koje treba razmotriti |
|---|---|
| Podaci | Ko je zadužen za integitet podataka? (centralizovana odgovornost nasuprot lokalne odgovornosti) |
| Raspodela procedura | Koji poslovni procesi treba da budu centralizovani?<br>Koji procesi treba da ostanu pod lokalnom kontrolom? |
| Transakcije | Da li ERP obezbeđuje praćenje transakcija?<br>Da li ERP obezbeđuje praćenje agregiranih transakcija? |
| Upravljanje podacima | Da li će ERP podržati centralizovano ili lokalno upravljanje podacima? |

## 6.2. ASAP metodologija implementacije SAP sistema

SAP (*Systems, Applications, Products*) AG je jedna od najpoznatijih softverskih kuća za proizvodnju ERP sistema, osnovana 1972. u Waldorfu u Nemačkoj, 1992. SAP je predstavio svoj sistem nazvan R/3 koji se bazira na klijent/server tehnologiji. Poslednjih 10-tak godina, SAP R/3 dominira ERP tržištem. R/3 softver omogućava integraciju svih poslovnih operacija kompanije u jedan sveobuhvatan sistem za planiranje, kontrolisanje i praćenje. Na raspolaganju je preko 1000 standardizovanih procesa koji predstavljaju rešenja najbolje poslovne prakse koja odražavaju iskustva, sugestije i zahteve vodećih kompanija. SAP R/3 sistem obezbeđuje integrisani skup poslovnih aplikacija koje pokrivaju kompletan opseg procesa koji se koriste gotovo u svim poslovanjima. R/3 aplikacije su moduli koji se mogu samostalno koristiti ili kombinovati sa drugim rešenjima.

R/3 sistem nije sektorski orjentisan, već podržava praktično sve tipove industrije, od diskretne proizvodnje preko procesne industrije, inženjeringa prema porudžbini do serijske proizvodnje. Primeri proizvodnje i prodaje različitih proizvoda, tj. grupa proizvoda i njihova klasifikacija na tipove proizvodnje, prikazani su u tabeli 6.5.

Tabela 6.5. R/3 proizvodi i odgovarajući tip proizvodnje

| Proizvod | Tip proizvodnje |
|---|---|
| Personalni računari | Serijska proizvodnja |
| Liftovi | Inženjering prema porudžbini |
| Motori | Proizvodnja prema porudžbini |
| Automobili | Serijska proizvodnja bazirana na prodaji |
| Pumpe | Proizvodnja prema porudžbini |
| Sijalice | Masovna proizvodnja |
| Farbe/lakovi | Procesna industrija, hemikalije |
| Lekovi | Procesna industrija, farmaceutika |
| Hrana/kozmetika | Maloprodaja |

Postoje dva pristupa uvođenja R/3 sistema:

1. **Analiza stanja i detaljno projektovanje R/3 softvera**. R/3 je izuzetno veliki, ima puno parametara koje treba postaviti na osnovu analize stanja. Da bi se na taj način radilo, zahteva se izuzetno poznavanje R/3 softvera od strane SAP konsultanata i dobra sistem analiza kod korisnika.

SAP obezbeđuje ASAP (*Accelerated*SAP) metodologiju i alate za brzo implementiranje R/3 rešenja. Ovaj način pruža više mogućnosti za kastomizaciju i prilagođavanje poslovnim procesima korisnika.

2. **Korišćenje predefinisanog SAP-ovog rešenja za određeni model proizvodnje**. Primeri kako je to urađeno za određene modele mogu se naći na IDES-u, sistemu za Internet demonstraciju i evaluaciju (*Internet Demonstration and Evaluation System*). IDES predstavlja jednu međunarodnu kompaniju koja se sastoji od nekoliko odelenja lociranih u različitim zemljama. Svrha IDES-a je da pokaže sveobuhvatne funkcije R/3 sistema. Međutim, fokus nije na samoj funkcionalnosti, već na poslovnim procesima i njihovoj integraciji. Takođe, IDES sadrži brojne poslovne scenarije koji se bave problemima upravljanja visokom inflacijom ili razumevanjem integracije elektronskog KANBAN sistema u MRPII okruženju, *just in time* proizvodnje i sl. IDES-om upravlja SAP koji redovno ažurira njegove podatke.

SAP R/3 sistem je razvijen tako da podrži distribuiranu višeslojnu klijent/server arhitekturu. Postoji jasna razlika između prezentacionog, aplikacionog, sloja baze podataka i Internet sloja. Prezentacioni sloj može da koristi brojne različite grafičko korisničke interfejse. SAP je razvio sopstveni softver korisničkog interfejsa SAPGUI koji pokriva 20 različitih jezika, ali takođe se mogu koristiti i *Microsoft Windows* ili *Internet browser* interfejsi. Aplikacioni serveri sadrže kompletnu logiku poslovnih procesa R/3 aplikacija. Ovi aplikacioni serveri se mogu aktivirati na Windows NT sistemima, UNIX operativnim sistemima i AS/400 sistemima. Sloj baze podataka upravlja komponentama aplikacije R/3 sistema i podacima preduzeća. Podržavaju IBM DB2, *Informix On-line*, *Microsoft SQL Server* i *Oracle* relacione sisteme za upravljanje bazama podataka. Internet sloj omogućava pristup R/3 sistemu putem Interneta ili intraneta.

R/3 sistemi nude i standardne interfejse koje omogućavaju integraciju R/3 sa aplikacijama razvijenih od strane drugih vendora. Ovi objektno-orijentisani interfejsi se nazivaju BAPI (*Business Application Programming Interfaces*) koji su kompatibilni sa *Microsoft*-ovim DCOM-om (*Distributed Component Object Model*) i OMG-ovom CORBA-om (*Common Object Request Broker Architecture*).

SAP je razvio sopstveni programski jezik nazvan ABAP (*Advanced Business Application Programming*) koji predstavlja jezik četvrte genera-

cije. R/3 takođe nudi kompletno razvojno okruženje pomoću koga developeri mogu da modifikuju postojeći SAP kôd ili da razvijaju sopstvene funkcije, prave izveštaje ili čak da razvijaju kompletne transakcione sisteme u okviru SAP *framework*-a.

Vizija SAP-a je da nastavi da obezbeđuje kompletna, integrisana rešenja. Svoju Internet strategiju, SAP je realizovao razvijanjem rešenja mySAP.com. Portali mySAP.com omogućavaju korisnicima svih softverskih rešenja da pristupaju svojim aplikacijama i R/3 bazama podataka putem Interneta i Web pretraživača sa svog *desktop* ili portabilnog uređaja. Rešenje mySAP.com je integrisan sa R/3 i predstavlja interfejs za sve SAP proizvode: kolaborativne, *front office* i *back office*. Na primer, prodavac na terenu može da popuni elektronsku formu porudžbine na svom *laptop* računaru, koja će zatim biti prevedena kao ulaz za R/3 sistem. Uvođenjem SAP rešenja u organizacije, osigurava se dobijanje međunarodnih sertifikata za kvalitet što otvara put na međunarodno tržište.

Projekat uvođenja SAP-a je izrazito veliki projekat, koji ne traje kratko, zahteva velika finansijska ulaganja i ima veliki uticaj na poslovanje organizacije koja ga uvodi. Teškoće pri implementaciji SAP-a proizilaze upravo iz činjenice da uspeh implementacije zavisi od tačnog mapiranja svih poslovnih procesa organizacije u SAP sistem. Ovo povlači za sobom tačnu konfiguraciju svih traženih procesa od samog početka projekta. SAP je vrlo fleksibilan, ali da bi se izvršila precizna konfiguracija nije dovoljno poznavati samo procese organizacije, već je potrebno dobro poznavati i SAP funkcionalnosti.

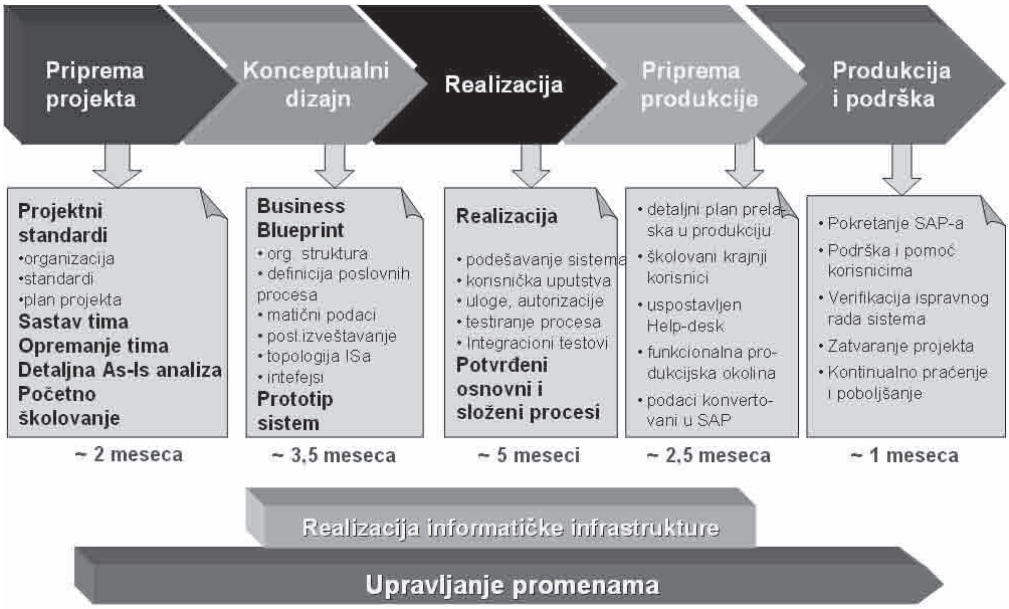
*Accelerated* ASAP (Ubrzani SAP) je metodologija brze implementacije koju je 1996. godine razvio SAP. ASAP nudi brojne alate i pomoć pri implementaciji procesa. Prednosti ove metodologije su brza implementacija, optimalna iskorišćenost resursa i budžeta i poboljšanje kvaliteta. Međutim, metodologija neće rešiti sve probleme, jer svaka organizacija ima posebne zahteve i na svaku SAP implementaciju će uticati organizaciona struktura.

Koraci ASAP metodologije su (Slika 6.1): priprema projekta, analiza zahteva, realizacija i priprema za produkciju, konačna priprema, puštanje u rad.

Slika 6.1. ASAP metodologija implementacije SAP rešenja

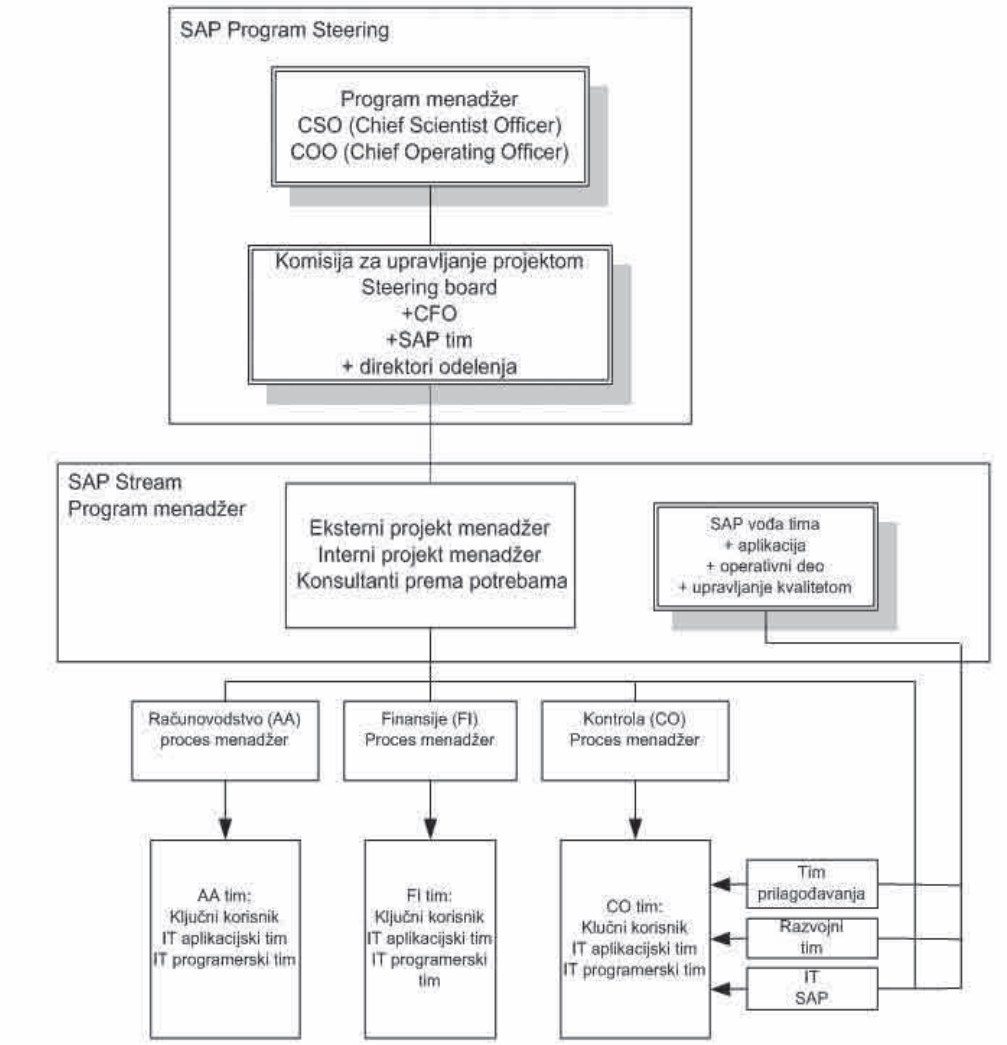
### Priprema projekta

Nakon pozitivne poslovne opravdanosti ulaganja u ERP sistem, pristupa se definisanju cilja, domena i organizacije projekta, zatim definiše se strategija implementacije i dužnosti ključnih korisnika. Na slici 6.2 se prikazuje plan prve faze projekta na primeru jedne organizacije koja se odlučila za uvođenje SAP rešenja, u okviru koje je faza pripreme trajala 170 dana.




Slika 6.2. Faze projekta

Primer projektne organizacije je prikazan na slici 6.3.




Slika 6.3. Primer projektne organizacije za implementaciju SAP modula [3]

### Analiza zahteva

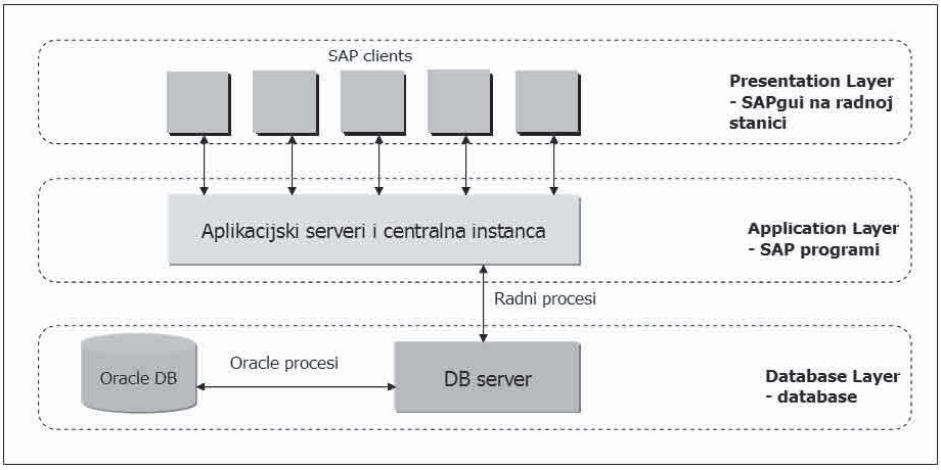
Svrha ove faze je kreiranje dokumenata specifikacije zahteva (tzv. *Business Blueprint*) čime se dokumentuju svi zahtevi poslovnih procesa organizacije. Tim dokumentom postiže se zajedničko razumevanje izvođenja poslova unutar SAP sistema. Primer terminskog plana faze specifikacije zahteva je prikazan na slici 6.4.

| Task Name | Durati | Start | Finish |
|---|---|---|---|
| **Conceptual design** | **16 dys** | **Fri 8.6.01** | **Fri 29.6.01** |
| Conceptual design ready for FI | 1 dy | Fri 15.6.01 | Fri 15.6.01 |
| Conceptual design ready for FI-AA | 1 dy | Fri 15.6.01 | Fri 15.6.01 |
| Conceptual design ready for CO | 2 dys | Fri 8.6.01 | Mon 11.6.01 |
| Complete process list | 1 dy | Fri 29.6.01 | Fri 29.6.01 |
| **Detailed design - Preparation of the prototyp** | **67 dys** | **Tue 12.6.01** | **Mon 10.9.01** |
| **Customizing** | **26 dys** | **Tue 12.6.01** | **Fri 13.7.01** |
| FI | 16 dys | Mon 25.6.01 | Thu 12.7.01 |
| FI-AA | 15 dys | Mon 18.6.01 | Fri 6.7.01 |
| CO | 15 dys | Tue 12.6.01 | Mon 2.7.01 |
| Closing of detailed design documents FI/AM/CO | 1 dy | Fri 13.7.01 | Fri 13.7.01 |

Slika 6.4. Terminski plan faze Specifikacije zahteva

U ovoj fazi zadatak menadžmenta projekta je identifikovanje nastalih promena između poslovnih procesa i organizacione strukture i analiza potencijalnih rizika koji su utvđeni u prvoj fazi. Zadatak menadžmenta promene procesa je upravljanje problemima promena kako bi se optimizovali procesi u organizaciji. Kreira se mapa poslovnih uticaja koja se koristi kao polazna tačka u upravljanju promenama procesa.

Sledeći korak je da se definiše i dokumentuje tehnički dizajn celokupnog sistema, a pomoć za ovaj posao traži se od partnera za hardver. Primer arhitekture SAP sistema u jednoj organizaciji je troslojna (Slika 6.5).




Slika 6.5. SAP R/3 arhitektura

U ovoj fazi se vrši i obuka korisnika kako bi korisnici razumeli SAP funkcionalnosti i postigli potrebno tehničko znanje čime je omogućen početak usklađivanja zahteva i SAP funkcionalnosti.

Dokument specifikacije poslovnih zahteva za poslove na primer finansijskog računovodstva bi obuhvatao sledeće definicije:

- Svrha i ocena dokumenta
- Povezanost sa drugim dokumentima (Proces, Dodela konta, Cena)
- Zaključak menadžmenta
- Konceptualni dizajn
- Organizaciona struktura
- Stavke
- Standardni procesi
- Izveštavanje
- Korisničke uloge i autorizacije
- Prilagođavanje
- Razvoj programa
- Test prihvatanja (*Business Acceptance Test*)
- Puštanje u rad (*Going live*)
- Otvorene teme (*Open issue*)
- Problemi, pretpostavke i rizici
- Potpisi

Dokument specifikacije zahteva potpisuju svi članovi tima i njime se potvrđuje dizajn procesa, organizaciona struktura, osnovni cilj, tehnički dizajn i sistemsko okruženje.

Na kraju druge faze sprovodi se provera kvaliteta i SAP preporučuje sledeću listu provere aktivnosti:

- Provera statusa projektnog tima i održavanje sastanaka upravljačkog odbora;
- Provera da li se uspešno održala obuka projektnog tima;
- Overa da su se održavale aktivnosti izgradnje tima (team building);
- Osiguranje uloga i odgovornosti korisnika;
- Potvrda razvoja tehničkog dizajna;

- Provera postavki okruženja razvojnog sistema;
- Overa da su tačno postavljene funkcije administriranja sistema;
- Provera inicijalne konfiguracije;
- Provera da je dokument specifikacije zahteva potpun i postpisan.

### Realizacija i priprema za produkciju

U fazi realizacije sprovodi se implementacija poslovnih i procesnih zahteva baziranih na dokumentu specifikacije zahteva i puštanja sistema u stvarni rad. U ovoj fazi radi se konfiguracija sistema koja se sastoji iz dva dela i to osnovne i konačne konfiguracije. Osnovna konfiguracija se odvija sekvencionalno kroz cikluse, sve dok se problemi ne reše i sve dok konfigurisani sistem nije spreman za konačni test integracije.

Sistem se konfiguriše i testira na osnovu liste glavnih poslovnih procesa (*Business Process Master List*). Nakon konfiguracije sistema sprovodi se test integracije koji predstavlja skup najvažnijih poslovnih scenarija koji osiguravaju da se celokupni sistem uključujući i eksterne komponente, potpuno integriše kako bi se zadovoljila specifikacija poslovnih zahteva. Glavni fokus je usmeren na testiranje SAP-a sa svim programima prenosa podataka iz drugih sistema na izveštaje, sve izlazne obrasce i prenose matičnih slogova od drugih klijenata.

U fazi realizacije postoje mnogi tehnički zadaci koji moraju da se obavljaju paralelno sa konfiguracijom, kao što su razvijanje poslovnog toka (*workflow*), izrada programa prenosa iz drugih sistema ili modula, kodiranje, poboljšanje, prilagođavanje izveštaja i obrazaca, kreiranje uloga ili autorizacije korisnika i arhiviranje.

Izlazni dokumenti iz ove faze su 4 glavne liste koje definišu poslovni scenario i SAP transakcije:

- Procedure poslovnih procesa koje opisuju SAP transakcije i koriste se za testiranje i dokumentaciju;
- Konfiguracija i plan testiranja koji određuju kako će se sprovoditi konfiguracija i kako će biti testirana;
- Razvijanje programa tako da se omoguće dodatni detalji eksternih zahteva za programiranjem;
- Kreiranje materijala za obuku krajnjih korisnika.

### Konačna priprema

U fazi konačne pripreme usklađuju se sve aktivnosti iz prethodnih faza, sprovodi se testiranje, obuka korisnika, upravljanje sistemom i rešavaju se bitna otvorena pitanja. U ovoj fazi SAP eksperti proveravaju konfiguraciju pojedinačnih komponenti sistema i daju korisne savete kako optimizovati sistem. Glavni cilj je proveriti da li je sistem spreman za rad u produkciji.

## Puštanje u rad

Puštanje sistema u rad i podrška sistemu podrazumeva prenos rada na produkciju. Program podrške je vrlo kritičan, a naročito u prvih nekoliko nedelja stvarnog rada. Pružanje usluga korisnicima (tzv. *service desk*), njegovo praćenje i podešavanje zavisi od povratnih informacija krajnjih korisnika. Odgovornost IT odelenja se svodi na implementaciju funkcionalnih zahteva poslovanja, pružanje neophodne podrške pri testiranju i funkcionisanje svih SAP sistema. Poboljšanje poslovnih procesa, merenjem ključnih indikatora performansi sistema (KPIs – *Key Performance Indicators*), nije završeno nakon što je sistem pušten u rad, već se ta aktivnost nastavlja.

Svi zahtevi nakon zatvaranja projekta se sprovode kroz menadžment promena koji se odvija u razvojnom sistemu, zatim se testira i nakon prihvatanja rezultata testiranja, prenosi na sistem produkcije.

Cena projekta implementacije SAP rešenja se sastoji iz 4 dela: licence, konsalting, hardver i održavanje softvera u iznosu od 17% na ukupnu vrednost licence. Prema do sada urađenim analizama 77% projekta implementacije SAP-a je završeno za manje od godinu dana., a ulaganje se u proseku vraća u periodu između 18 i 24 meseca.

## Rezime

1. Metodologije obezbeđuju jedan konzistentan i reproduktivni prilaz koji može biti primenjen na sve projekte, one smanjuju rizik od grešaka i izdaju kompletnu i konzistentnu dokumentaciju za trenutne i buduće projekte.
2. Modeliranje je opšti pristup u svim inženjerskim disciplinama i predstavlja projektovanje softverskih aplikacija pre njihovog kodiranja odnosno programiranja. U oblasti razvoja softvera, modeliranje se može izvoditi klasičnim (funkcionalnim) pristupom – definisanjem modela podataka i modela procesa ili objektno-orijentisanim pristupom korišćenjem UML-a.
3. Modeliranje procesa je tehnika koja organizuje i dokumentuje procese sistema i/ili implementira logiku, politike i procedure sistema. Neki od modela procesa sistemske analize su dijagram toka podataka (*Data Flow Diagram*, kraće DFD), IDEF0 metodologija i dr.
4. Model podataka (definisan i kao Model Objekti-Veze - MOV ili E-R *Entity-Relationship* model), preko skupa podataka i njihovih međusobnih veza, predstavlja stanje sistema u jednom trenutku vremena. Postoji nekoliko notacija modela objekti-veze. Mnogi su imenovani prema njihovim tvorcima (npr., Chen, Martin, Bachman, Merise) ili prema objavljenim standardima (npr., IDEF1X).
5. Objektno-orijentisana analiza zahteva da se identifikuju objekti, njihovi atributi, ponašanja i relacije koji podržavaju zadate poslovne zahteve sistema. Faze obavljanja objektno-orijentisane analize su: modelovanje funkcija sistema slučajevima korišćenja, modelovanje aktivnosti slučajeva korišćenja upotrebom dijagrama aktivnosti i pronalaženje i identifikacija poslovnih objekata i njihovih relacija.
6. Jedna od reprezentativnijih metodologija razvoja poslovnih rešenja je *Microsoft Solution Framework* (MSF) procesni model koji opisuje generalne sekvence aktivnosti za izgradnju i uvođenje poslovnih rešenja, kombinovanjem najboljih principa modela vodopada i spiralnog modela. MSF obuhvata discipline za upravljanje timovima, procesima, tehnologijom i rizicima, tj. elementima sa kojima se projekti najčešće susreću.

7. Proces implementacije gotovih ERP rešenja se razlikuje od tradicionalnog procesa razvoja sistema. Proces razvoja ERP sistema obuhvata planiranje, analizu zahteva, projektovanje, detaljan dizajn, implementaciju i održavanje. Planiranje započinje sa procenom potreba koje predstavljaju poslovnu opravdanost kupovine softvera. Procena potreba je veoma bitna usled investicije u ERP sistem i njegovog uticaja na celokupno poslovanje. Faza analize zahteva jednog ERP projekta obuhvata određivanje poslovnih procesa koji će biti podržani ERP paketom. U fazi projektovanja se uvode najbolje prakse koje podržava ERP sistem, što povlači reinženjering poslovnih procesa. Ovde se uočava osnovna razlika u odnosu na tradicionalni pristup razvoja sistema, gde projektanti definišu nove poslovne zahteve i implementiraju softver prema tim zahtevima. Jedna od osnovnih odluka u implementaciji ERP paketa je da li da se redizajniraju organizacioni poslovni procesi kako bi odgovarali softveru ili kastomizirati softver prema organizacionom poslovnim praksama.
8. *Accelerated* ASAP (Ubrzani SAP) je metodologija brze implementacije koju je razvila kompanija SAP AG. ASAP nudi brojne alate i pomoć pri implementaciji procesa. Prednosti ove metodologije su brza implementacija, optimalna iskorišćenost resursa i budžeta i poboljšanje kvaliteta. Koraci implementacije su priprema projekta, analiza zahteva, realizacija i priprema za produkciju, konačna priprema i puštanje u rad.

## Pitanja za razgovor i vežbe

1. Zašto su važne metodologije kod razvoja informacionih sistema?
2. Koja je bitna razlika između strukturne i objektno-orijentisane analize?
3. Definišite termin polimorfizam i navedite primer.
4. Kako implementacija ERP rešenja doprinosi reinženjeringu poslovnih procesa?
5. Čemu služi matrica kompromisa projekta (*project tradeoff matrix*) koju predlaže MSF okvir?
6. U okviru dokumenta procene rizika kako se izračunava mera izloženosti riziku?
7. Koje tehnike za ocenu ekonomske izvodljivosti poznajete i ukratko ih obrazložite?

8. Koja rola prema MSF okviru kreira funkcionalnu specifikaciju poslovnog rešenja?
9. Koji dijagrami se iscrtavaju kod fizičkog dizajna poslovnog rešenja?
10. Šta je to arhitektura aplikacije i kakvu ulogu imaju kod razvoja informacionih sistema?
11. Koji su glavni zadaci faze stabilizacije rešenja?
12. Kako se tradicionalni sistemi životnog ciklusa razvoja sistema razlikuju od projektovanja i implementacije ERP sistema?

## Reference


[1] Ambler, S.W., Introduction to the Diagrams of UML 2.0, http://www.agilemodeling.com/essays/umlDiagrams.htm

[2] Atanasijević, V., „Bug Management – best practice", Sinergija, 2003.

[3] Čupić, A., „Strategija uspješnog uvođenja sustava za planiranje resursa poduzeća", magistarski rad, Ekonomski fakultet Zagreb, Sveučilište u Zagrebu, 2005.

[4] Eckel, B., *Misliti na Javi*, CET, Beograd, 2002.

[5] Hoffer, J.S., M.B. Prescott, F.R. McFadden, *Modern Database Management, sedmo izdanje*, Prentice Hall, New Jersey, 2005.

[6] IBM, „DB2 Family Fundamentals", Student Notebook, International Business Machines Corporation, 2005.

[7] IEEE, IEEE Standard Conceptual Modelling Language – Syntax and Semantics for IDEF1X, eeexplore.ieee.org/iel5/6170/16492/00761916.pdf.

[8] Lazarević, B., Z. Marjanović, N. Aničić, S. Babarogić. *Baze podataka*, Fakultet organizacionih nauka, Beogradski univerzitet, Beograd, 2003.

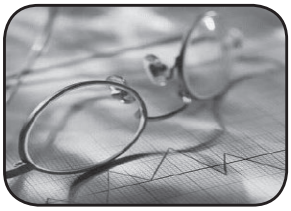
[9] Microsoft, „Analyzing Requirements and Defining Microsoft. NET Solution Architectures", Workbook, Microsoft Corporation, 2003.

[10] Njeguš, A., A. Veljović, *Internet poslovno programiranje*, Megatrend univerzitet, Beograd, 2004.

[11] Schildt, H., *Java 2: Kompletan priručnik*, četvrto izdanje, Mikro knjiga, Beograd, 2001.

[12] UML standardi (www.rational.com i www.omg.com).

[13] Veljović., A. i A. Njeguš, *Osnove relacionih i analitičkih baza podataka*, Megatrend univerzitet, Beograd, 2004.



# DEO III: ANALITIČKI POSLOVNI INFORMACIONI SISTEMI

## Ciljevi poglavlja

Današnji poslovni informacioni sistemi ne mogu da se zamisle bez ugrađene poslovne inteligencije. Vodeći IT analitičari predviđaju da će se u narednih pet godina softverska industrija fokusirati na poslovnu inteligenciju (*Business Intelligence* – BI) i analitiku, usled potrebe organizacija za donošenjem odluka na svim nivoima menadžmenta. Operaciona BI rešenja uvode inteligenciju u sve poslovne procese organizacije pružajući pravu informaciju, pravim ljudima u pravo vreme. Današnji izazov za organizacije nije samo pružanje informacija ljudima koji vode posao, već omogućavanje svim entitetima u preduzeću da pristupaju i sarađuju nad istim informacijama, kako i kada se za to javi potreba.

U ovom delu se proučavaju delovi poslovne inteligencije i to *Data warehouse*, OLAP sistemi i *data mining* i na kraju se navode koraci razvoja jednog BI rešenja.

Nakon proučavanja ovog poglavlja znaćete da:

- objasnite razliku između OLTP i OLAP alata;
- objasnite pojmove: *data mart*, *data warehouse*, OLAP i *data mining*;
- napravite dimenzioni model putem šeme zvezda;
- sagledate funkcionalnosti OLAP sistema i kreirate kocku nad željenim podacima;
- objasnite tipove arhitekture OLAP sistema;
- objasnite tehnike data mining-a;
- objasnite korake razvoja jednog BI rešenja.

# 7. Skladište podataka

Kompanija svakodnevno prikuplja velike količine podataka. Ti podaci su često sirove činjenice koje odražavaju tekuće stanje poslovanja. Prevođenje tih sirovih podataka u dragocene informacije je ključni analitički proces, na osnovu kojih kompanija donosi poslovne i operativne odluke. Naredni scenario prikazuje razliku između sirovih i izvedenih podataka.

**Scenario:**

Sirovi podaci: Finansijska institucija prikuplja podatke o svim računima i ušteđevinama klijenata. Sirov podatak na primer, može pokazati da je Sefan M. podigao 50 evra jutros, sa svog računa u Amsterdamu.

Izvedene informacije: Stefan živi u Beogradu, ali u proteklih pet meseci, Stefan je podizao novac u Londonu, Oslo-u, Stockolm-u, što dovodi do zaključka da on često putuje po Evropi. S toga bi on možda bio zainteresovan za specijalnu kreditnu karticu koji mu omogućava neograničen pristup svom računu u 16 različitih zemalja uz odgovarajuću godišnju članarinu. Pitanja koja se postavljaju nakon ove analize su: Koji je prosečan dnevni bilans njegovog računa?, Za koje usluge bi bio zainteresovan?

Nakon što su se uvidele vrednosti poslovne analize, zahtevi za podacima i informacijama postaju sve brojnije i učestalije. Izlaženje u susret ovim zahtevima može biti veoma kompleksan zadatak, jer je neophodno pretraživati ogromne količine prikupljenih izvora podataka, konsolidovati ih, analizirati i distribuirati informacije drugim članovima organizacije.

Da bi izašla u susret ovim zahtevima, kompanija obično implementira sisteme za podršku odlučivanju kako bi obezbedila podatke i informacije koji se mogu koristiti za obavljanje poslovne analize. Investicija u ove sisteme je obično veoma velika, naročito ako se uzme u obzir vreme implementacije, troškovi i napor. Povratak investicija (ROI) odslikava koliko dobro sistemi za podršku odlučivanju mogu da zadovolje potrebe poslovanja.

Kompanija može imati jedan ili više operacionih sistema koji obavljaju osnovne poslovne procese. Ovi sistemi se mogu nalaziti na odvojenim

serverima i čak različitim mrežama. Te operacione sisteme nazivamo **OLTP** (*On-line transaction processing*) koji prikupljaju transakcije poslovanja i snabdevaju skladište podataka (*data warehouse*) sa podacima.

Karakteristike OLTP sistema su:

- obrađuju poslovne transakcije u realnom vremenu;
- sadrže strukture podataka optimizovane za unošenje i izmenu podataka;
- obezbeđuju ograničene sposobnosti za podršku odlučivanju (izveštaji se sastoje od tekućih podataka).

Primeri OLTP sistema su: aplikacije praćenja porudžbina, aplikacije pružanja usluga klijentima (npr., otvaranje računa), bankarske funkcije (npr, krediti) i dr.

Sistem skladišta podataka (*Data warehouse* – DW) sadrži komponente koje pomeraju podatke sa izvornih sistema ka korisnicima koji žele da vrše analize podataka. Primarna funkcija sistema skladišta podataka je da pruži podršku organizaciji u obavljanju poslovne analize. Koncept skladištenja (*warehousing*) podrazumeva skladištenje agregiranih, ekstrahovanih i filtriranih podataka u meta baze, koje omogućavaju slojevit, multidimenzionalni pristup podacima, kakav je potreban za donošenje odluka najvišeg strateškog nivoa.

**Skladištenje podataka (*Data Warehousing*) je proces integracije podataka u jedan repozitorijum iz kojeg krajnji korisnici mogu sprovoditi *ad-hock* analize podataka i praviti izveštaje.** Skladište podataka je analitička baza podataka namenjena samo za čitanje i koristi se kao osnova sistema za podršku odlučivanju.

Skladište podataka obezbeđuje podatke za procese poslovne analize, integriše podatke sa heterogenih sistema i skladišti podatke u fizičkim strukturama koje su optimizovane za analizu podataka. Skladište podataka je informaciona baza podataka dizajnirana za podršku jedne ili više klasa analitičkih zadataka, kao što su nadgledanje i izveštavanje, analiza i dijagnoza, simulacija i planiranje.

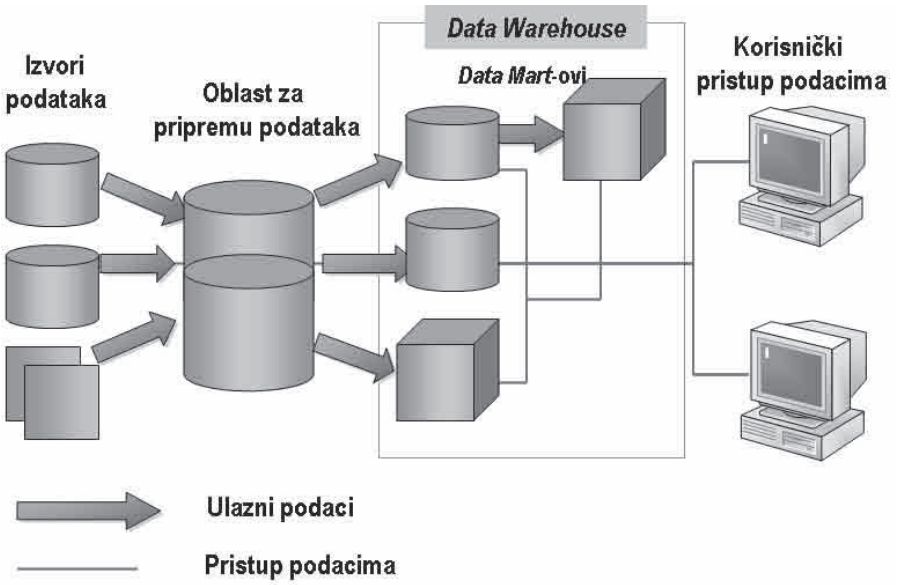
Karakteristike skladišta podataka su:

- **Organizacija -** podaci su organizovani po predmetu i sadrže relevantne informacije za podršku odlučivanju.

- **Konzistentnost -** podaci u različitim operacionim bazama podataka se drugačije šifriraju. U skladištu će ti podaci biti šifrovani na konzistentan način.
- **Vremenski podaci -** podaci se čuvaju mnogo godina kako bi se iskoristili za praćenje trendova, prognoza i vremensko poređenje.
- **Multidimenzionalnost -** obično skladište podataka koristi multidimenzionalnu strukturu.
- **Web-zasnovanost –** u današnje vreme **s**kladište podataka je dizajnirano tako da obezbedi jedno efikasno okruženje za Web aplikacije.

Sistem skladišta podataka sadrži mnoge komponente koje prenose podatke sa izvornih sistema do korisnika koji izvršavaju analizu podataka. Komponente sistema skladišta podataka su (Slika 7.1):

- **Izvori podataka** – Izvorni sistemi su operacioni sistemi, npr. OLTP sistemi koji mogu biti relacioni.
- **Oblast za pripremu podataka** – podrazumeva skup procesa koji čisti, transformiše, povezuje i priprema izvorne podatke za korišćenje u DW. Podaci se transformišu u konzistente formate. Oblast za pripremu podataka se nalazi na jednom ili nekoliko računara, ne mora da bude zasnovana na relacionoj tehnologiji i ne podržava korisničke izveštaje.
- ***Data Mart*** – je podskup DW koji sadrži podatke specifične za određenu poslovnu oblast kao što su finansije ili analiza klijenata. *Data mart*-ovi mogu biti uključeni u DW, mogu se izgraditi u relacionim ili OLAP bazama podataka i mogu da sadrže detaljne ili sumarne podatke koje se mogu ili ne, deliti kroz *data mart*-ove.
- **Data Warehouse** – može se definisati i kao virtuelna unija *data mart*-ova sa integrisanim informacijama koje su deljive kroz *data mart*-ove ili kao centralizovano, integrisano skladište podataka koje obezbeđuje podatke *data mart*-ovima.




Slika 7.1. Komponente DW sistema

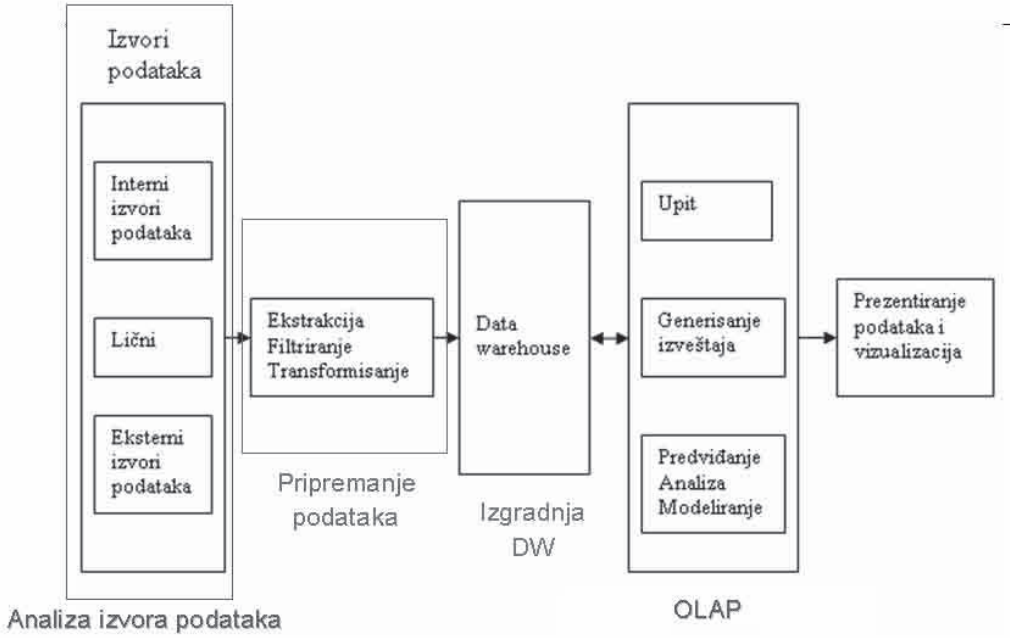
## 7.1. Razvoj skladišta podataka

Za razliku od transakcionih sistema (OLTP), koji su orjentisani poslovnim procesima, skladišta podataka su subjektno orjentisana, što znači da su fokusirana na subjekte u poslovnim procesima, kao što su kupci, zaposleni i dobavljači. Integrisanost podataka u skladištima podataka obezbeđuje da se podaci predstavljaju u konzistentnim formatima korišćenjem konvencija pri zadavanju imena i ograničenja nad domenima, atributima i merama. Podaci u skladištima podataka su vremenski zavisni, što znači da je svaki podatak koji se nalazi u skladištu podataka u vezi sa nekim vremenskim trenutkom. Na kraju, podaci u skladištima podataka su nepromenljivi, tj. čim se neki podatak upiše u skladište podataka, moguće mu je samo pristupati. Na slici 7.2 su prikazani svi elementi potrebni za razvoj skladišta podataka.

Pri izgradnji skladišta podataka najbitniji su sami podaci, a ne poslovni procesi i funkcije, kao što je to slučaj sa transakcionim sistemima. Baze podataka namenjene sistemima za podršku odlučivanju mogu biti veoma velike (terabajtne), pri čemu neke tabele mogu sadržati i gigabajt podataka. Zato se veličina baze podataka mora uzeti u obzir pri planiranju skladišta podataka.

Za razvoj skladišta podataka potrebno je:

1. izvršiti analizu izvora podataka
   a. prikupljanje zahteva;
   b. planiranje skladišta podataka;
   c. izbor tehnike analize podataka.
2. pripremiti podatake
   a. ekstrakcija i čišćenje podataka,
   b. transformacija podataka.
3. izgraditi skladište podataka.
   a. denormalizacija podataka,
   b. definisanje hijerarhija,
   c. kreiranje agregacija,
   d. kreiranje fizičkog modela,
   e. generisanje baze podataka,
   f. učitavanje podataka.

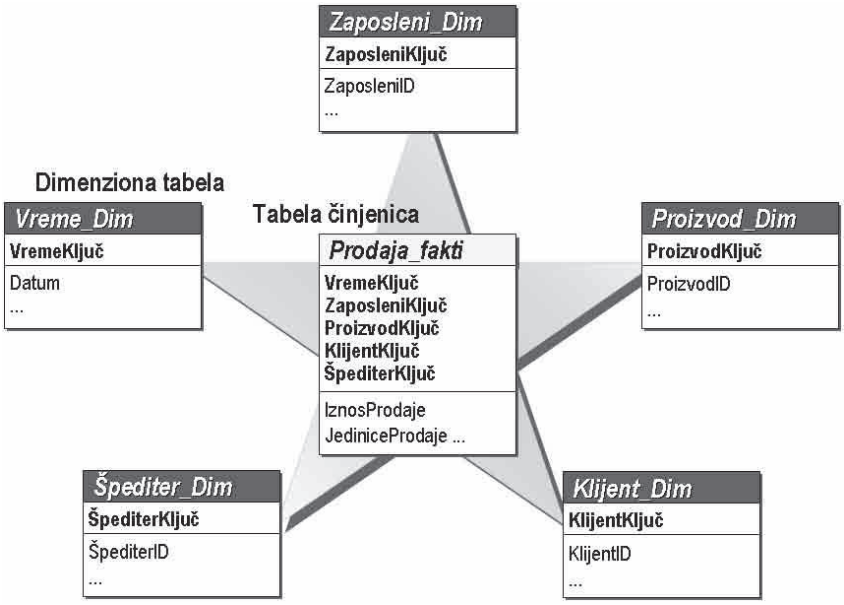



Slika 7.2. DW i OLAP

### Dimenziono modeliranje

Dimenziono modeliranje je tehnika logičkog dizajna čiji je cilj prezentacija podataka u obliku koji obezbeđuje visoke performanse sistema radi vršenja analize podataka. U dimenzionom modeliranju, strukture podataka su tako organizovane da opisuju mere i dimenzije. **Mere** su numerički podaci smešteni u centralnoj, takozvanoj tabeli činjenica (fakt tabela). Mere predstavljaju analizirane vrednosti, kao što su jedinica prodaje ili broj zaposlenih. **Dimenzije** su standardni poslovni parametri koji definišu svaku transakciju.

Osnovu za izradu dimenzionog modela predstavljaju meta podaci, na osnovu kojih se vrši definisanje hijerarhija, elemenata i atributa, normalizacija i denormalizacija i definisanje agregacija. Dimenzione tabele sadrže opisne tekstualne informacije. Svaka dimenziona tabela ima svoj primarni ključ koji predstavlja kolonu ili grupu kolona u tabeli čiji sadržaj jedinstveno identifikuje zapise, a svi oni učestvuju u stvaranju primarnog ključa tabele činjenica. **Tabele činjenica** sadrže podatke koji su, najčešće, numeričkog tipa i mogu sadržati veliki broj zapisa. Fizička arhitektura dimenzionog modela je šema zvezde. Na slici 7.3 se prikazuje šema zvezde kroz primer.




Slika 7.3. Šema zvezde

Pri dizajniranju škladišta podataka najčešće se koristi šema zvezde. Ona se sastoji od relativno malog broja tabela sa dobro definisanim vezama. Šema zvezde polako postaje standard za izradu skladišta podataka zbog svojih prednosti u odnosu na ostale relacione strukture:

- obezbeđuje kraće vreme odziva na upit, jer se smanjuje broj fizičkih veza između tabela;
- model je jednostavan i lako se mogu vršiti modifikacije;
- pojednostavljuje razumevanje i navigaciju meta podataka;
- održavanje je relativno jednostavno;
- proširuje skup alata koji se mogu koristiti za rad sa podacima.

Osnovna karakteristika šeme zvezde jeste da su dimenzione tabele denormalizovane. Denormalizacija je pristup gde se podaci u bazi podataka ponavljaju zbog pojednostavljenja dizajna i karakteristika. Ovim postupkom se smanjuje broj potrebnih veza koje se moraju obraditi zadavanjem upita. Time se direktno utiče na poboljšanje performansi sistema, jer što je manji broj veza, to sistem brže nalazi tražene podatke. Primer normalizovane i denormalizovane prezentacije podataka dat je na sledećoj slici.

| ProizvodID | NazivProizvoda | Karakteristike | DobavljačID |
|---|---|---|---|
| 101 | N1 | K1 | XYZ |
| 102 | N2 | K2 | XYZ |
| 103 | N3 | K3 | ABC |

| DobavljačID | NazivDob |
|---|---|
| XYZ | Sony |
| ABC | Panasonic |

a) normalizovana prezentacija

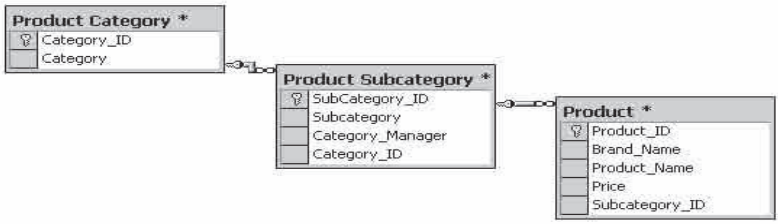
| ProizvodID | NazivProizvoda | Karakteristike | DobavljačID | NazivDob |
|---|---|---|---|---|
| 101 | N1 | K1 | XYZ | Sony |
| 102 | N2 | K2 | XYZ | Sony |
| 103 | N3 | K3 | ABC | Panasonic |

b) denormalizovana prezentacija

Slika 7.4. Različiti načini prezentacije podataka

Dimenzije veoma često mogu biti organizovane u hijerarhiji. Svaki hijerarhijski nivo se nastavlja sa nekim drugim hijerarhijskim nivoom. Na

primer, unutar vremenske dimenzije, dani se nastavljaju na nedelje, koji se nastavljaju na kvartale itd. Za dimenziju proizvoda vezuje se proizvodna grupa, koja se nastavlja na proizvodne vrste.




Slika 7.5. Primer hijerarhije proizvoda

Dimenzioni elementi su specijalna kategorija podataka, koja predstavlja određeni nivo u dimenzionoj hijerarhiji. Za svaki hijerarhijski nivo postoji po jedan dimenzioni element. Na primer, kod dimenzije proizvod, mogu postojati tri dimenziona elementa: prozvod, grupa i vrsta proizvoda. U ovom modelu možemo reći da dimenzioni element "proizvod" predstavlja najniži hijerarhijski nivo u dimenziji proizvod, dok Kategorija proizvoda predstavlja najviši nivo (Slika 7.5).

Posmatranje podataka iz različitih, ali blisko povezanih perspektiva omogućava da korisnik analizira podatke na različitim nivoima detalja. Postupak prelaska sa nivoa sa manjim brojem detalja na nivo sa većim brojem detalja naziva se spuštanje u dubinu (*drill down*) i predstavlja zahtev korisnika da mu se prikaže više detalja. Postupak prelaska sa nivoa sa većim brojem detalja na nivo sa manjim brojem detalja, na tzv. sumarne podatke, naziva se dizanje naviše (*drill up*). Na primer, upit bi mogao prezentovati prodaju u odnosu na neke regione. Pošto pronađemo vrh prodaje u nekom regionu, spuštamo se naniže da bi smo saznali kako se prodaja odvija po opštinama. Na primer, geografski podaci vezani za prodaju mogli bi se organizovati u sledeću hijerarhiju:

SVET → KONTINENT → DRŽAVA → OBLAST → GRAD → OPŠTINA

Pored operacija *drill down* i *drill up*, postoji i operacija *drill across*, koja se koristi za povezivanje dve ili više činjeničnih tabela na istom nivou hijerarhije.

Dobro dizajnirana šema zvezde mora obezbediti postojanje različitih nivoa detalja, tj. hijerarhija.

Bitan element kod dimenzionog modeliranja je i kreiranje agregiranih atributa. Agregacija je proces skupljanja činjeničnih podataka po unapred definisanim atributima. Na primer, tipični korisnik će često postaviti zahtev da mu se prikaže ukupna prodaja za ceo mesec. Ovaj zahtev bi se u bazi podataka interpretirao kao potreba da se saberu svi podaci o dnevnoj prodaji u toku određenog meseca. Ako bi, na primer, tokom jednog dana postojalo 1000 transakcija u svakoj od 1000 prodavnica, onda bi ovaj upit morao da obradi 30 miliona redova da bi se dobio odgovor. Ovakav upit bi znatno trošio sve raspoložive resurse. Za podatke kojima se češće pristupa poželjno je izvršiti sumiranje. Time se omogućava da se već postojeći sumarni podaci mogu odmah koristiti, čime se znatno smanjuje vreme odziva na upit koji treba da obradi te sumarne podatke. Na primer, ako bi postojala tabela u kojoj bi se čuvali sumarni podaci o prodaji za svaku od 1000 prodavnica, onda bi upit o ukupnoj prodaji za ceo mesec morao da obradi 1000 redova. Prema tome, postojanje tabele sa sumarnim podacima, u ovom primeru, smanjuje potrebu obrade 30 miliona redova podataka.

Glavni razlozi kreiranja agregacija su da se poboljšaju performanse upita, tj. da se smanji vreme odziva na upit, kao i da se smanji broj resursa potrebnih za izvršenje upita. Jedini problem je što se upotrebom agregacija povećava sama baza podataka.

## 7.2. Primena skladišta podataka

Primena skladišta podataka je širokog dijapazona, počev od državnih organa, zdravstva i obrazovanja, pa do finansija, prodaje, marketinga, nabavke i proizvodnje. Primena skladišta podataka u finansijama vezana je za, npr.:

- izveštaj o protoku novca po grupama proizvoda i organizacionim jedinicama koji omogućava analizu prihoda i troškova po velikom broju parametara;
- detaljnu analizu profita, multidimenzione izveštaje po kategorijama profitnih i troškovnih centara;
- izradu bilansnih računa;
- izradu sumarnih izveštaja o ključnim finansijskim parametrima.

Primena skladišta podataka u marketingu vezana je za izradu strateških marketinških analiza koje se implementiraju putem multidimenzionih izveštaja, sa detaljima o prodaji viskoprofitnih proizvoda i praćenjem po vremenu, regionima, distributerima, sektorima potrošnje i dr. Mogućnost slobodnog izvlačenja sumarnih izveštaja i zalaženja u detalje u područjima koja ne ispunjavaju očekivanja, daje sektoru marketinga uvid u promene na tržištu. Kako su marketinške kampanje skupe, arhiviranje podataka o tržištu daje odgovor na pitanje koji je segment tržišta reagovao, preko kojih kanala, na kom geografskom području i preko kojih medija. Ovi podaci su od neprocenjivog značaja za organizovanje narednih kampanja.

Primena skladišta podataka u prodaji vezana je za analizu prodaje i davanje odgovora na pitanja kao što su:

- Koji proizvod donosi najveći profit?
- Koji kupci kupuju najprofitabilnije proizvode? i sl.

Primena multidimenzionih izveštaja dovodi do uočavanja Pareto principa[1], pokazujući odnos između proizvoda, profatibilnosti i geografske distribucije (npr. 20% proizvoda donose 80% profita). Precizna analiza može biti osnova za stimulaciju prodavaca (stimulacija po ostvarenom profitu, a ne po prihodu) i za planiranje prodaje.

Primena skladišta podataka u nabavci veoma je važna jer nabavka određuje profitabilnost preduzeća na više načina. Pre svega, to su troškovi materijala, praćenje odnosa sa dobavljačima kroz vreme. Još su veće mogućnosti upravljanja zalihama, u smislu oslobađanja obrtnog kapitala zarobljenog u zalihama. Uz dobro planiranje omogućava se dobro analitičko praćenje performansi dobavljača, tačno ispunjenje dogovorenih rokova, kvaliteta i drugih parametara.

Primena skladišta podataka u proizvodnji vezana je za davanje odgovora na pitanja kao što su:

- Koliko vremena treba da se napravi proizvod A?
- Gde je usko grlo proizvodne linije?
- Gde se javlja najviše problema vezanih za kvalitet?

1 Paretov princip, poznatiji kao pravilo 80/20, je tvrdnja da u svakom društvu posledice proizilaze iz malobrojnih uzroka. 80% posledica je izazvalo 20% uzroka. 20% klijenata donosi 80% profita. 80% procesorskog vremena se troši na 20% aplikacijskog koda i druga tumačenja.

Analiza upravljanja kapacitetima rešava probleme u operativnom planiranju i otvara mogućnost za "provlačenje" proizvodnje kroz ograničene kapacitete bez većih investicija. I ovde se može uključiti Pareto analiza, tj. 80% problema potiče od 20% uzroka, što je posebno važno za praćenje kvaliteta proizvoda. Da bi se problem kvaliteta mogao dobro analizirati i u transakcionim sistemima i u skladištu podataka, potrebno je obezbediti potpuno praćenje proizvodnog procesa, od ulaznih sirovina do gotovog proizvoda.

Primena skladišta podataka u upravljanju kadrovima vezana je za analizu i planiranje razvoja kadrova i predstavlja prvi korak u uvođenju upravljanja znanjem kao novog kvaliteta i prednosti organizacije na tržišnoj utakmici. Skladišta podataka se mogu primeniti i za praćenje i analizu zarada i troškova (po grupama, organizacionim jedinicama, regionima), efikasno ulaganje u obrazovanje kadrova i dr.

# 8. OLAP sistemi

OLAP (*On-line Analytical Processing*) rešenja omogućavaju korisnicima brz i fleksibilan pristup podacima i predstavljaju nadgradnju skladišta podataka. Interaktivno analitičko procesiranje (OLAP) namenjeno je *on-line* analizama i izveštavanjima, za razliku od produkcionih sistema namenjenih ažuriranju baza podataka i obradi transakcija (*On-line Transaction Processing* – OLTP).

Jedna od karakteristika koja razdvaja transakcione sisteme (OLTP) od analitičkih (OLAP) jeste dizajn baze podataka:

- Transakcioni sistemi su dizajnirani tako da preuzimaju podatke, vrše izmene nad postojećim podacima, daju izveštaje, održavaju integritet podataka i upravljaju transakcijama što je brže moguće.
- Analitički sistemi nisu predviđeni da obavljaju ove poslove. Oni se dizajniraju za veliki broj podataka namenjenih samo za čitanje, obezbeđujući informacije koje se koriste za donošenje odluka.

Postavlja se pitanje, šta je to što je krajnjem korisniku potrebno. Krajnji korisnik zahteva sledeće:

- da može da postavi bilo koje poslovno pitanje;
- da bilo koji podatak iz preduzeća koristi za analizu;
- da dobijeni podaci budu integrisani i pouzdani;
- mogućnost neograničenog izveštavanja i dr.

U početku su upiti korisnika bili relativno jednostavni. Međutim, vremenom su korisnički upiti postali toliko složeni da relacioni alati (OLTP alati) nisu bili u mogućnosti da daju odgovore u prihvatljivom vremenskom periodu. Upravo u tu svrhu se koriste OLAP sistemi. Oni omogućavaju jednostavnu sintezu, analizu i konsolidaciju podataka. Koriste se za intuitivnu, brzu i fleksibilnu manipulaciju transakcionim podacima. OLAP sistemi podržavaju kompleksne analize koje sprovode analitičari i omogućavaju analizu podataka iz različitih perspektiva (poslovnih dimenzija). OLAP je način obrade podataka koji karakterišu ad-hoc upiti, slabo struktuirani izveštaji i analiza koja obuhvata relativno mali broj transakcija, ali koja uključuje veliki broj tabela i zapisa u njima.

Osnovni elementi OLAP sistema su:

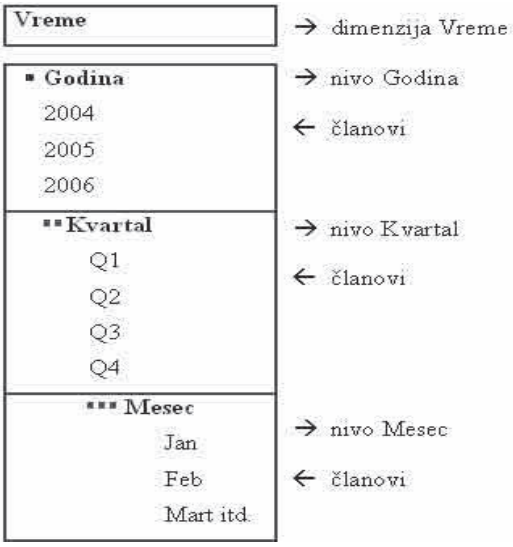
- baza podataka, koja služi kao osnova za analizu,
- OLAP server, za upravljanje i manipulaciju podacima,
- interfejs sistem, prema korisniku i prema drugim aplikacijama,
- alati za administriranje.

**OLAP baza podataka** je definisana sledećim komponentama:

- **Numeričke mere** – mere su vrednosti podataka ili činjenice koje korisnici analiziraju. Primeri mera su iznos prodaje, troškovi prodate robe itd.
- **Dimenzije** – dimenzije predstavljaju poslovne kategorije koje obezbeđuju kontekst numeričkim merama. Dimenzijama OLAP-a je lakše navigirati nego dimenzijama šeme zvezde.
- **Kocke** – Kocke kombinuju sve dimenzije i sve mere u jedan konceptualni model.

**OLAP dimenzije** sadrže sledeće hijerarhijske elemente (Slika 8.1):

- **Dimenzije** – organizovani nivoi i članovi u strukturi drveta.
- **Nivoi** – grupa članova dimenzije koji imaju isto značenje.
- **Članovi** – Svaka diskretna vrednost u dimenziji.

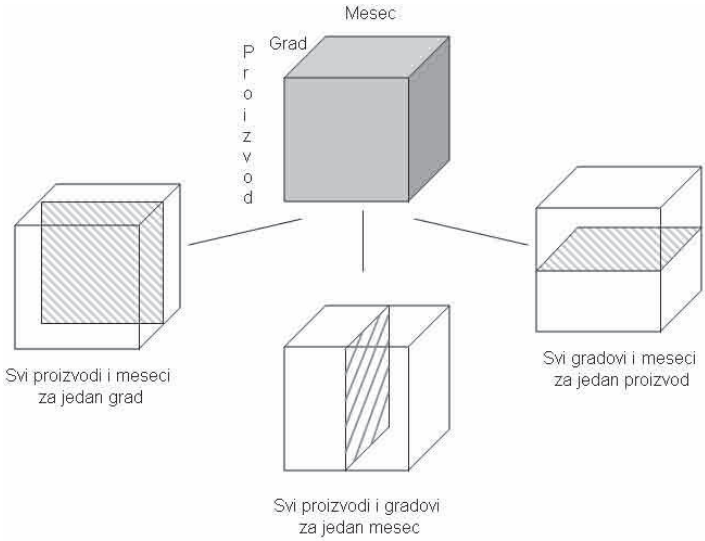



Slika 8.1. Elementi dimenzije

**Kocka** je logička struktura skladištenja OLAP baze podataka. Kocka kombinuje dimenzije i mere kako bi korisnici mogli da prave upite. Kocka definiše skup povezanih dimenzija koje formiraju jednu n-dimenzionalnu mrežu:

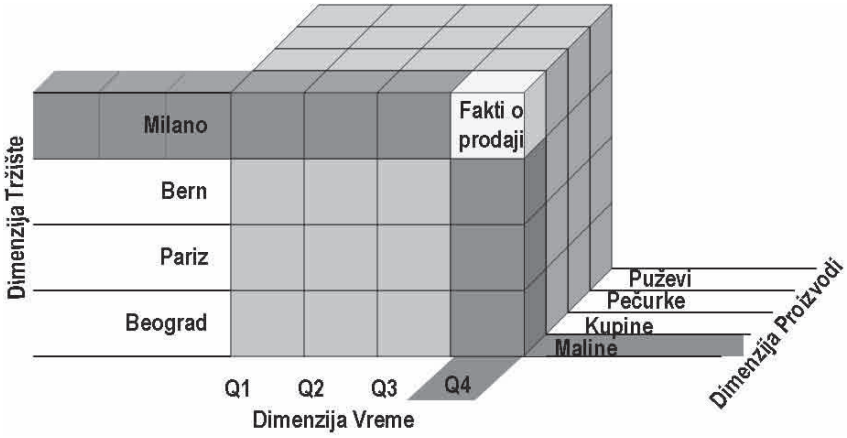
- Svaka ćelija kocke sadrži jednu vrednost;
- Vrednost svake ćelije je presek dimenzije.

Mere su numeričke vrednosti koje korisnici analiziraju. Svaka kocka mora da sadrži barem jednu meru, ali ne može da ima više od 1024 mera. Iako su podaci sačuvani samo jednom i to na jednom mestu, svaki korisnik može dobiti različite poglede na jedne iste podatke. Jedan od primera dat je na sledećoj slici.




Slika 8.2. Različiti pogledi na iste podatke

Kao što smo ranije rekli, dimenzija predstavlja kategoriju podataka, kao što su tip proizvoda, region, vreme itd. Svaka ćelija kocke sadrži agregirane podatke koji su u vezi sa dimenzijama. Na primer, jedna ćelija može sadržati podatke o ukupnoj prodaji za dati proizvod i region u toku jednog meseca. Na slici 8.3 se može videti da kocka skladišti vrednosti prodaje za svaki proizvod, svako tržište i za svaki period vremena. Da bi dobili ukupnu godišnju vrednost, korisnici biraju proizvod i tržište i sumiraju ćelije iz sva četiri kvartala.




Slika 8.3. OLAP kocka „Prodaja“

Kocka “Prodaja” (Slika 8.3) sadrži tri dimenzije: Vreme, Proizvodi i Tržišta. Činjenice o prodaji su skladištene u presecima svih dimenzija u kocki. Korisnik koji nadgleda prodaju malina u Milanu želi upit za Q4 prodajne vrednosti.

Glavna svrha OLAP baza podataka je da obezbede fleksibilne modele za pronalaženje podataka. Dimenzije i hijerarhije omogućavaju tu fleksibilnost. Dimenzije omogućavaju *slice* i *dice*:

- ***Slice* (kriška ili podskup kocke)** - izbor jednog člana iz dimenzije. Na primer: ukoliko želite da se fokusirate na samo jedan proizvod, *slice* vam omogućava da ignorišete sve osim željenog proizvoda.
- ***Dice* (domine ili slično pivot tabeli)** – kada primenjujete *dice* na kocki, onda postavljate više članova iz jedne dimenzije na jednu osu i više članova druge dimenzije na drugu osu. Ovakav način vam omogućava da sagledate međuodnose članova različitih dimenzija.

| | | Country | | | |
|---|---|---|---|---|---|
| | | Canada | Mexico | USA | Grand Total |
| Category | | | | | |
| Bread | Sales Units | 1046 | 1289 | 12197 | 14532 |
| Dairy | Sales Units | 1884 | 2407 | 10717 | 15008 |
| Meat | Sales Units | 785 | 1470 | 3901 | 6156 |
| Grand Total | Sales Units | 3715 | 5166 | 26815 | 35696 |

Slika 8.4. Zbirni prikaz (*dice*)

Hijerarhija vam omogućava *drill down* i *drill up*:

- ***Drill Down*** - Sve dimenzije sadrže hijerarhiju i za većinu dimenzija hijerarhija se sastoji od više nivoa. Više nivoa hijerarhije omogućava *drill down* po jednom članu hijerarhije. *Drill down* omogućava da se fokusirate samo na određene podatke ili oblast problema.

| Category | Subcategory | Product Name | Sales Units |
|---|---|---|---|
| ⊟ Bread | ⊟ Bagels | Colony Bagels | 264 |
| | | Fantastic Bagels | 299 |
| | | Great Bagels | 334 |
| | | Modell Bagels | 304 |
| | | Sphinx Bagels | 340 |
| | | Total | 1541 |
| | ⊞ Muffins | | 6445 |
| | ⊞ Sliced Bread | | 6546 |
| | Total | | 14532 |

Slika 8.5. Fokus na konkretne podatke (*drill down*)

- ***Drill Up*** – Vide se samo zbirne informacije članova. Omogućava da se sagleda opšta slika.

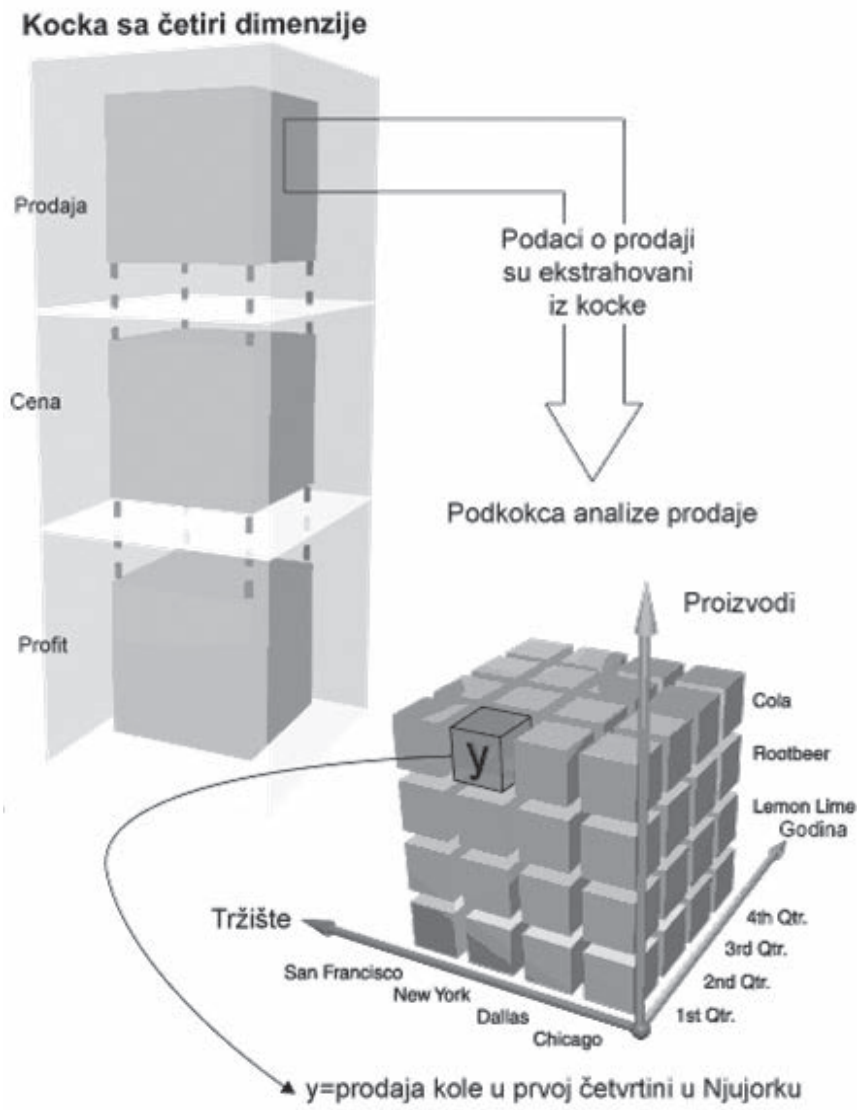
OLAP pristup mora od hardvera da poseduje poseban računar, tzv. OLAP server, na koji se povezuju relacione BP, eksterni izvori podataka i ostali interni podaci. OLAP serveri koriste višedimenzione strukture za čuvanje podataka i veza između njih. Višedimenzione strukture se najbolje vizuelizuju kao kocke podataka i kao kocke u kockama podataka. Svaka strana kocke se naziva dimenzijom. OLAP serveri podržavaju tipične analitičke operacije:

- konsolidacija – ovom operacijom se vrši agregacija podataka po zadatom kriterijumu;
- *drill down/up*
- isecanje (*slice* & *dice*)

Interfejs OLAP sistema treba da omogući korisniku komforan rad, samostalno izvođenje analitičkih operacija i dobijanje pregleda i poslovne grafike, bez znanja programiranja i strukture baze podataka. OLAP alati veoma efikasno omogućuju prelaženje sa tabela na višedimenzione grafikone korišćenjem ekrana koji su dinamički promenljivi (*dashboards*). Ovako definisana OLAP ili hiper kocka sadrži desetine hiljada mogućih izveštaja koji se lako menjaju, brzo definišu i još brže izvršavaju.

Princip rada OLAP analize je sledeći (Slika 8.6):

- Definisanje izvora podataka;
- Definisanje strukture kocke;
- Podešavanje dimenzija i mera;
- Izvršavanje upita nad kockom.




Slika 8.6. OLAP analiza [5]

## 8.1. Arhitekture OLAP sistema

Prenošenje samo izmenjenih podataka zahteva kompleksno programiranje i održavanje, dok kompletno zanavljanje podataka u skladištu postaje ozbiljan zadatak kada osnovna baza naraste. Neki od OLAP alata fizički prenose sve povezane transakcione podatke iz relacione baze podataka i iz drugih izvora u višedimenzionalnu bazu podataka, koristeći meta nivo i pune skladište podataka preko noći (*batch* pristup), odnosno izvode kompletno zanavljanje podataka u određenim vremenskim intervalima. Drugi pristup, tzv. *on-line*, prenosi svaku pojedinačnu izmenu relacione u višedimenzionu bazu podataka. Ovo su dva krajnja pristupa, međutim, obično se koristi srednje rešenje, gde je *batch* osnovna metoda, dok se samo neke izmene na starim podacima prenose pojedinačno.

Oba gore definisana pristupa uslovila su pojavu dve osnovne arhitekture, tzv. multidimenzioni OLAP (MOLAP) i relacioni OLAP (ROLAP). MOLAP je rešenje kada se koriste multidmenzione baze podataka, a ROLAP nastaje kao nadgradnja relacionih baza podataka. Tendencija je ka kombinovanju ova dva pristupa za izgradnju skladišta podataka. Dakle, trenutno postoje sledeće arhitekture OLAP sistema:

- višedimenzioni OLAP (MOLAP),
- relacioni OLAP (ROLAP),
- hibridni OLAP (HOLAP).

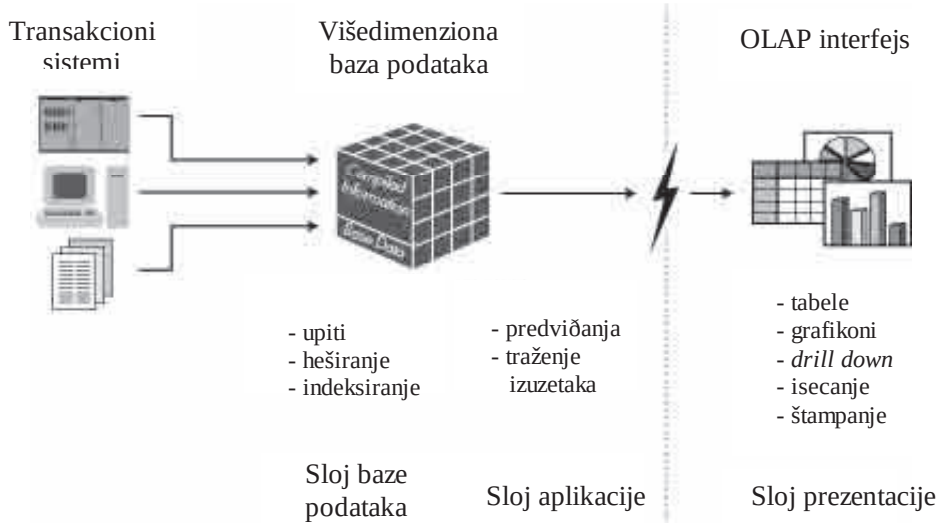
MOLAP i ROLAP se razlikuju po načinu fizičkog čuvanja podataka. Kod MOLAP sistema podaci se čuvaju u višedimenzionoj strukturi, a u slučaju ROLAP sistema podaci se čuvaju u relacionim bazama podataka.

### 8.1.1. Višedimenzioni OLAP (MOLAP)

MOLAP (*Multidimensional On-line Analytical Processing*) baze podataka imaju ograničenje fizičke veličine skupa podataka sa kojima mogu da rade i ograničenje na broj dimenzija. Da bi se vršila bilo kakva analiza, potrebno je prvo učitati podatke u višedimenzione strukture, pri čemu se vrše razni proračuni kako bi se kreirale agregacije i popunili podaci. Po završenom procesu, korisnik može započeti analizu.

Prednost MOLAP sistema je što obezbeđuje odlične performanse sistema, naročito kada se radi sa već izračunatim podacima (agregacijama).

Nedostatak MOLAP sistema je teškoća dodavanja novih dimenzija. Na sledećoj slici je prikazana arhitektura MOLAP sistema.



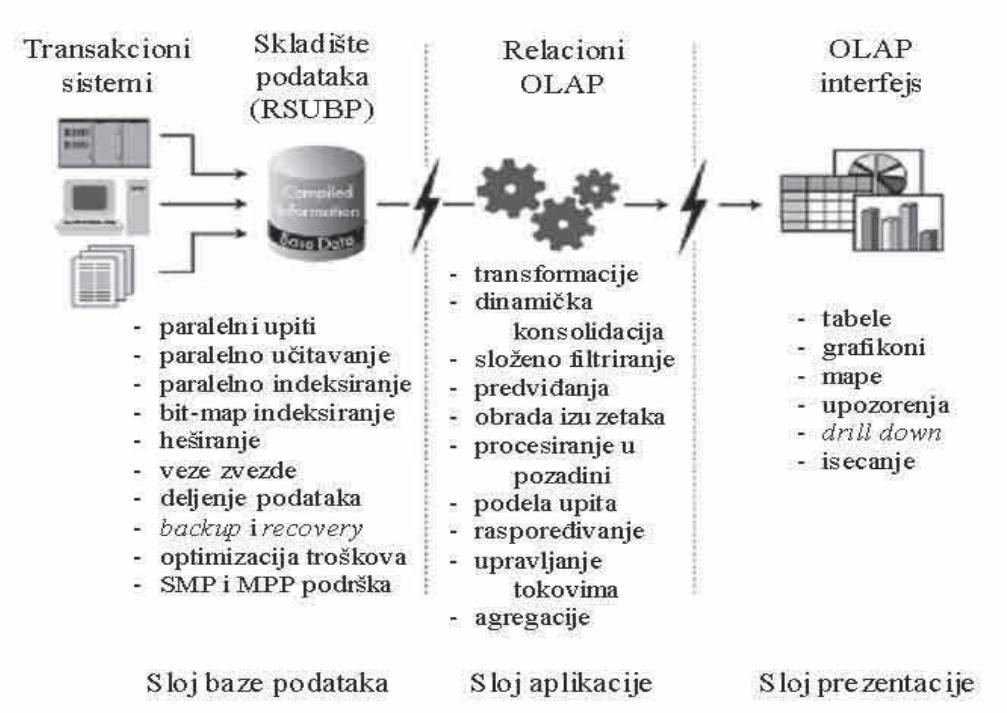

Slika 8.7. Arhitektura MOLAP sistema

Sa slike 8.7 se vidi da se podaci iz različitih transakcionih sistema učitavaju u višedimenzionu bazu podataka pomoću *batch* rutina. Kada se završi sa učitavanjem podataka, prelazi se na kreiranje agregacija, nakon čega je baza podataka spremna za rad. Korisnici zadaju svoje zahteve za OLAP izveštajima putem interfejsa.

### 8.1.2. Relacioni OLAP (ROLAP)

ROLAP sistemi pristupaju podacima direktno iz skladišta podataka i rade sa relacionim bazama podataka. Ovi sistemi mogu da rade sa velikim skupovima podataka. Čim se odredi izvor podataka, korisnik može započeti analizu. S obzirom da se radi direktno nad bazom podataka, korisniku su uvek na raspolaganju tekući podaci. Takođe, kod ROLAP sistema ne postoje ograničenja po pitanju broja dimenzija koja postoje u slučaju MOLAP sistema.

Na sledećoj slici je prikazana arhitektura ROLAP sistema.




Slika 8.8. Arhitektura ROLAP sistema

Nakon što se definiše model podataka za skladište podataka, podaci iz transakcionih sistema se učitavaju u skladište podataka. Višedimenziona analiza, koju korisnik zahteva, dinamički se transformiše u niz SQL naredbi koje se dalje prenose na relacionu bazu podataka.

ROLAP sistemi su optimizovani za pristupanje podacima, dok su MOLAP sistemi optimizovani za prikupljanje podataka. Neke od karakteristika MOLAP i ROLAP sistema su:

- višedimenziona analiza moguća je korišćenjem ROLAP i MOLAP sistema,
- za manje količine podataka ROLAP sistemi imaju skoro iste performanse kao i MOLAP sistemi,
- MOLAP sistemi nisu pogodni za rad sa velikim skupom podataka,
- MOLAP sistemi su manji od ROLAP sistema, te je potrebno manje ulazno/izlaznih (U/I) operacija pri pribavljanju podataka, što uslovljava da su MOLAP sistemi brži.

### 8.1.3. Hibridni OLAP (HOLAP)

Najbolje rešenje predstavlja HOLAP, čiji alati mogu pristupati i relacionim i višedimenzionim bazama podataka. Cilj korišćenja HOLAP alata jeste da se iskoriste prednosti MOLAP alata (kratko vreme odziva sistema i analitičke mogućnosti) i ROLAP alata (dinamički pristup podacima). Pri tome se ne može reći da je HOLAP prost zbir MOLAP-a i ROLAP-a, već je zapravo ROLAP koji ima mogućnost izvršavanja vrlo složenih SQL naredbi. Dakle, cilj je bio da se zadrže sve prednosti ROLAP-a, ali da se pri tome dodaju i neke nove mogućnosti za rad sa višedimenzionim bazama podataka. Jedan od HOLAP alata je i *Oracle Express*.

Potrebe korisnika su:

- višedimenzioni pogled na podatke – ovu mogućnost poseduju i MOLAP i ROLAP alati,
- odlične performanse sistema – ovu mogućnost poseduju MOLAP alati,
- analitička fleksibilnost (za potrebe simulacija) – ovu mogućnost poseduju MOLAP alati,
- pristup podacima u realnom vremenu – ovu mogućnost poseduju ROLAP alati,
- veliki kapacitet podataka – ovu mogućnost poseduju ROLAP alati.

HOLAP alati moraju imati sledeće karakteristike:

- dimenzije se mogu dinamički ažurirati – ne samo da treba obezbediti brz pristup samim podacima, već se mora omogućiti i jednostavna izmena struktura podataka;
- mogućnost višedimenzionih pogleda zasnovanih na meta podacima relacionih sistema za upravljanje bazama podataka – HOLAP alati moraju koristiti meta podatke relacionih DBMS pri kreiranju višedimenzionog modela;
- brz pristup svim nivoima agregacionih podataka;
- jednostavno održavanje agregacija.

Sledeći akronimi se takođe ponekad koriste, ali nisu toliko rasprostranjeni kao prethodni:

- *desktop* OLAP (DOLAP) - obezbeđuju lokalnu multidimenzionalnu analizu na klijent mašini nad podacima koji su prikupljeni i

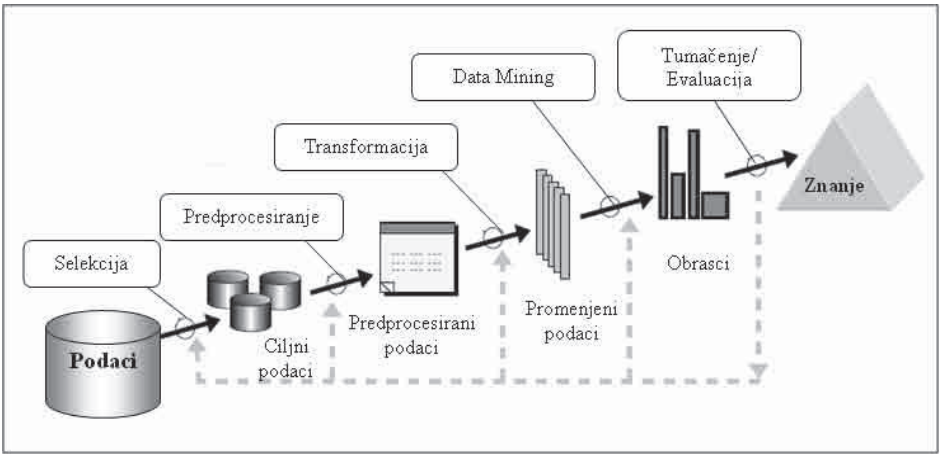
sa relacionih i multidimenzionalnih servera baze podataka. Korisnik *download*-uje manje kocke sa *data mart*-ova ili *data warehouse*-a radi obavljanja multidimenzionalne analize i to u *off-line* režimu. Ova funkcionalnost je posebno korisna za mobilne korisnike.

- Web zasnovan OLAP (WOLAP) – rešenje koje spaja dve vrlo dinamične tehnologije: World Wide Web i OLAP alate. WOLAP alati obezbeđuju OLAP funkcionalnosti kao što su *drill-up* i *drill-down* kroz hijerarhije dimenzija i odlične performanse kombinovane sa svim prednostima Web aplikacija.

# 9. Otkrivanje znanja i *data mining*

Korisnici informacionih sistema s pravom zaključuju da su im uvođenjem poslovnog rešenja obećavali sve i svašta, a dobili su samo gomilu podataka. To je tačno, tim pre što se baze podataka uvećavaju zbog realnih potreba poslovnog procesa. Ono što se nameće jeste potreba za automatizacijom analize podataka. Upravo je to ono što omogućavaju tehnike i alati otkrivanja znanja u bazama podataka (*Knowledge Discovery in Databases* – KDD).

Otkrivanje znanja u bazama podataka, kraće KDD, je proces pronalaženja znanja u podacima i obezbeđivanje automatizovanih analitičkih rešenja. Proces otkivanja znanja uzima sirove rezultate iz *data mining*-a (proces izvlačenja trendova ili obrazaca iz podataka) i precizno ih transformiše u korisne i razumljive informacije. Proces ekstrakcije znanja iz velikih baza podataka, KDD vrši koristeći metode *data mining*-a (algoritme). Sveukupni koraci otkrivanja znanja su prikazani na slici.




Slika 9.1. Pregled koraka KDD procesa

Celokupan proces pronalaženja i tumačenja obrazaca dobijenih iz podataka prolazi kroz sledeće korake:

1. **Razumevanje** domena aplikacije, relevantnog prethodnog znanja i ciljeva krajnjih korisnika;
2. **Selekcija** - odabiraju se (segmentiraju) podaci po nekom kriterijumu (na primer, svi ljudi koji poseduju automobile) da bi se odredili

podskupovi podataka; kreira se ciljni skup podataka i to izborom skupa podataka ili fokusiranjem na podskup promenljivih ili na uzorke podataka nad kojim će se vršiti otkrivanje znanja;

3. **Čišćenje podataka i preprocesiranje** - faza "čišćenja" podataka, u kojoj se određene informacije odbacuju jer se smatraju nepotrebnim, a koje bi usporile obradu upita;
4. **Transformacija** – u ovoj fazi se vrši redukcija podataka i njihova prekonfiguracija kako bi postali korisni i jednostavniji za pretraživanje;
5. Izbor ***data mining*** zadatka (klastering, regresija, klasifikacija i dr.) i algoritma (izbor metoda koji će se koristiti za pronalaženje obrazaca u podacima); odlučivanje o tome koji su odgovarajući modeli i parametri; pripajanje određenog metoda *data mining*-a celokupnomm kriterijumu KDD procesa;
6. **Tumačenje/Evaluacija** - obrasci koji su otkriveni u prethodnim fazama interpretiraju se kao znanje koje se dalje može koristiti za procese podrške odlučivanju.
7. **Konsolidovanje** otkrivenog znanja.

Ne treba mešati termine KDD i *data mining*-a. KDD se odnosi na interaktivni i iterativni proces otkrivanja korisnog znanja iz podataka koji uključuje pripremu podataka, izbor obrazaca i evaluaciju znanja. *Data mining* (tzv. iskopavanje podataka) se odnosi na aplikacije algoritama za izvlačenje obrazaca iz podataka. *Data mining* otkriva obrasce i trendove u sadržaju informacije. Primer otkrivanja određenih podataka je i matični broj građana, gde su u strukturi broja smešteni podaci koji se mogu koristiti kao elementi za pretraživanje.

*Data mining* otkriva relacije našeg svakodnevnog komuniciranja sa podacima. Omogućava sagledavanje informacija na način koji ranije nije bio sagledavan. Osnovna namena *data mininga* jeste da se iz ogromne količine operativnih podataka i veza koje se ne mogu odmah sagledati definišu odgovarajuće relacije, obrasci ponašanja, što u krajnjem slučaju treba da od podataka da potrebne informacije.

Dakle, *data mining* se može definisati kao proces podrške odlučivanju u kojem se traže šabloni (obrasci) infomacija u podacima. Osnovni cilj *data mining*-a jeste otkrivanje skrivenih veza, predvidivih sekvenci i tačnih klasifikacija.

Višestruke su oblasti primene *data mining*-a. Na primer *data mining* se može koristiti za klasifikovanje grupa klijenata sa sličnim informacijama, kako bi se ciljno reklamiralo. Kada se korisnik na primer registruje na *e-commerce* Web sajt koji prodaje sportsku opremu tada DBMS prikuplja informacije o klijentu, kao što su pol, godine, omiljeni sport i dr. Korišćenjem tehnika *data mining*-a, Web sajt će prikazivati, na primer, baner sa motivima golfa za muškarce i sl.

Kada kupujete putem Interneta, ponekad vam se ponude i dodatni proizvodi za koje je Web sajt predvideo da ćete možda biti zainteresovani. Takva preporuka se zasniva na tehnikama *data mining*-a koji pretražuje obrasce klijenata koji su na primer kupili istu knjigu koju vi sada kupujete. Sistem preporučuje: "Ukoliko vam se dopada knjiga x, proverite i sledeće ponuđene knjige".

Kada uzimate kredit, banka prikuplja širok opseg informacija o vama, kao na primer prihodi, godine staža, bračni status, kreditna sposobnost itd. Korišćenjem *data mining* tehnika, banka može da predvidi da li ste dobar ili rizičan klijent za davanje kredita i takva informacija će odlučivati o odobravanju kredita.

## 9.1. Tehnike *data mining*-a

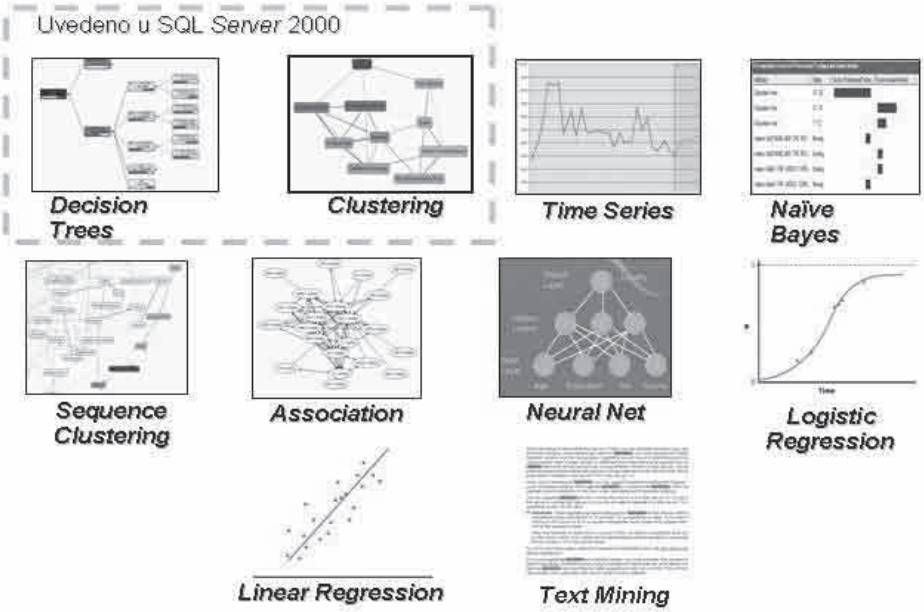
Tehnike *data mining*-a su specifične implementacije algoritama *data mining*-a koje omogućavaju identifikovanje obrazaca u ogromnim broju podataka. Modeli *data mining*-a u okviru *Analysis Services* SQL *Server*-a 2005 su (Slika 9.2):

- **Stablo odlučivanja (*Decision Trees*)** – popularan metod za klasifikaciju i predviđanje. Korišćenjem serije pitanja i pravila za kategorizaciju podataka, mogu se predvideti da će izvesni tipovi imati specifične ishode. Na primer, osoba u starosnom dobu između 25-35 godina koja zarađuje €60.000 godišnje, najverovatnije će biti zainteresovana da podigne kredit za stan nego neko u starosnoj grupi od 15-24 godina. Na osnovu godina, dohotka i dr. istorijskih činjenica, algoritam drveta odlučivanja će izračunati izglede da nekoj osobi trebaju neke određene usluge.
- **Pravila asocijacije (*Association Rules*)** – ovaj algoritam pomaže u identifikovanju relacija između različitih elemenata. On grupiše po sličnosti, odnosno koristi se za pronalaženje grupe artikala koji

se najčešće zajedno događaju u jednoj transakciji. Na primer, koristi se kod unakrsne prodaje gde se beleže veze između artikala i predviđa za koji proizvod će još biti zainteresovan da kupi. Ovaj algoritam može da radi sa enormno velikim katalozima. Bio je testiran na pola miliona artikala.

- ***Naive Bayes*** – ovaj algoritam se zasniva na *Bayes*-ovoj teoremi sa pretpostavkama o nezavisnosti promenljivih kod različitih elemenata podataka. Algoritam izračunava verovatnoću između ulaza i predvidljivih kolona i pretpostavlja da su kolone nezavisne. Na primer, promenljiva: dohodak jednog domaćinstva se razlikuje za svakog klijenta u bazi podataka i može da posluži kao predskazatelj za buduće kupovine.
- **Sekvencionalno otkrivanje obrazaca (*Sequential Pattern Discovery*) –** ova tehnika data mining-a je slična tehnici asocijacije, s tim što senkvencionalno otkrivanje povezuje događaje vremenski i određuje u kakvoj su vezi elementi tokom određenog vremenskog perioda. Na primer, koristeći ovu tehniku može se predvideti da osoba koja kupi mašinu za pranje veša bi u narednom periodu od 6 meseci sa verovatnoćom 0.7, kupila i mašinu za sušenje veša. Prodavnica ovaj podatak može da iskoristi i da ponudi kupcu koji je kupio mašinu za pranje veša popust od 10% za nabavku mašine za sušenje u roku od 4 meseca.
- **Klasifikacija (*Classification*) –** posmatra ponašanja i atribute predeterminisanih grupa. Grupe mogu biti npr., lojalni klijenti, visoki potrošači, ljudi koji se odazivaju direktnim *mail* kampanjama i sl. Primer klasifikacije bi bio otkrivanje karakteristika klijenata koji kupuju (ili ne kupuju) određenu vrstu proizvoda. Ovo znanje će rezultirati u smanjenju troškova promocije i slanja direktnih *mail*-ova.
- **Klastering *(Clustering)*** – tehnika klasteringa omogućava grupisanje zapisa podataka koji su slični na osnovu sekvenci prethodnih događaja. Na primer, sa klasteringom možete segmentirati klijente sa sličnim karakteristikama u grupe. Korisnici Web aplikacije često prate različite putanje kroz sajt. Ovaj algoritam može da grupiše klijente prema njihovom redosledu otvaranja stranica na sajtu kako bi pomogli u analizi korisnika i u određivanju koje su putanje profitabilnije od drugih. Ovaj algoritam se takođe može koristiti u predviđanju koju će sledeću stranicu korisnik posetiti.

- **Predviđanje (*Forecasting*)** – tehnika predviđanja se posmatra kroz dva aspekta:
  - **regresiona analiza (*regression analysis*)** koristi poznate vrednosti podataka za predviđanje budućih vrednosti ili budućih događaja koji se zasnivaju na istorijskim trendovima i statistici. Na primer, obim prodaje opreme za sportska kola se može predvideti na osnovu broja prodatih sportskih kola u prošlom mesecu.
  - **otkrivanje vremenske sekvence (*time sequence discovery*)** se razlikuje od regresione analize u tome što predviđa samo vremenski zavisne vrednosti podataka. Na primer, određuje procente saobraćajnih nesreća tokom praznika na osnovu broja nesreća koje su se dogodile tokom istog perioda u protekloj godini.
- **Neuronske mreže (*Neural Nets*)** – kao što čovek uči na osnovu iskustva tako može i računar. Neuronske mreže modeluju neuronske veze u ljudskom mozgu i na taj način simuliraju učenje. Ukoliko sastavljate podatke gde su ulazne i izlazne činjenice poznate, računar može da nauči iz tih obrazaca i postavi pravila i matematičke faktore kako bi npr., pomogao izračunavanje ili predvideo izlaznu vrednost. Pretpostavimo da želite da prodate kola, nekoliko faktora utiče na prodajnu cenu kao što su godine, stanje, proizvođač, model itd. Analizirajući cene kola, neuronske mreže mogu da kreiraju seriju ulaznih i izlaznih faktora kako bi predvideli cenu prodaje.
- ***Text Mining*** – ovaj algoritam analizira nestruktuirane tekstualne podatke. Na primer, kompanije mogu da analiziraju nestruktuirani podatak kao što je deo za komentare gde klijenti unose svoje utiske, zadovoljstvo o proizvodu i druge komentare.




Slika 9.2. Algoritmi *data mining*-a

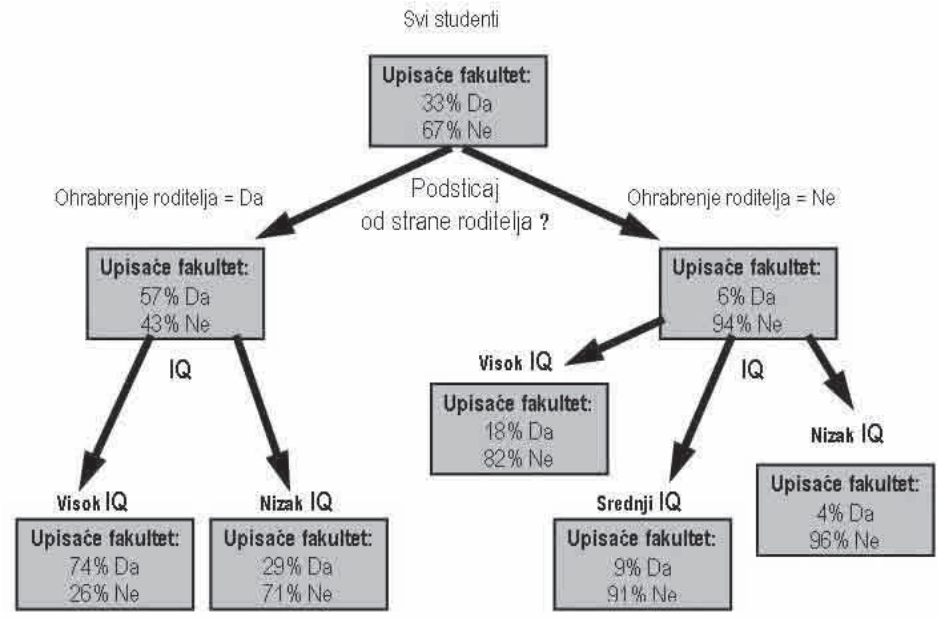
Razmotrimo sledeći primer koji tehnikom drveta odlučivanja pokušava da odgovori na pitanje: Koji je ključni atribut za predviđanje da li će svršeni srednjoškolci upisati fakultet ili ne?

Prikupljeni su odgovori na sledeća pitanja:

- Kog su pola?
- Koliki je prihod njihovih roditelja?
- Koliki im je koeficijent inteligencije - IQ?
- Da li ih roditelji ohrabruju da nastave studiranje ili ne?
- Da li planiraju da upišu fakultet?

Da bi na osnovu prikupljenih podataka utvrdili koliko studenata će nastaviti školovanje, neophodno je da se postavi upit koji broji zapise studenata koji žele i onih koji ne žele da nastave školovanje. Pretpostavimo da ste zainteresovani da odredite koji atribut ili kombinacija atributa imaju najveći uticaj da predvidi verovatnoću studenata koji će upisati fakultet. S obzirom da je ovo složeniji upit, zahteva se korišćenje tehnika *data mining*-a.

Primenjujući algoritam drveta odlučivanja (Slika 9.3) otkrivene su sledeće relacije: „Najuticajniji atribut je ohrabrivanje njihovih roditelja da upišu fakultet. Oni studenti koje roditelji ohrabruju da upišu fakultet, 60% planira da upiše fakultet i to uglavnom oni sa visokim koeficijentom inteligencije."




Slika 9.3. Stablo odlučivanja

Koraci kod izgradnje *data mining* modela su:

1. Izbor tehnike *data mining*-a – npr. stablo odlučivanja;
2. Identifikovanje slučaja (*case*) - slučaj (*case*) je element koji se koristi za klasifikaciju i grupisanje podataka (npr. klijenti ili proizvodi);
3. Izbor entiteta koji treba da se predvidi – npr. profit ili kreditna kartica i sl.;
4. Identifikovanje podataka za analizu – npr. analiziraće se određeni atributi klijenta kao što su pol, bračni status, obrazovanje, mesečni prihodi i sl.;
5. Opciono kreiranje dimenzije i virtuelne kocke iz rezultujućeg modela;
6. Obrada modela i prikupljanje rezultata.

OLAP i *data mining* su integralni delovi svakog procesa podrške odlučivanju. Korišćenjem OLAP pristupa, korisnik postavlja pitanja tipa "Koje su prodaje po gradovima i po mesecima?" i sistem daje odgovor. Međutim, sam pristup nije dovoljan, s obzirom da se na taj način dobija samo prikaz podacima koji se nalaze u skladištu podataka. Korisniku je potrebno da pretražuje podatke po više dimenzija ne bi li našao različite šablone podataka koji postoje u OLAP prostoru (na primer, kako na profit utiču dimenzije Grad i Vreme). Prema tome, OLAP sistemi se moraju fokusirati ne samo na pristup podacima, već i na proces otkrivanja znanja. Komponente OLAP *data mining*-a su:

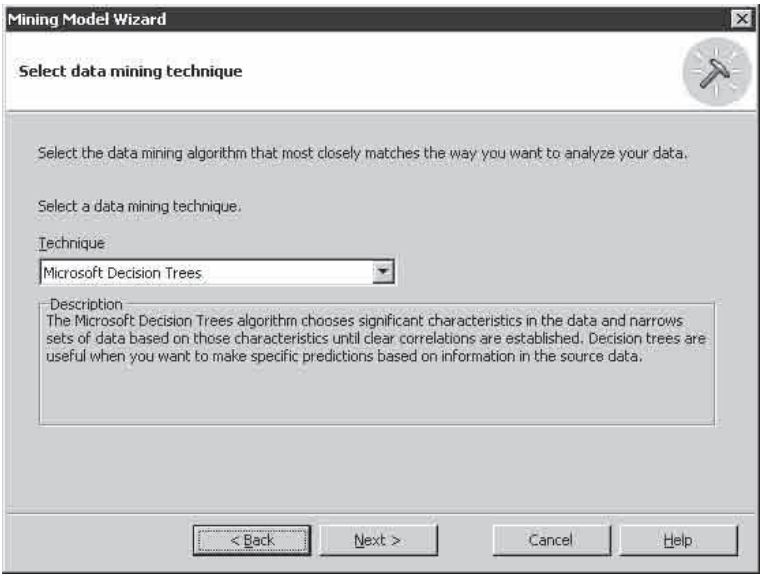
- relaciona baza podataka koja sadrži granularne podatke (ne mora biti skladište podataka),
- OLAP koji obezbeđuje brz pristup sumarnim podacima između više dimenzija,
- višedimenzioni proces otkrivanja koji će vršiti otkrivanje između dimenzija i spajati rezultate.

Osnovni koncept je da se sa objektima manipuliše u višedimenzionom prostoru. Osnovna jedinica višedimenzionog prostora je kocka, kao što je osnovna jedinica u relacionom svetu tabela. Kocke imaju atribute, kao što relacije imaju kolone. Svaka kocka predstavlja agregaciju izračunavanja nad skupom dimenzija (npr., suma svih prodaja za sve proizvode u jednoj prodavnici). Tačka kocke se naziva koordinata ili primerak (npr., prodaja za jakne u jednoj prodavnici). Za vreme procesa OLAP otkrivanja, fokus neprestano rotira između različitih dimenzionih tabela, dok se prelazi hibridni prostor da bi se izvršila analiza uticaja.

Razmotrimo sledeći primer izgradnje *data mining* modela sa OLAP podacima. Direktor marketinga želi da oceni trenutni program kreditnih kartica. Da bi zadržao postojeće klijente i ispunio njihova očekivanja, želi da identifikuje mogućnosti kako bi povećao nivo usluga kod svih vrsta kartica: zlatna, srebrna, bronzana i obična. Raspoložive informacije od klijenata su pol, bračni status, godišnji prihodi i nivo obrazovanja.

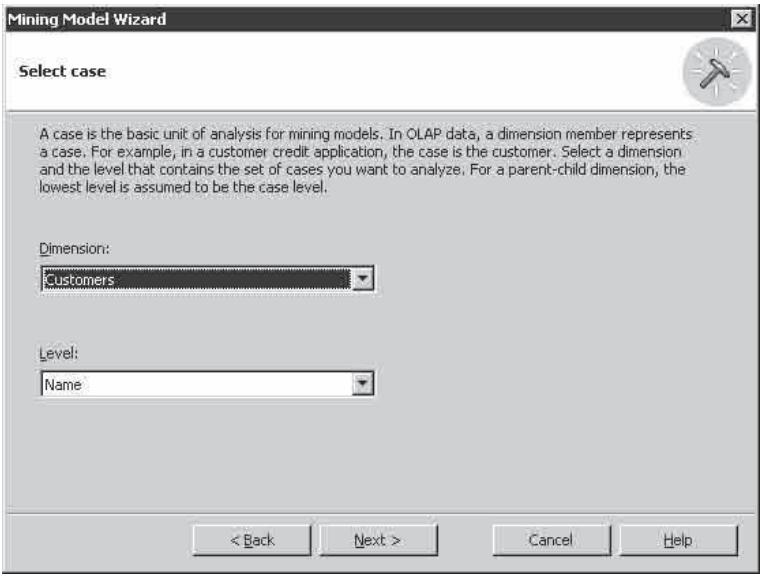
Da bi predvideli faktore koji utiču na izbor odgovarajuće kartice koristićemo d*ata mining*:

1. Odabraćemo tehniku drveta odlučivanja da bi pronašli obrazac za izbor kreditne kartice.



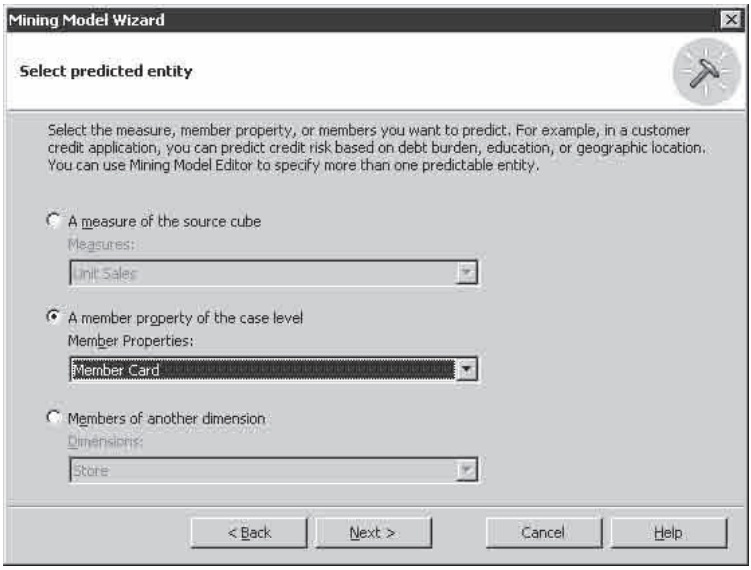

Slika 9.4. Ekran *Mining Model Wizard*-a za izbor tehnike *data mining*-a

2. Odabraćemo Klijente kao dimenziju slučaja (*case dimension*).



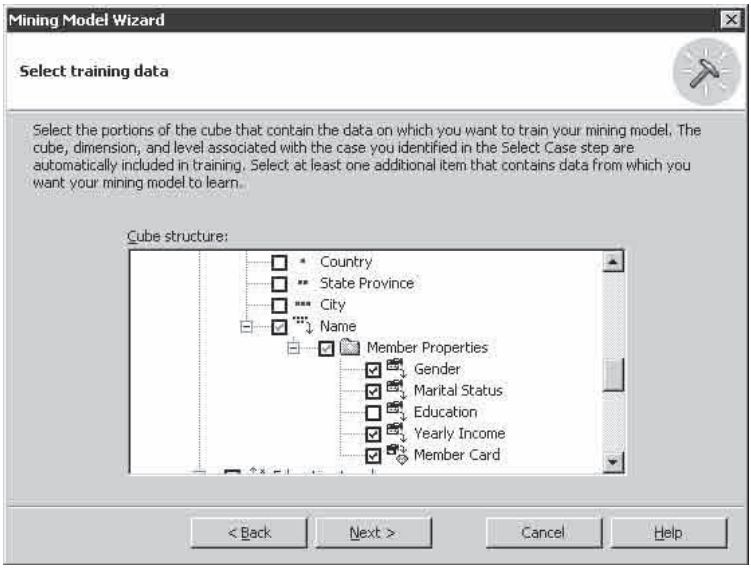

Slika 9.5. Ekran za izbor slučaja (*case*) i nivoa

3. Odabrati Kreditnu karticu kao informaciju koju će koristiti algoritam *data mining*-a da bi identifikovao obrasce.




Slika 9.6. Ekran izbora objekta predviđanja

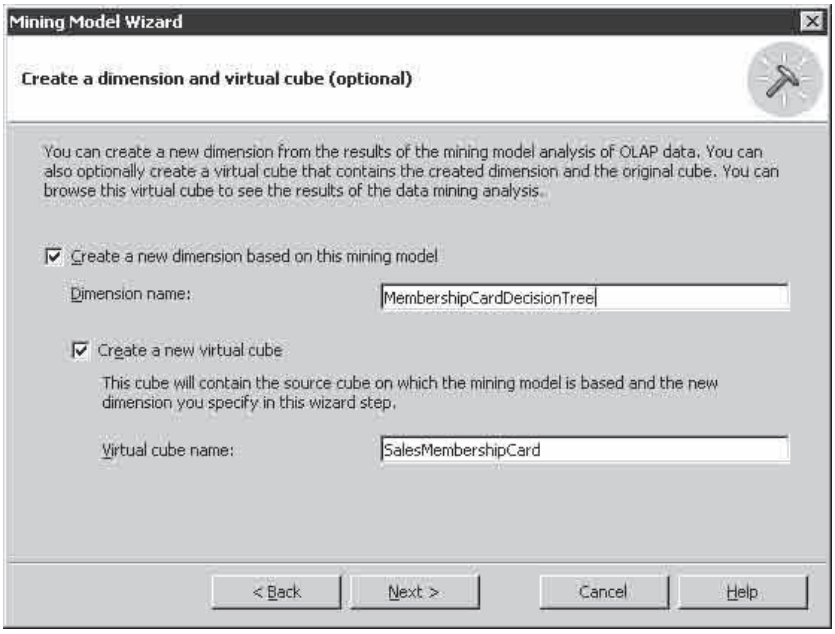
4. Iskoristiće se raspoložive informacije o klijentima kako bi se pronašao obrazac.




Slika 9.7. Ekran za izbor atributa posmatrane dimenzije

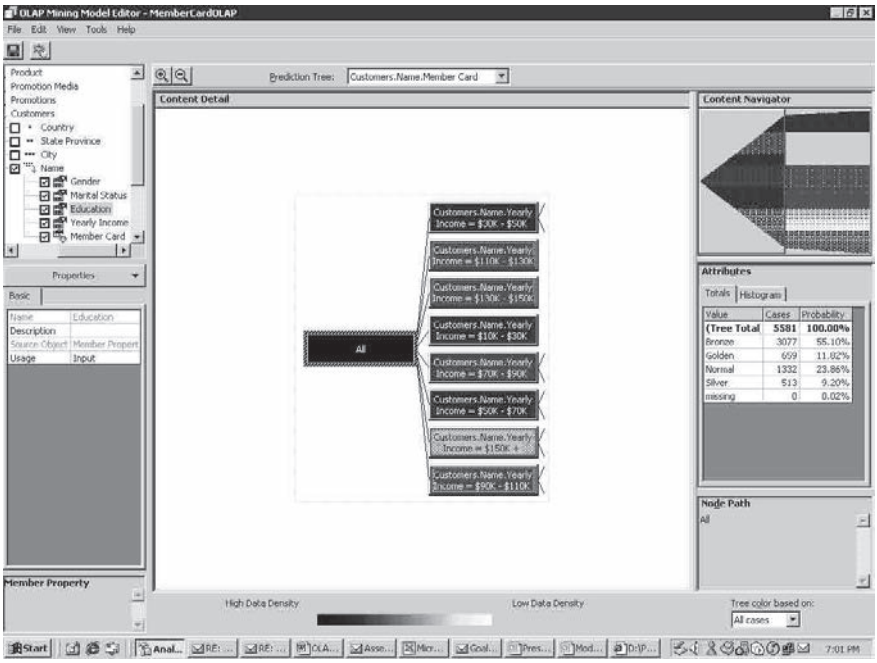
5. Ukoliko želite na interaktivan ad-hoc način da isptujete stablo

odlučivanja onda možete da uključite opciju kreiranja nove dimenzije i uključivanje iste u virtuelnu kocku.

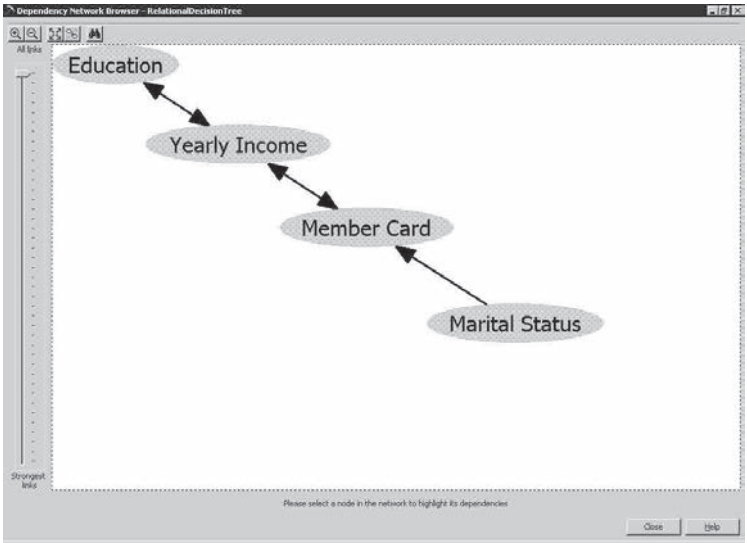



Slika 9.8. Kreiranje nove dimenzije i virtuelne kocke

6. Ispitati stablo odlučivanja – uočavanje zavisnosti među atributima. Na ovom primeru se može uočiti da je obrazovanje glavni faktor uticaja na izbor kreditne kartice. Iz ovoga bi se moglo zaključiti da bi direktor marketinga trebao da posveti pažnju edukaciji i uopšte informisanju korisnika o prednostima koje se dobijaju korišćenjem kreditnih kartica kao i načinu njihove upotrebe. Atribut bračni status najmanje uticaja ima na izbor kreditne kartice.




Slika 9.9. Ispitivanje *data mining* modela




Slika 9.10. Ispitivanje zavisnosti atributa

## 9.2. Web mining

Danas postoji nekoliko biliona HTML dokumenata, slika i drugih multimedijalnih fajlova dostupnih preko Interneta, a čiji broj se i dalje uvećava. Prikupljanje specifičnih sadržaja postaje težak zadatak. *Web mining* koristi ideje i principe *data mining*-a i KDD-a kako bi izvukao specifične podatke. *Web mining* (WM) predstavlja korišćenje *data mining* (DM) tehnika za automatsko otkrivanje i ekstrakciju korisnih informacija iz Web dokumenata.

*Web mining* se može definisati kao otkrivanje i analiza korisnih informacija na WWW-u. *Web mining* se deli na tri istraživačke oblasti, u zavisnosti od toga koji deo *Web*-a ispituju, a to su: otkrivanje sadržaja na *Web*-u (*Web Content Mining*), otkrivanje strukture veza na *Web*-u (*Web Structure Mining*) i otkrivanje obrazaca u korišćenju *Web*-a (*Web Usage Mining*).

**Otkrivanje sadržaja na Web-u (*Web Content Mining*)** predstavlja otkrivanje korisnih informacija iz Web sadržaja, podataka i dokumenata. Sadržaj Web-a se sastoji od nekoliko vrsta podataka, kao što su tekstualni podaci, slike, audio i video zapisi i metapodaci, poznatiji kao hiperlinkovi. Poslednja istraživanja u otkrivanju višestrukih vrsta podataka se nazivaju multimedijalni DM, tako da možemo smatrati multimedijalni DM kao jedan deo sadržaja *Web Mining*-a. Ipak, najveći deo podataka na Web-u čine nestruktuirani tekstualni podaci. Istraživanja u oblasti primene DM tehnika na nestruktuirane tekstove se nazivaju otkrivanje znanja u tekstovima (*Knowledge Discovery in Texts* - KDT) ili tekst *mining*.

Istraživanja urađena u oblasti *Web Content Mining*-a mogu se podeliti na dva različita pristupa: pristup zasnovan na agentima i pristup zasnovan na bazama podataka. Cilj *Web Content Mining*-a u pristupu zasnovanom na agentima je da pomaže korisnicima u pronalaženju relevantnih informacija ili u njihovom filtriranju, na osnovu odgovarajućih profila korisnika. U pristupu zasnovanom na bazama podataka cilj je da se formira model podataka na Web-u i da ga integriše sa mnogo sofisticiranijim upitima nego što su ključne reči, na osnovu kojih pretraživači vrše pretraživanja.

**Otkrivanje strukture veza na Web-u (*Web Structure Mining*)** nastoji da otkrije fundamentalni model strukture linkova na Web-u. Model mora biti zasnovan na topologiji hiperlinkova sa opisom linkova, ili bez

opisa. Ovaj model može biti korišćen za kategorizaciju Web strana i npr. u određivanju sličnosti i veza između različitih Web sajtova.

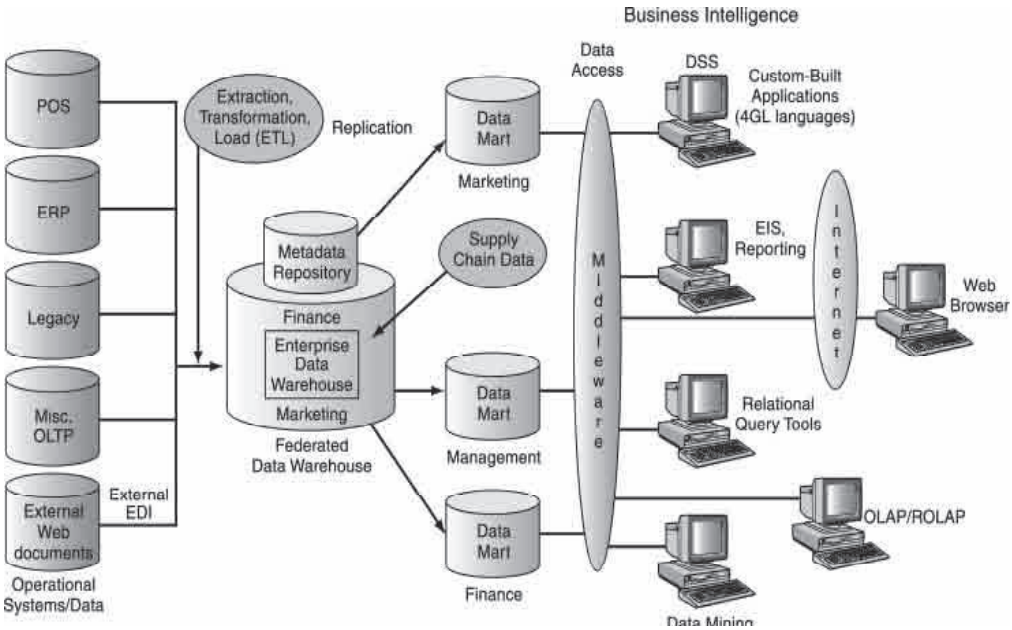
**Otkrivanje obrazaca u korišćenju Web-a (*Web Usage Mining*)** pokušava da da smisao podacima generisanim u Web korisničkim sesijama ili podacima o ponašanju korisnika. Dok *Web Content Mining* i *Web Structure Mining* koriste realne i primarne podatke na Web-u, *Web Usage Mining* otkriva sekundarne podatke izvedene iz interakcija korisnika u toku njihovog rada na *Web*-u. Ova oblast koristi podatke iz pristupnih logova *Web* servera, logova proksi servera, logova pretraživača, korisničkih profila, registracionih podataka, korisničkih sesija ili transakcija, korisničkih upita, pokretanja ili pritiskanja tastera miša i bilo koje druge podatke koji nastaju kao rezultat interakcije korisnika.

Organizacije prilikom obavljanja svakodnevnih operacija prikupljaju ogromne količine podataka koje Web serveri automatski generišu i prikupljaju u logovima za pristup serveru. Druge izvore korisničkih informacija predstavljaju referencirani logovi (*referrer logs*) koji sadrže informacije o referenciranim stranama za svaku referencu na strani, zatim informacije o registraciji korisnika ili pregledu prikupljenih podataka preko CGI skripta. Analiza ovakvih podataka može pomoći organizacijama u određivanju životnog ciklusa potrošača, životnog ciklusa proizvoda, u formulisanju marketing strategije, u efektivnijoj promotivnoj kampanji i dr. Takođe može obezbediti informacije o tome kako treba prestrukturirati Web sajt organizacije u cilju kreiranja efikasnijeg prisustva organizacije na Web-u, ali i efikasnijeg upravljanja komunikacijama između radnih grupa.

Za potrebe reklamiranja prodaje na WWW-u, analiza obrasca u korisničkim pristupima serveru može pomoći u kreiranju reklama za specifične grupe korisnika. Mnoštvo postojećih alata za Web analize obezbeđuje mehanizme za izveštavanje o korisničkim aktivnostima na serverima, kao i mehanizme za različite oblike filtriranja podataka. Korišćenje takvih alata omogućava određivanje broja pristupa serverima i pojedinačnim fajlovima, određivanje vremena pristupa, kao i naziv domena i URL-a korisnika. Ipak, ovi alati su dizajnirani u cilju smanjivanja manipulacija u poslovanju na Web-u, pa obično omogućavaju male ili skoro nikakve, analize povezanosti podataka preko pristupnih fajlova i direktorijuma unutar Web prostora.

# 10. Poslovna inteligencija

**Poslovna inteligencija ili inteligencija o poslovanju (*Business Intelligence* – BI)** je arhitektura koja predstavlja zbirni naziv za kolekciju integrisanih alata, aplikacija i baza podataka koje obezbeđuju organizaciji efikasan i lak pristup poslovnim podacima, analizu i međusobno deljenje informacija u cilju donošenja kvalitetnijih, brzih i relevantnijih odluka i poboljšanja sveukupne poslovne efektivnosti. BI softver je opšti pojam koji opisuje sisteme za podršku odlučivanju (*Decision Support Systems - DSS*), ranije izvršne informacione sisteme (*Executive Information Systems* – EIS), *data warehouse* softvere, ekspertne sisteme i *data mining* tehnike za interpretiranje podataka (Slika 10.1).



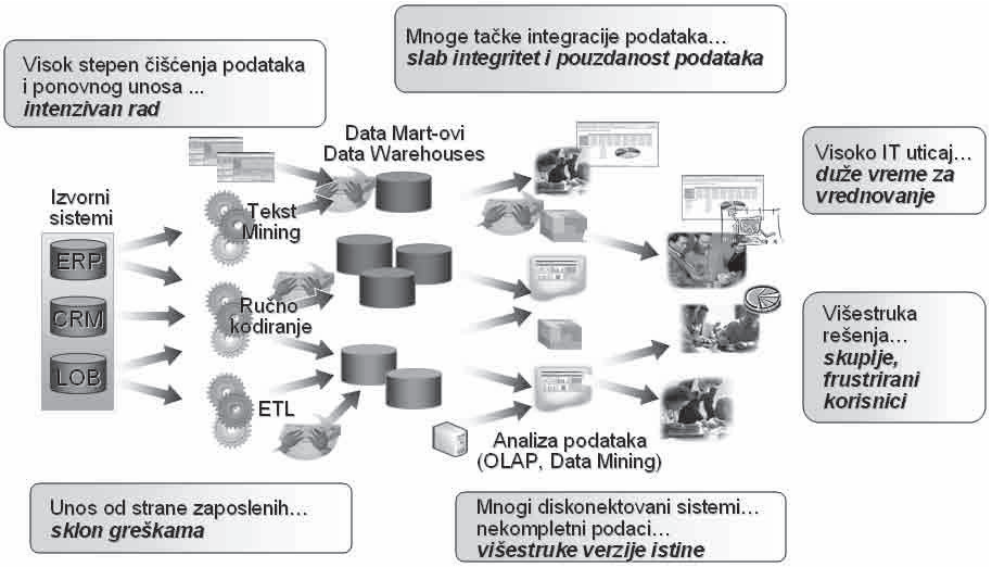

Slika 10.1. Poslovna inteligencija

BI aplikacije uključuju sledeće aktivnosti: višedimenzionalnu analizu, npr. OLAP; *data mining*; upravljanje znanjem; poslovne analize; implementaciju portala preduzeća; predviđanja i dr.

BI arhitektura predstavlja životni ciklus projekta razvoja BI aplikacija

korišćenjem struktuiranih i nestruktuiranih (*text*, *content* i *voice mining*) podataka, uključuje i izgradnju portala preduzeća sa inkorporiranim XML servisima i raspoređuje odgovarajuće poslovne role (uloge).

Razlozi uvođenja BI u organizaciju su brojni (Slika 10.2). BI daje pogled na celokupnu organizaciju, pri čemu svi zaposleni mogu dobiti informaciju koja im je u tom trenutku neophodna radi donošenja boljih, bržih i relevantnijih odluka. BI omogućava proaktivan način vođenja preduzeća, što znači da se može predvideti budućnost, izraditi nekoliko scenarija i biti pripremljen za svaku situaciju. Problem je kako pretvoriti informaciju u znanje. Danas se preduzeća vode na osnovu znanja o konkurenciji, kupcima, dobavljačima, procesima i dr. BI proizvodi znanje koje je osnova za donošenje poslovnih odluka.

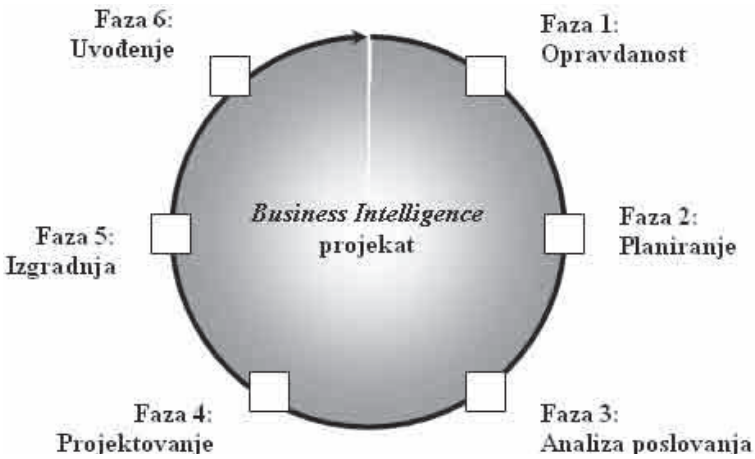



Slika 10.2. Razlozi uvođenja BI

## 10.1. Koraci razvoja BI projekta

Kao i svi inženjerski projekti i ovaj projekat prolazi kroz šest faza, kao što je ilustrovano na slici 10.3 [3]:

- **Opravdanost (*Justification*)** – procenjuju se poslovne potrebe;
- **Planiranje (*Planning*)** – razvoj strateških i taktičkih planova koji ukazuju na to kako će se projekat razvijati i uvoditi u sistem;
- **Analiza poslovanja (*Business analysis*)** – obavlja se detaljna analiza poslovnih problema ili poslovnih mogućnosti kako bi se razumeli poslovni zahtevi za potencijalnim rešenjem (proizvodom);
- **Projektovanje (*Design*)** – projektovanje proizvoda koji rešava poslovne probleme;
- **Izgradnja (*Construction*)** – izgradnja proizvoda koji treba da obezbedi povraćaj investicija u okviru predefinisanog vremenskog okvira;
- **Uvođenje (*Deployment*)** – implementacija i prodaja završnog proizvoda, zatim merenje njegove efektivnosti da bi se odredilo da li rešenje dostiže očekivani povraćaj investicija (*return on investment* – ROI).




Slika 10.3. Faze razvoja BI projekta

Detaljniji koraci razvoja BI projekta su (Slika 10.4) [3]:

**1. Faza Opravdanosti projekta**:

**1.1. Ocena poslovnih slučaja (*Business Case Assessment*)** – na osnovu definisanih poslovnih problema i mogućnosti predlaže se BI rešenje. Opravdanost BI rešenja treba da bude poslovno vođena, a nikako tehnološki.

**1.1.1. Potrebe za informacijama** - U ovom koraku neophodno je odrediti krajnje rezultate koji se žele postići analiziranjem poslovanja, kao npr.: „Zašto gubimo 50% tržišnog udela ABC kompanije u Engleskoj?“. Zatim treba definisati koje informacije se zahtevaju (oblasti poslovanja, vreme, nivoi detalja, granularnost podataka, spoljni podaci i dr.), kako bi se dobio odgovor na gore postavljeno pitanje. Sledeći korak je identifikovanje poslovnih rola (menadžeri, biznis analitičari itd.) koji će biti aktivni u različitim funkcijama podrške odlučivanju.

**1.1.2. Tipovi izvora podataka** - U ovom koraku, najveći izazov predstavlja spajanje podataka iz različitih tipova izvora podataka. Postoje tri glavna tipa izvora podataka i to:

- **operativni** – OLTP sistemi obezbeđuju operativne podatke o određenoj oblasti, kao što su finansije, logistika, prodaja, kadrovi, istraživanje, inženjering i sl.
- **privatni podaci** – podaci koji se nalaze u *spreadsheets* datotekama, bazama podataka i dr. fajlovima analitičara, statističara, menadžera i drugog osoblja koje obavlja svoje aktivnosti u okviru posmatrane oblasti;
- **eksterni izvori podataka** – podaci koji se nalaze na Internetu, u bazama podataka poslovnih partnera, a koji mogu biti razvrstani po kategorijama, npr. industrijski podaci (tehnološki trendovi, marketing trendovi, menadžment nauke, informacije o trgovini i sl.), podaci o konkurenciji (proizvodi, usluge, cene, promocije prodaja i sl.), podaci o prodaji i marketingu (npr., lista potencijalnih klijenata), kreditni podaci (podaci o kreditnoj sposobnosti klijenata, bilansi poslovanja i sl.), ekonomski podaci (politički indikatori, cene na berzi, kretanja kamatnih stopa i sl.), demografski podaci (gustina populacije, starosno doba i sl.), podaci o robi (npr., cene sirovina), psihometrijski podaci (npr., profili klijentata), meteorološki podaci (vremenski uslovi, temperature naročito za agrikulturne i putničke industrije i sl.), ekonometrijski podaci i dr.

Spajanje i standardizovanje podataka je zahtev skoro svake BI aplikacije. Najveći problem je taj što su podaci na različitim izvorima memorisani u različitim strukturama i na različitim platformama.

Slika 10.4. Detaljni koraci razvoja BI aplikacije

Problem nekonzistentnosti podataka još je veći ukoliko se doda informacija da različiti ljudi u organizacijama imaju različite autoritete u određivanju poslovnih pravila i politika.

**1.1.3. Analiza troškova i koristi** - Drugi aspekt koji se u ovom koraku razmatra je analiza troškova i koristi (*cost-benefit analysis*). Koristi BI projekta je obično teže kvantifikovati nego troškove. Jedan od efektivnijih metoda za opravdanje troškova jeste da se ukaže direktno na poslovni problem, npr., pretpostavimo da organizacija gubi 5 miliona evra svake godine jer ne može da zauzda prevare osiguranja usled nepouzdanih i nedovoljnog broja podataka. Ukoliko BI aplikacija može da reši konkretan problem, onda bi bilo veoma lako opravdati investiciju. Stoga, treba biti koliko je moguće detaljan u identifikovanju koristi, čak i onda kada je veoma teško kvantifikovati precizno ROI. Na ovaj način može se steći poverenje kod poslovnih menadžera i izvršioca i dobiti odobrenje za početak rada na BI projektu.

Kod identifikovanja koristi, neophodno je imati u vidu sledeće kategorije koristi:

- **Povećanje sveukupne dobiti** – npr., identifikovanjem novog tržišta ili efektivnija prodaja, brže prepoznavanje poslovnih mogućnosti i sl.
- **Povećanje profita** – npr., poboljšana *mail* promocija, upozorenja pada na tržištu, identifikovanje neefikasnih proizvodnih linija, efikasnije upravljanje prodajom i sl.
- **Poboljšanje satisfakcije klijenta** – npr., bolje razumevanje prioriteta klijenata, poboljšano spajanje klijent-proizvod, poboljšana prodaja klijentima, brže rešavanje žalbi klijenata i sl.
- **Povećanje uštede** – npr. smanjenje otpadaka ili smanjenje zahteva za kastomiziranim izveštajima.
- **Poboljšanje tržišne pozicije** – npr., povećan broj klijenata pridošlih od konkurencije, viši nivo održavanja klijenata u poređenju na prethodne godine i prema konkurenciji.

**1.1.4. Procene rizika (*Risk assessment*)** - rizici su faktori ili stanja koji mogu da ugroze projekat. Rizike bi trebalo oceniti prema sledećim kriterijumima:

- **Tehnološki rizik** – odnosi se na tehnologiju koja se koristi za implementaciju BI projekta. Treba razmotriti sledeća pitanja:
  - Da li je izabrana tehnologija zastarela u odnosu na tržište i organizaciju?
  - Koliko različitih tehnologija koegzistira?
  - Da li imamo nekompatibilne operativne sisteme?
  - Da li imamo nekompatibilne sisteme za upravljanje bazama podataka (DBMS)?
- **Rizik složenosti** – razmatra se kompleksnost sposobnosti i procesa koji treba da se implementiraju. Razmatraju se sledeća pitanja:
  - Koliko je kompleksno celokupno IT okruženje?
  - Koliko je složena sama BI aplikacija?
  - Koliko će se menjati *workflow*? Da li će biti kompletno redizajniran?
  - Koliko sajtova će biti podržano?
  - Koji je stepen distribuiranosti podataka, procesa i kontrola?
- **Rizik integracije** – razmatra se integracija različitih komponenata i podataka:

- Koliko će interfejsa imati BI aplikacija?
- Da li postoje uključeni i spoljni interfejsi?
- Koliko ima redundantnosti kod izvora podataka?
- Da li se mogu primarni ključevi iz različitih izvora podataka spojiti?
- Zašto nemamo standarde ili imamo nekompatibilne?
- Da li imamo zapise „bez roditelja“ kao rezultat problema referencijalnog integriteta?

- **Organizacioni rizik** – razmatraju se sledeća pitanja:
  - Koliko rizika će menadžment da toleriše?
  - Koliko rizika će tolerisati IT menadžeri?
  - Koliku finansijsku i moralnu podršku možemo očekivati kada projekat naiđe na prepreke?
- **Rizik projektnog tima** – razmatraju se veštine, ponašanja, nivoi obavezanosti kadrova za organizaciju:
  - Da li osoblje ima dovoljno iskustva u uspešnom implementiranju BI aplikacija?
  - Koliko je tim uravnotežen?
  - Kakav je moral tima?
  - Koja je verovatnoća gubitka jednog ili više članova tima?
  - Da li veštine članova tima pokrivaju sve osnovne discipline?
  - Koliko je jak i sposoban projektni menadžer?
- **Rizik finansijske investicije** – iskazuje se u ROI:
  - Za koliko se može očekivati ROI?
  - Koja je verovatnoća da troškovi prevaziđu koristi?
  - Da li se finansijski rizik može umanjiti jedino upotrebom dokazanih tehnologija?

Za ocenu rizika može se koristiti matrica ocene rizika (Tabela 10.1), gde su sa različitim bojama istaknuta jačina rizika:

- zelena – nizak rizik što ukazuje na to da se može nastaviti sa projektom;
- žuta – srednji rizik koji ukazuje da treba obazrivo nastaviti;
- crvena – visok rizik, podrazumeva da treba stati i izvršiti ponovnu evaluaciju pre nego što se nastavi dalje.

Tabela 10.1. Osnovna matrica ocene rizika

| Varijable | Nivoi rizika | | |
|---|---|---|---|
| | Zelena (nizak rizik) | Žuta (srednji rizik) | Crvena (visok rizik) |
| Tehnologija | Iskustvo sa zrelom dokazanom tehnologijom | Minimalno iskustvo sa tehnologijom | Nove tehnologije, malo iskustva |
| Kompleksnost | Minimalni uticaj na *workflow* | Umeren do srednji uticaj na *workflow* | Kritična misija, zahtevaće širi redizajn |
| Integracija | Bez integracije | Zahteva se određena integracija | Zahteva se šira integracija |
| Organizacija | Čvrsta interna podrška | Osrednja podrška | Mala interna podrška |
| Projektni tim | Iskusni, talentovani, dobar stav, vođeni poslom | Osrednje iskusni, fer stav, vođeni poslom | Bez iskustva, vođeni tehnologijom, loš stav, ograničen talenat |
| Investicija | Mogućnost dostizanja ROI u vrlo kratkom vremenskom periodu | Mogućnost dostizanja ROI-a u srednjem vremenskom okviru | Mogućnosti dostizanja ROI-a nakon nekoliko godina |

**2. Faza planiranja**

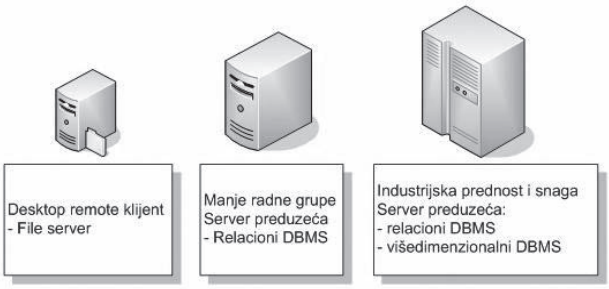
**2.1. Ocena infrastrukture preduzeća** – pojedine komponente infrastrukture preduzeća postoje, a druge bi trebalo razvijati u toku razvoja BI projekta. Kada se razmatra infrastruktura preduzeća posmatraju se dve komponente:

**2.1.1. Tehnička infrastruktura** – uključuje hardver, softver, *middleware*, DBMS sisteme, operativne sisteme, komponente mreže, skladišta meta podataka, uređaje i dr.

- hardver:
  - Novi hardver treba da se uklopi u postojeću konfiguraciju hardvera.
  - DBMS na izabranoj hardver platformi mora da se izvršava podjednako dobro sa rastom pristupa bazi podataka i upotrebe. Skalabilnost je glavno pitanje koje treba zadovoljiti.
  - Izbor platforme je ograničeno potrebom za interoperabilnošću između različitih hardverskih platformi.

- Troškovi i ROI su faktori koji moraju da se kontrolišu.
- *Middleware* – odnosi se na softverski sistem koji se nalazi u sloju između aplikacionih programa i operativnog sistema. Ponaša se kao most integracije aplikacionih programa i drugih softverskih komponenti u okruženju gde postoji nekoliko operativnih sistema, mnogo softverskih proizvoda, višestrukih čvorova mreže. *Data management middleware* povezuje aplikacije ili DBMS na jednoj platformi sa DBMS-om koji se aktivira na drugoj platformi.

- DBMS – infrastruktura baze podataka se menja sa veličinom BI okruženja, što utiče na izbor DBMS-a, kao što je prikazano na slici 10.5.




Slika 10.5. Infrastruktura baze podataka

**2.1.2. Netehnička infrastruktura** – uključuje standarde meta podataka, standarde imenovanja podataka, logički model podataka preduzeća, metodologije, vodiče, procedure za testiranje, procedure za razrešavanje sporova i sl. Netehnička infrastruktura preduzeća se opisuje kroz poslovne funkcije, poslovne procese i poslovne podatke. Svaki model sadrži definicije standarda, poslovna pravila i politike. Svrha ovih modela je da se dokumentuje skup poslovnih akcija.

**2.2. Planiranje projekta** – BI projekti su ekstremno dinamični. Svaka promena osoblja, budžeta, tehnologije, sponzora, oblasti projekta može da utiče na sam uspeh projekta. Stoga je neophodno detaljno izvršiti planiranje projekta, a svaki napredak nadgledati i izveštavati. Najčešća pitanja koja se ovde postavljaju su:

- Šta će biti isporučeno?
- Kada će rešenje biti isporučeno?

- Koliko će to koštati?
- Ko će to uraditi?

Ovakva pitanja potpadaju u četiri glavna ograničenja projekta, a to su oblast (domen projekta), vreme, budžet i resursi. Koraci kod planiranja projekta su sledeći:

- **Definisati BI projekat** – navesti ciljeve, oblast, rizike, ograničenja, pretpostavke, procedure kontrole promena, procedure o pitanjima menadžmenta.
- **Napisati projektni plan** – izlistati aktivnosti, zadatke i podzadatke, oceniti neophodna vremena za ove aktivnosti i zadatke, dodeliti im resurse, odrediti zavisnosti između zadataka, odrediti zavisnosti resursa, odrediti kritičnu putanju i na kraju kreirati detaljan projektni plan.

**3. Faza analize poslovanja**

**3.1. Definisanje zahteva projekta** – Upravljanje oblašću projekta je jedan od najtežih zadataka na BI projektima. Želja da se isprojektuje sve i to odmah je teško limitirati, tako da uskraćivanje određenih želja predstavlja jedan od važnijih aspekata pregovaranja o zahtevima. Projektni tim treba da očekuje da će se zahtevi, tokom razvojnog ciklusa, menjati, jer poslovni ljudi postaju svesniji mogućnosti i ograničenja BI tehnologija, tokom projekta. Zahtevi se posmatraju sa dva aspekta:

- **poslovni zahtevi visokog nivoa** – zahtevi za BI okruženje koji su identifikovani još u koraku BI inicijative i koji se periodično revidiraju;
- **specifični zahtevi projekta** – detaljni zahtevi koji se očekuju od svake verzije BI aplikacije.

Aktivnosti definisanja zahteva projekta su:

- **Definisanje zahteva za tehničkom infrastrukturom** – prethodno je trebalo već utvrditi da li komponente tehničke infrastrukture mogu da podrže BI aplikaciju ili se zahtevaju izmene u infrastrukturi. U ovoj aktivnosti se razmatraju zahtevi za: novim ili dodatnim hardverom; novim DBMS-om ili nadogradnjom (*upgrade*) postojećeg, novim razvojnim alatima, novim alatima za pristup i izveštavanje; novim alatima *data mining*-a; novim skladištima meta podataka i novim zahtevima za mrežom.

- **Definisanje zahteva za netehničkom infrastrukturom** – uključuje role i odgovornosti, standarde, metodologije, procese bezbednosti, procese testiranja, funkcije podrške, komunikacije i dr.
- **Definisanje zahteva za izveštavanjem** – tokom intervjua, treba sakupiti ili skicirati izveštaje i upite, definisati poslovna pravila za kreiranje agregacija, sumarnih podataka i uopšte izvođenje podataka.
- **Definisanje zahteva za izvorima podataka** – definisati detaljne zahteve za podacima i izabrati najadekvatnije izvore podataka (fajlove, baze podataka i sl.). Odrediti zahteve za čišćenjem podataka i definisati kritična poslovna pravila za podacima.
- **Pregled oblasti projekta** – uporediti detaljne zahteve sa domenom projekta. Utrvrditi da li je cilj projekta i dalje ostvarljiv u okviru određenog domena projekta.
- **Proširiti logički model podataka** – koristeći informacije iz intervjua, proširiti logički model podataka sa novim entitetima, relacijama i atributima.
- **Definisati preliminarne dogovore na nivou usluga** – razmatra se raspoloživost, bezbednost, vreme odziva, čistoća i pouzdanost podataka, podrška i sl.
- **Napisati dokument zahteva aplikacije** – navesti zahteve za funkcijama, podacima, performansama, bezbednosti i pouzdanosti.

**3.2. Analiza podataka** – najveći izazov kod BI projekata je kvalitet izvora podataka. Analiza podataka se razlikuje od faze analize sistema u okviru koje se koristi tradicionalna metodologija. Aktivnosti koje se odvijaju tokom analize podataka su usmerene ka razumevanju i korigovanju postojećih neslaganja u poslovnim podacima, nezavisno od bilo kog metoda sistemskog projektovanja ili implementacije. Stoga, analiza podataka je poslovno, a ne sistemski orjentisana aktivnost. Zahtevaju se dve komplementarne metode za analizu podataka:

- *top-down* (sa vrha na dole) logičko modeliraje podataka radi integracije i konzistencije podataka – koristi se model objekti-veze (*entity-relationship* – E-R);
- *bottom-up* (odozgo na gore) analiza izvora podataka radi standardizacije i kvaliteta podataka – ukoliko se ne obavi *bottom-up* analiza

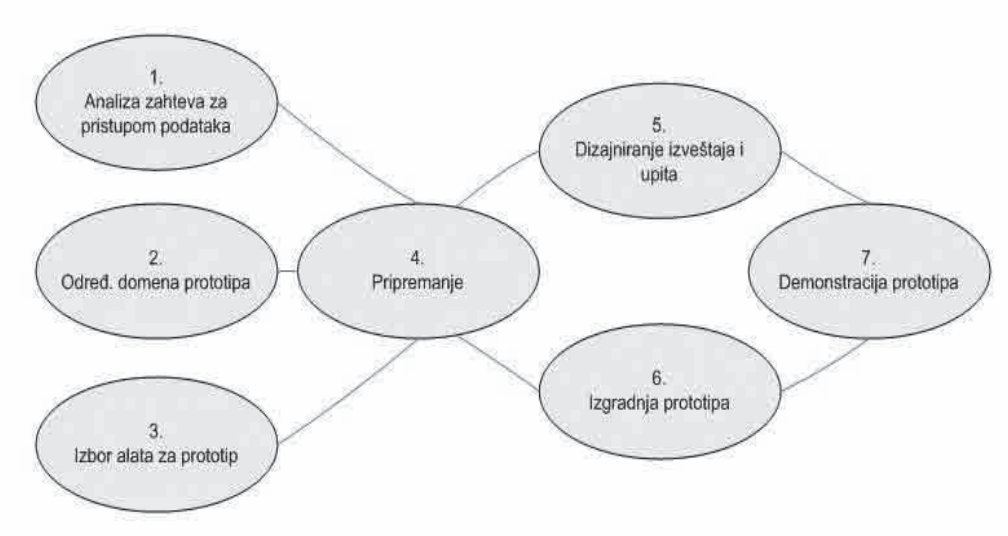
tada se problemi podataka i narušavanja poslovnih pravila i politika ne bi ni otkrili sve dok se ne implementira ETL proces. Mapiranje izvora podataka se ne odnosi samo na pravila konverzije podataka (u okviru koje se mapiraju tipovi podataka, dužina i logika programa) već i na pravila domena podataka i integriteta.

Jedan od najfrekfentnijih ciljeva kod BI aplikacija je isporuka čistih, integrisanih i usklađenih (izmirenih) podataka. Arheologija podataka (proces pronalaženja podataka), čišćenje podataka (proces korigovanja loših podataka) i kvalitet podataka (proces prevencije od grešaka u podacima) su poslovne odgovornosti, a ne IT odgovornosti. Ovo podrazumeva da poslovni ljudi moraju biti uključeni u aktivnosti analize podataka i upoznati sa pravilima mapiranja izvornih podataka. Ključne tačke selekcije podataka su:

- **integritet podataka** – Što je niži integritet podataka, veći su zahtevi za čišćenjem podataka;
- **preciznost podataka** – Da li su podaci precizni, jasni? Kako su podaci predstavljeni? Za numeričke podatke, koja je skala i preciznost podataka? Za podatke datuma, kako su formatirani?
- **ispravnost, tačnost podataka** – Da li su podaci tačni?
- **pouzdanost podataka** – Koliko su podaci stari? Da li je podatak dobijen direktno sa izvora ili iz *download*-a? Da li je poznat izvor podataka? Da li je podatak dupliciran na drugom skladištu podataka?
- **format podataka** – Što je format podataka bliži destinacionom formatu podataka, manji će biti zahtevi za konverzijom podataka. Da li su podaci iz relacione baze podataka ili iz fajlova i sl.?

**3.3. Prototipovanje aplikacije** – Prototipovanje pomaže poslovnim ljudima da sagledaju mogućnosti i ograničenja tehnologije, kako bi mogli da prilagode svoje projektne zahteve i očekivanja. Prototipovanje predstavlja jedan efektivan metod za validaciju zahteva i pronalaženje delova koji nedostaju. Poslovni ljudi obično ne misle na sve detalje kada postavljaju svoje zahteve. Često zaborave da uključe zavisne procese ili povezane podatke. Prototip im može pomoći da se fokusiraju na konkretne zahteve i da sagledaju mogućnosti BI tehnologije. Ukoliko to vreme i budžet dozvoljavaju, izgradnja prototipa omogućava testiranje, proširivanje ili izmene zahteva u ranim fazama kada još uvek nije visok uticaj na raspored projekta. Takođe, svrha prototipovanja je i da potvrdi da li bi dizajn, izabrani alati, DBMS i druge komponente tehnologije bili

odgovarajući za BI okruženje. Ukoliko sve komponente funkcionišu onako kako se i očekivalo tokom prototipovanja, onda postoje velike šanse da BI implementacija uspe. Aktivnosti prototipovanja su prikazane na slici.




Slika 10.6. Aktivnosti prototipovanja aplikacije

**3.4. Analiza skladišta meta podataka** – Skladište meta podataka je baza podataka. Međutim, za razliku od obične baze podataka, skladište meta podataka nije projektovano da skladišti poslovne podatke za aplikacije, već da skladišti kontekstualne informacije o poslovnim podacima. Primeri kontekstualnih informacija o poslovnim podacima su:

- značenje i sadržaj poslovnih podataka;
- politike koje se odnose na poslovne podatke;
- tehnički atributi poslovnih podataka;
- specifikacije koje transformišu poslovne podatke;
- programi koji rade sa poslovnim podacima.

Kontekstualne informacije o poslovnim podacima postoje u svakoj organizaciji, bez obzira na to da li su dokumentovane ili ne. Ukoliko su dokumentovane onda su poznate kao meta podaci. Ukoliko nisu dokumentovane, onda nisu poznate svima u organizaciji. Kao rezultat toga, poslovni ljudi često postavljaju sopstvena poslovna pravila, čime se kreiraju redundantni podaci i procesi, često ne razmišljajući o tome da takvi podaci možda već postoje.

Meta podaci opisuju jednu organizaciju u terminima poslovnih aktivnosti i objekata nad kojima se one izvršavaju. Na primer, zaposleni prodaje proizvod klijentu. Prodaja je poslovna aktivnost, a proizvod, klijent i zaposleni su poslovni objekti nad kojima se aktivnost prodaje izvršava. Poslovne aktivnosti i objekti, bilo da se izvršavaju ručno ili automatski, se ponašaju u skladu sa relacijama i poslovnim pravilima. Meta podaci se nalaze u CASE, ETL, OLAP i *data mining* alatima. Meta podaci dokumentuju transformacije i čišćenje izvornih podataka, obezbeđujući time praćenje i periodično punjenje podataka. Takođe, meta podaci pomažu u praćenju zahteva za bezbednošću i merama kvaliteta BI. Njima se može upravljati centralno ili distribuirano. U oba slučaja, svaka instanca meta podatka bi trebala da bude jedinstvena, bez obzira na fizičku lokaciju.

Kategorije meta podataka su:

- **poslovni meta podaci** – opisuju načine pristupanja podacima u BI okruženju u rečniku koji je razumljiv i netehničkim poslovnim ljudima;
- **tehnički meta podaci** – obezbeđuju informacije o aplikacijama i bazama podataka, koje su neophodne tehničkim licima u održavanju BI aplikacija.

Tehničke meta podatke treba prevesti u poslovne meta podatke, a zatim ih treba uskladištiti u repozitorijume (skladišta) meta podataka. Zahteve za podacima treba dokumentovati u logičkom modelu meta podataka. Aktivnosti analize skladišta meta podataka su:

- **Analiza zahteva za skladištem meta podataka** – identifikovati koji meta podaci su obavezni, bitni i opcioni. Ukoliko skladište već postoji trebalo bi ga ažurirati.
- **Analiza interfejsa** – bez obzira da li je skladište izgrađeno ili kupljeno, ono mora da prihvata meta podatke iz različitih izvora, kao što su CASE alati, Word dokumenti, *spreadsheet* datoteke i sl. Tehnički meta podaci će biti izvučeni iz DBMS rečnika, ETL alata, alata za čišćenje podataka, OLAP alata, *data mining* alata, od onih koji pišu izveštaje itd.
- **Analiza zahteva za pristupanjem i izveštavanjem** – identifikovati zahteve za pristupanjem meta podacima, zahteve za bezbednošću i *help* funkcija. Proceniti i druge načine prikazivanja kao što su PDF (*Portable Document Format*) dokumenta, HTML, SQL i sl.
- **Kreirati logički model meta podataka** – nacrtati logički model

kao E-R model i u slučaju da se implementira kao objektna (*object-oriented* – OO) baza podataka.

**4. Faza projektovanja**

**4.1. Projektovanje baze podataka** – u zavisnosti od zahteva za izveštavanjem, zavisiće i način skladištenja poslovnih podataka (detaljni ili agregirani podaci). Nisu svi zahtevi strategijski, niti su multidimenzionalni. Šema baze podataka mora da odslikava zahteve za pristupanjem informacijama. Aktivnosti projektovanja baze podataka su:

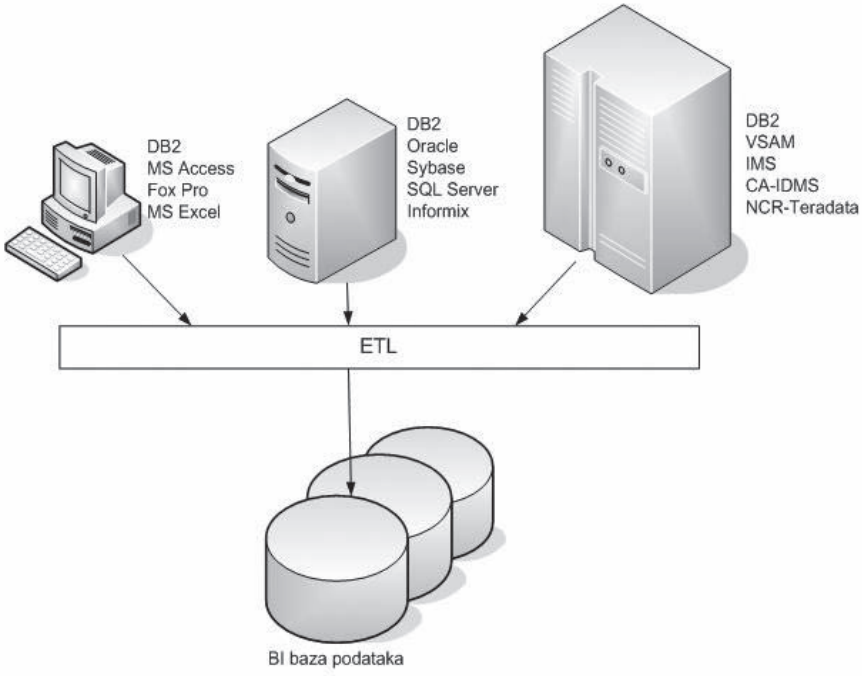
- **Pregled zahteva za pristupom podacima** – administrator baze podataka sagledava zahteve za pristupanjem i analiziranjem podataka.
- **Određivanje zahteva za sumiranjem i agregacijama** – treba obratiti pažnju na one zahteve na koje su korisnici ukazali da će možda jednog dana zatrebati.
- **Projektovanje ciljne BI baze podataka** – većina BI baza podataka će se zasnivati na multidimenzionalnoj šemi, usled potrebe za *slice* i *dice* analizama.
- **Izgradnja ciljne BI baze podataka** – fizička baza podataka je izgrađena onda kada je pokrenut DDL (*Data Definition Language*) sa odgovarajućeg DBMS-a. Bezbednost podataka je uspostavljena onda kada je pokrenut DCL (*Data Control Language*).
- **Razviti procedure za održavanje baze podataka** – kada je baza podataka puštena u rad, veoma je važno razmotriti *backup*-ovanje baze ili reorganizaciju fragmentiranih tabela.
- **Priprema za nadgledanje baze podataka** – i najbolje isprojektovana baza podataka ne garantuje dobre performanse tokom upotrebe. Jedan od razloga je što se upotreba BI baze podataka tokom vremena menja. Poželjno je praćenje performansi upita i drugih dijagnostičkih mogućnosti.
- **Priprema nadgledanja dizajniranih upita** – s obzirom da su performanse pravi izazov kod BI aplikacija, neophodno je isprobati sve varijacije upita. Paralelno izvršavanje upita bi mogao da poboljša performanse upita.

**4.2. Projektovanje ETL (Ekstrakcija/Transformacija/Punjenje) procesa** – ETL je najkoplikovaniji proces u čitavom BI projektu. Izvori podataka za BI aplikacije se nalaze na različitim platformama, koje su

upravljane različitim operativnim sistemima i aplikacijama. Svrha ETL procesa je da spoji podatke iz heterogenih platformi u standardni format (Slika 10.7).

ETL proces počinje sa reformatiranjem podataka koji treba da unificira formate podatka sa različitih izvora. U drugom koraku se rešava problem konzistentnosti koji se javlja usled redundantnosti podataka. Na kraju se pristupa čišćenju onih podataka koji narušavaju poslovna pravila.

Generalno, prvi zadatak je proces konverzije sistema gde se mapiraju najpogodniji elementi podataka u ciljne fajlove ili baze podataka. Kada se kaže „najpogodniji elementi podataka“ misli se na one podatke koji su najsličniji po imenu, definiciji, veličini, dužini i funkcionalnosti.




Slika 10.7. Heterogeni izvori podataka

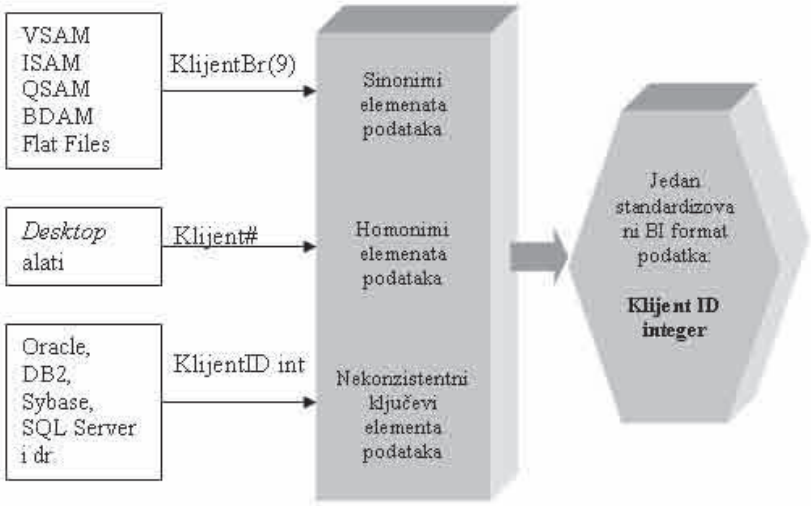
Drugi zadatak je pisanje programa konverzije (transformacije) kako bi se transformisali izvorni podaci. Ovi programi moraju da reše probleme dupliciranih zapisa, prilagođavanja primarnih ključeva i odsecanja ili povećavanja veličine elemenata podataka. Ono što uglavnom nedostaje ETL programima su čišćenje i usklađivanje podataka, na koje treba obratiti pažnju kod projektovanja procesa punjenja.

Kod procesa punjenja istorijskih podataka koji su obično statični, treba obratiti pažnju na one podatke koji nisu više u upotrebi i novih podataka koji se dodaju tokom godina.

**Programi ekstrakcije** podataka treba da vrše sortiranje, filtriranje, čišćenje i da agregiraju sve zahtevane podatke. Programi ekstrakcije moraju da prepoznaju koji od redundantnih izvornih fajlova ili baza podataka su zapisi sistema. Na primer, isti izvorni element podatka kao što je Naziv klijenta može da postoji u nekoliko izvornih fajlova i baza podataka. Ova redundantnost treba da se sortira i konsoliduje, što uključuje korake sortiranja i spajanja, preko određenih ključeva i vrednosti podataka.

Koristeći pravilo 80/20, 80% ETL procesa je transformacija podataka, dok je ostalih 20% ekstrakcija i punjenje. Projektovanje **programa transformacije** je veoma komplikovano, naročito kada su podaci ekstrakovani iz heterogenih operativnih okruženja. Tipični problemi izvora podataka su:

- **nekonzistentnost primarnih ključeva** – često se primarni ključevi izvornih zapisa podataka ne poklapaju. Na primer, može postojati pet fajlova o klijentima, gde svaki od njih ima različiti atribut kao primarni ključ klijenta. Ovi različiti ključevi klijenata se moraju konsolidovati ili transformisati u jedan standardizovani BI ključ klijenta (Slika 10.8).




Slika 10.8. Rešavanje problema nekonzistentnosti primarnih ključeva

- **nekonzistentnost vrednosti podataka** – mnoge organizacije du-

pliciraju svoje podatke. Termin dupliciranje se odnosi na elemente podataka koji su kopija originalnog podatka. Tokom vremena, usled anomalija ažuriranja, ovi duplicirani podaci imaju totalno različite vrednosti.

- **različiti formati podataka** – elementi podataka kao što su datumi i novčani podaci (*currencies*) mogu biti uskladišteni u totalno različitim formatima.
- **netačne vrednosti podataka** – da bi se korigovale netačne vrednosti podataka, mora se definisati logičko čišćenje. ETL algoritmi čišćenja podataka treba da se aktiviraju svaki put kada se podatak puni. Stoga, programi transformacije ne smeju biti pisani na brzinu, već se moraju razviti na jedan struktuiran način.
- **sinonimi i homonimi** – redundantne podatke nije uvek lako prepoznati usled toga što isti elementi podataka imaju različite nazive. S obzirom da sinonimi i homonimi[1] ne smeju postojati u BI okruženju, neophodno je preimenovati date elemente podataka.
- **ugrađena logika procesa** – neki operacioni sistemi su ekstremno stari. Oni često sadrže nedokumentovane i arhaične relacije između pojedinih elemenata podataka. Takođe, obično koriste i neke kodove, kao na primer, vrednost „00“ podrazumeva da je pošiljka vraćena, dok „FF“ znači da je prosleđena na kraju meseca. Specifikacije procesa transformacije moraju da reflektuju ovu logiku.

Pored transformisanja izvornih podataka zbog nekompatibilnosti tipa podataka, dužine ili netačnosti, najveći deo transformacione logike će uključivati i preračunavanje podataka za multidimenzionalno skladištenje.

Finalni korak kod ETL procesa je punjenje ciljne BI baze podataka, koja se postiže na dva načina, i to: unošejem novih redova u tabele ili koristeći DBMS-ov alat za punjenje. Kod projektovanja **programa za punjenje** treba obratiti pažnju na referencijalni integritet i indeksiranje.

Na kraju je neophodno dokumentovati ETL specifikacije transformacije pomoću dokumenta mapiranja izvora-ka-cilju (*source-to-target mapping document*) koji treba da lista sve BI tabele i kolone sa njihovim tipovima i dužinama podataka (Tabela 10.2). Takođe, treba prikazati ETL dijagram

1 Homonimi (homonym) su reči koje se isto pišu i izgovaraju, ali imaju različita značenja (čest slučaj u engleskom jeziku).

toka procesa (ETL *process flow diagram*) koji prikazuje zavisnosti procesa između ekstrakovanja, sortiranja i spajanja, transformacije, privremeno kreiranih fajlova i tabela, procesa rukovanja sa greškama, aktivnosti usklađivanja nekonzistentnosti i redosleda punjenja podataka.

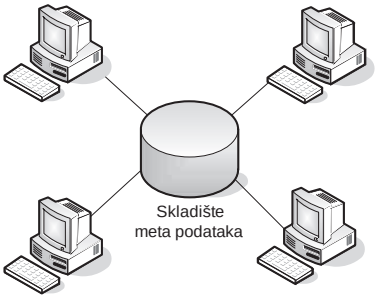
Tabela 10.2. Primer dokumenta mapiranja izvora-ka-cilju

| Tabela | Kolona | Tip/ dužina | Elementi podataka | Tip/ dužina | Izvor | Specifikacije transformacije |
|---|---|---|---|---|---|---|
| KLIJENT | KlijentID | Integer | KlijentBr | X(9) | KlijentMaster | if Code=„R“ then |
| | | | KlijentID | Int | KLIJENT | |
| | | | Klijent# | X(12) | PARTNER | Dodeli novi broj klijenta zapisu sa KlijentBr |
| | | | | | | Else |
| | | | | | | if Code="S" Dodeli novi broj klijenta zapisu sa KlijentID |
| | | | | | | Else |
| | | | | | | Dodeli novi broj klijenta zapisu sa Klijent# |

Oblast prikazivanja (*staging area*) je mesto gde se izvršava ETL proces, a koji ukazuje na dodeljeni prostor na disku, biblioteke ETL programa, privremene i trajne fajlove i tabele, kao i na dodeljeni server. Ova oblast može biti centralizovana ili decentralizovana.

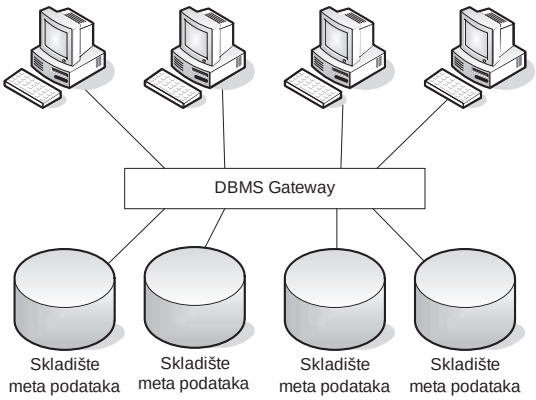
**4.3. Projektovanje skladišta meta podataka** – izvori meta podataka mogu biti CASE alati, DBMS rečnici, ETL alati, alati za čišćenje podataka, OLAP i *data mining* alati. Skladišta meta podataka mogu biti:

- **centralizovana** – postoji jedna baza podataka (relaciona ili objektno-orijentisana) i jedna aplikacija za održavanje.



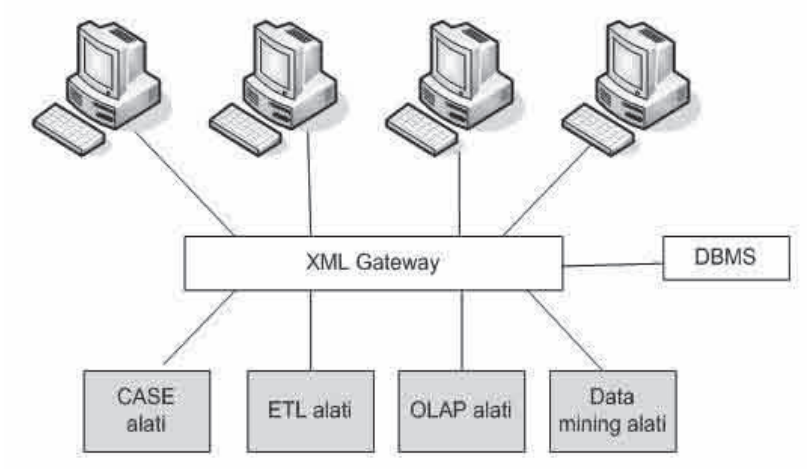

Slika 10.9. Centralizovano skladište meta podataka

- **decentralizovana** – skladište meta podatke u bazama podataka koje se nalaze na različitim lokacijama.




Slika 10.10. Decentralizovano skladište meta podataka

- **distribuirana** – preko XML rešenja, meta podaci ostaju na svojim originalnim pozicijama, odnosno na različitim alatima.




Slika 10.11. Distribuirano XML rešenje meta podataka

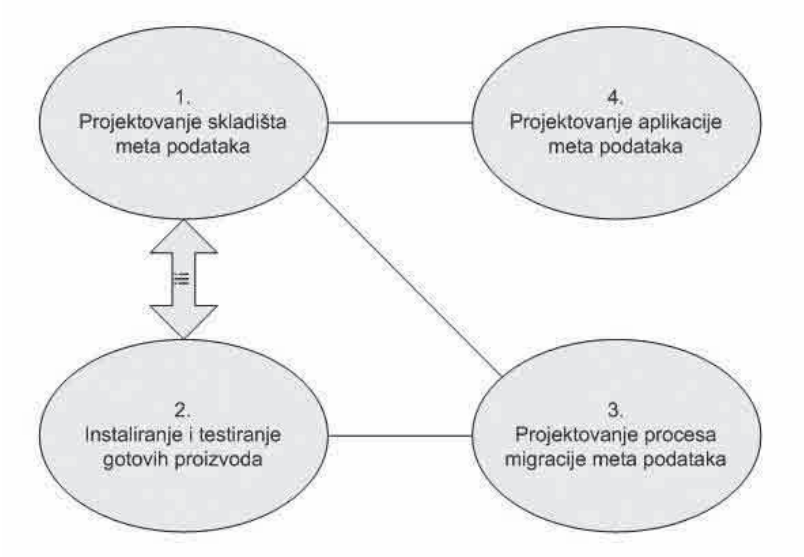
Ukoliko organizacija odluči da razvija skladište meta podataka u kući, onda treba da bira između E-R i OO dizajna. Međutim, ukoliko želi da kupi gotovo rešenje onda nikada ne treba da postavi pitanje tipa „Koji je najbolji proizvod ovog tipa na tržištu?“, već treba da postavi pitanja:

- Koji su naši zahtevi?
- Koji zahtevi su obavezni, korisni, a koji opcioni?
- Koji proizvodi odgovaraju obaveznim zahtevima, a koji korisnim?

Treba uporediti logičke modele meta podataka vendora ili njihove fizičke modele sa vašim logičkim meta modelom kako bi se uvidelo da li model vendora pokriva sve zahteve meta podataka. U svakom slučaju, model vendora mora da podrži obavezne zahteve za podacima. Takođe, treba sagledati mogućnosti proširenja skladišta meta podataka koje uključuju:

- dodavanje objekata meta podataka;
- dodavanje relacija;
- menjanje neodgovarajućih relacija;
- dodavanje atributa;
- menjanje veličina i dužina komponenata meta podataka;
- kastomiziranje izveštaja;
- imortovanje meta podataka iz drugih alata;
- eksportovanje meta podataka u druge alate.

Aktivnosti projektovanja skladišta podataka su prikazane na sledećoj slici.




Slika 10.12. Aktivnosti projektovanja skladišta meta podataka

## 5. Faza izgradnje

**5.1. Razvoj ETL-a** – u zavisnosti od zahteva za čišćenjem i transformacijom podataka, zavisiće i izbor ETL alata. BI projekti predstavljaju najbolju mogućnost eliminisanja zastarelih i beskorisnih podataka, jer omogućavaju poslovnim ljudima da sagledaju svoje zahteve za informacijama iz različitog ugla. Kada se ETL adekvatno implementira, tada aktivnosti transformacije podataka (čišćenje, sumiranje, izvođenje, agregiranje i integracija) će proizvesti podatke koji su čisti, sažeti, novi, kompletni i standardizovani, respektivno (Slika 10.13).

Slika 10.13. Aktivnosti transformacije podataka

Jedna od najčešćih žalbi na BI aplikacije je ta da se podatak u BI bazi ne poklapa sa podatkom iz izvornog sistema. Kao rezultat toga, poslovni ljudi često ne veruju BI podacima. Da bude ironija veća, u većini slučajeva podaci u BI bazi podataka su često tačniji nego podaci iz operacionih izvornih fajlova ili izvornih baza podataka, jer su oni preformatirani, standardizovani i očišćeni. Stoga se skladište meta podaci o kvalitetu podataka, statistici punjenja i sl. Često se testiranje ETL programa na BI projektima vrši veoma oskudno. Ta činjenica nije prihvatljiva, kao ni opravdanje da će to biti popravljeno u sledećoj verziji programa. Sledeća verzija će biti veća i komplikovanija, te će stoga zahtevati mnogo više vremena za testiranje.

Tipovi testiranja koji se primenjuju kod operacionih sistema, mogu se primeniti i na BI aplikacije:

- **Testiranje jedinica (*unit testing*)** podrazumeva:
  - kompajliranje (*compilation*) – testiraju se moduli i skriptovi kompajliranjem programa, gde svako kompajliranje programa mora uspešno da se završi.
  - funkcionalnost (*functionality*) – svaki modul programa mora da izvršava funkcije zbog kojih je i projektovan, kao i da obezbedi očekivane rezultate.
  - editovanje (*edits*) – svaki programski modul mora da hvata greške i da upozori na njih.

- **Testiranje integracije (*integration testing*)** je poznat i kao sistemsko testiranje:
  - interakcije – moraju se testirati interakcije između modula.
  - tok – ETL dijagram toka procesa treba da prikaže kojim redosledom se programi izvršavaju, koji se mogu paralelno odvijati i kada se mogu ubaciti operacije sortiranja i spajanja. Ovaj tok se mora testirati u odnosu na funkcionalnost i efikasnost. Testiranjem funkcionalnosti osigurava se da se pravi procesi izvršavaju nad pravim podacima u pravo vreme, odnosno da se programi izvršavaju prema pravilnom rasporedu.
- **Testiranje regresije (*regression testing*)** – cilj je da se uvidi da modifikacije nad postojećim ETL programom nisu nenamerno proizvele neke greške koje ranije nisu postojale.
- **Testiranje performansi** – proverava se ponašanje i performanse sistema. Može se izvoditi samo nad kritičkim programskim modulima.
- **Testiranje kvaliteta** – većina organizacija ima jasne procedure kod uvođenja aplikacije. Te procedure obično uključuju testiranje kvaliteta koje se odvijaju u posebnim QA (*quality assurance*) okruženjima.
- **Testiranje prihvatljivosti** – proveravaju se sve funkcije ETL procesa koje moraju biti tačne i kompletne.

**5.2. Razvoj aplikacija** – Kada prototipovanje potrvrdi sve funkcionalne zahteve, onda se može pristupiti razvoju aplikacije. Razvoj aplikacija može biti jednostavno finaliziranje prototipa ili može uključiti složenije različite alate pristupa i analize. Glavni cilj koji BI aplikacija treba da ispuni jeste da obezbedi brz i lak pristup podacima radi vršenja poslovnih analiza. Veći procenat ovog pristupa će biti zadovoljeno primenom predefinisanih obrazaca (šablona). Predefinisani obrasci podrazumevaju da su podaci izvedeni, agregirani, sumirani i uskladišteni na način da im se brže pristupa. Ovo je razlog velike popularnosti multidimenzionalnih OLAP alata koji čine glavnu komponentu BI aplikacija. OLAP alati omogućavaju inovativne načine analiziranja podataka:

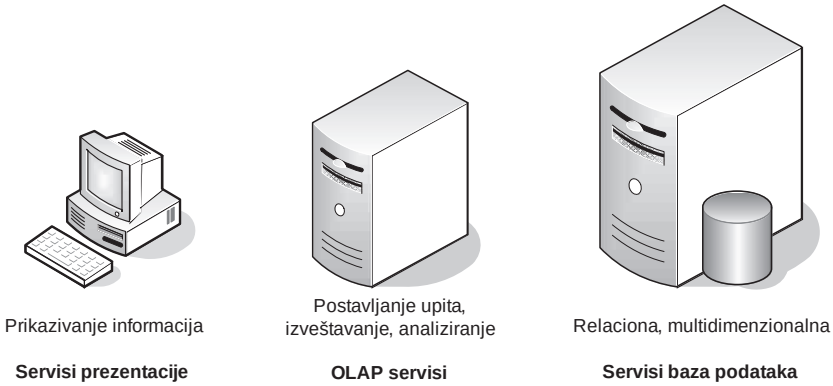
- multdimenzioni pogled na podatke koji je intuitivan i prepoznatljiv poslovnim ljudima. Na primer, organizacije uglavnom postavljaju svoje strategije kako bi povećali prihode i to bilo uvođenjem novog proizvoda, osvajanjem novih tržišta ili povećavanjem cena. Ove tri

karakteristike se mogu lako pratiti kroz multidimenzionalni pogled na podatke prodaje (po proizvodu, po regionu, po klijentu);

- sumiranje i agregacije podataka;
- interaktivno postavljanje upita i vršenja analiza tipa: „Šta-ako smanjimo cenu proizvoda za 5 evra? Koliko bi nam se povećao obim prodaje u određenom regionu?
- podrška analitičarima da mogu da kreraju članove unutar dimenzija, dodaju nove ili menjaju parametre upita ili kreiraju mere ili činjenice.
- podržavanje *drill-down, roll-up* i *drill-across* funkcija. Na primer, analitičar koji želi da pronađe način da smanji troškove proizvodnje robe, može *drill down* funkcijom da sagleda detaljne troškove kupljenih sirovina. Takođe, može da sumira troškove sirovina u predefinisane kategorije, *roll-up* funkcijom. Primenjujući *drill-across* može da ode do druge tabele kako bi uključio i troškove proizvodnje proizvedene robe.
- primena analitičkog modeliranja. Uzimajući u obzir prethodni primer, smanjivanje troškova proizvodene robe može se postići i smanjivanjem radnog kapitala. Primenom analitičkog modeliranja mogu se pronaći optimalni iznosi radnog kapitala.
- analize trendova i prognoziranje. OLAP analizira podatke iz prošlosti kako bi dao prognoze u budućnosti.
- prikazivanje podataka u grafikonima, dijagramima i tabelama omogućava vizuelni pregled, jer kao što se kaže da slika vredi hiljadu reči, tako i ove komponente čine sastavni deo OLAP alata.

OLAP arhitektura se sastoji od tri funkcionalne komponente: servisi prezentacije, OLAP servisi i servisi baze podataka (Slika 10.14).

- **Servisi prezentacije** moraju biti laki za korišćenje, fleksibilni i prilagodljivi. Često su to tabelarni izveštaji, ali se koriste i grafički i dijagramski prikazi. Meni, ikone i funkcije treba da se konfigurišu u zavisnosti od profila analitičara, njihovih računarskih sposobnosti i veština.




Slika 10.14. Funkcionalne komponente OLAP arhitekture

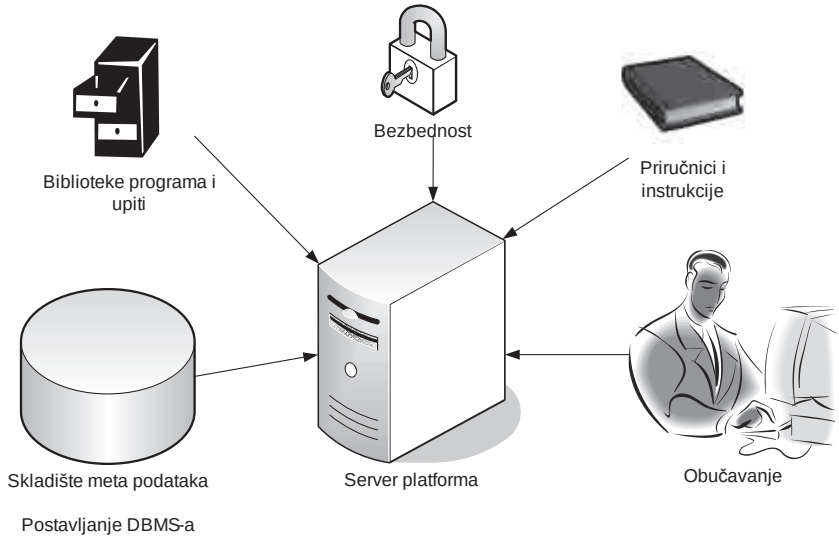
- **OLAP alati** treba da omoguće interaktivno, međupovezano i iterativno postavljanje upita, izveštavanje i analiziranje.
- **Servisi baze podataka –** u zavisnosti da li je ROLAP, MOLAP ili HOLAP.

Razvoj vitalnih poslovnih aplikacija se ne odvija ad hoc na nečijem PC-ju. Većina organizacija zahteva formalan i struktuiran pristup razvoju, testiranju i isporuci aplikacija. Organizacije obično postavljaju različita razvojna okruženja za različite namene. Veće organizacije imaju okruženje prototipovanja, razvojno okruženje, QA okruženje, proizvodno okruženje i Web okruženje.

**5.3. *Data mining*** – cilj je razvoj *data mining* baze podataka koja je projektovana i izgrađena prema specifičnom analitičkom modelu podataka i setu *data mining* operacija (algoritmi u okviru *data mining* alata)

**5.4. Razvoj skladišta meta podataka** – Bilo da je skladište meta podataka kupljeno ili razvijano u kući, pre nego što se pusti u proizvodnju trebalo bi izvršiti pripreme.

- **Server platforma** – Produkcionu server platformu, gde će biti smešteno skladište meta podataka, treba instalirati i testirati, što uključuje hardverske komponente, operativni sistem, uređaje za nadgledanje i mrežnu konekciju.
- **Produkcioni DBMS** – ukoliko se skladište meta podataka instalira na novom produkcionom serveru, tada bi trebalo kreirati instancu DBMS-a, podesiti parametre i testirati nad operativnim sistemom.

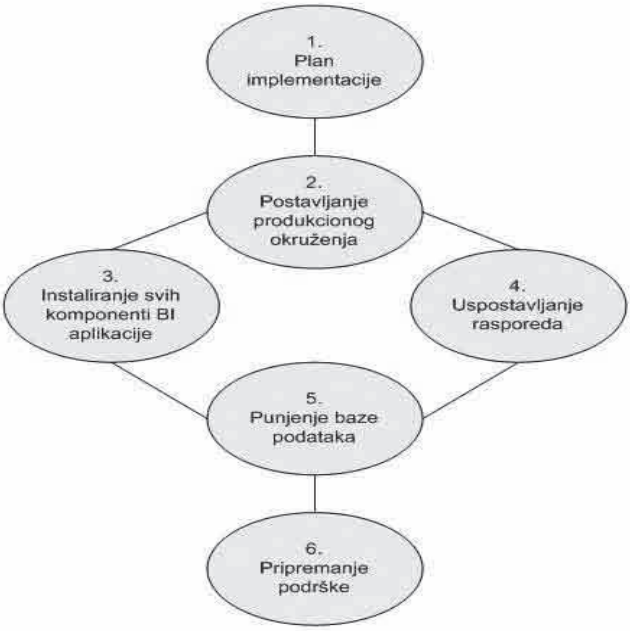



Slika 10.15. Pripremni koraci uvođenja skladišta meta podataka

- **Programske biblioteke i upiti** – svi meta podaci o migraciji programa (programi interfejs alata) i svi meta podaci aplikacionih programa (programe pristupa interfejsu, *on-line help* funkcije, izveštaje i upite) će se nalaziti u biblioteci. Programi za upravljanje bibliotekom se moraju instalirati i testirati pre nego što se puste u proizvodnju programi skladišta meta podataka.
- **Bezbednost** – produkcioni server, DBMS, skladište meta podataka i svi programi moraju da imaju implementirane odgovarajuće nivoe bezbednosti.
- **Priručnici i instrukcije** – kada se jednom pusti u proizvodnju skladište meta podataka, sa svakim ETL procesom prikupljaće se statistika punjenja i kvalitet podataka koji će se puniti u skladište meta podataka.
- **Obuka** – poslovni ljudi i tehničko osoblje treba da se obuče kako da koriste skladište podataka, bilo preko interfejsa, direktno ili interaktivno. Analitičari treba da znaju kako da pristupaju skladištu meta podataka kako bi selektovali BI podatke za izvršavanje ad hoc upita. Tehničko osoblje mora da zna kako da koristi meta podatke kako bi znali da održavaju BI aplikacije i da isporučuju meta podatke kao integralni deo izveštaja i upita BI aplikacija.

**6. Faza uvođenja**

**6.1. Implementacija** – Kada je BI aplikacija razvijena i testirana, tada je ona spremna da bude implementirana u proizvodno okruženje. Upravljanje bezbednošću treba da se prožima kroz sve korake razvoja BI aplikacije, ali se ovde posebno posmatra bezbednost za Internet pristup. Druga važna problematika o kojoj mora da se vodi računa je *backup* i oporavak baze podataka. Naredni korak je praćenje upotrebe resursa. Ovde se misli o angažovanosti računara, mreže i ljudi. Ono što, na kraju, treba imati u vidu jeste vremenski rast podataka, korisnika i hardvera. Aktivnosti implementacije su prikazane na slici 10.16.




Slika 10.16. Aktivnosti implementacije

**6.2. Evaluacija uvođenja** – Sa izgradnjom BI okruženja ne završava se proces razvoja BI projekta. BI projekti uvode mnoge nove prakse: nove tehnologije za analitičare, nove tehnike prototipovanja, nove tehnike projektovanja, nove arhitekture i nove tehnologije.

Navedene korake razvoja BI rešenja ne morate izvršavati sekvencionalno, mnogi projektni timovi će ih izvršavati paralelno.

## Rezime

1. Skladištenje podataka (*Data Warehousing* – DW) je predmetno orjentisana, integrisana, vremenski zavisna kolekcija podataka koja podržava proces donošenja menadžerskih odluka. DW je relaciona/multidimenzionalna baza podataka koja je projektovana za analizu i postavljanje upita, a ne za obradu transakcija. DW obično sadrži bivše podatke izvučenih iz transakcionih sistema. Okruženje DW se sastoji od ETL rešenja, OLAP *engine*, alata za klijent analizu i drugih aplikacija koji upravljaju procesom prikupljanja podataka i prikazivanje poslovnim korisnicima.
2. Data mart je podstkup DW koji podržava određeni region, poslovnu jedinicu ili funkciju. Data mart je projektovan za određenu liniju poslovanja kao što je prodaja, marketing ili finansije.
3. Dimenziono modeliranje podataka uključuje jednu ili više dimenzionih tabela i tabelu činjenica. Primeri dimenzija su lokacija, proizvod, vreme, organizacija itd. Dimenzione tabele skladište zapise koji se odnose na određene dimenzije (npr. dimenzija proizvod će sadržati kategoriju proizvoda, podkategoriju, naziv proizvoda i karakteristike proizvoda), a one neće sadržati činjenice, tzv. mere (npr. vrednost prodaje, ukupna količina prodatih proizvoda i sl.). Tabela činjenica sadrži mere i spoljne ključeve dimenzionih tabela. Dimenziono modeliranje podataka se koristi za računanje izvedenih (zbirnih, prosečnih i sl.) podataka.
4. Šema zvezde je relaciona šema baze podataka za predstavljanje multidimenzionalnih podataka.
5. ETL alati ekstrakuju, transformišu i pune podatke u DW. ETL pristupaju i rukuju izvorima podataka i pune ih u ciljne baze podataka. Prvi korak kod ETL procesa je mapiranje podataka iz izvornih sistema u ciljne baze podataka (DW ili *data mart*). Drugi korak je čišćenje izvornih podataka u oblasti za pripremu podataka (*staging area*). Treći korak je transformacija očišćenih izvornih podataka i njihovo punjenje u ciljne sisteme.
6. OLAP (*On-line Analytical Processing*) je pristup koji pomaže organizaciji da koristi sve prednosti podataka. OLAP kocke omogućavaju analizu podataka kroz višestruke dimenzije obezbeđujući multidimenzionalni pogled agregiranih i grupisanih podataka. OLAP ap-

likacije uključuju analize prodaje i klijenata, marketing analize, analize profitabilnosti, proizvodnje, predviđanje. Popularni OLAP alati su Cognos, Business Objects, Micro Strategy itd.

7. ROLAP (*Relational On-line Analytical Process*) obezbeđuje multidimenzionalnu analizu podataka koji su uskladišteni u relacioni sistem baze podataka (RDBMS).
8. MOLAP (*Multidimensional* OLAP) obezbeđuje analize podataka uskladištenih u multidimenzionalne kocke podataka.
9. HOLAP (*Hybrid* OLAP) je kombinacija ROLAP-a i MOLAP-a i može da obezbedi simultanu multidimenzionalnu analizu podataka uskladištenih i u multidimenzionalnoj bazi podataka i u relacionoj bazi podataka.
10. DOLAP (*Desktop* OLAP ili *Database* OLAP) obezbeđuju lokalnu multidimenzionalnu analizu na klijent mašini nad podacima koji su prikupljeni i sa relacionih i multidimenzionalnih servera baze podataka.
11. *Data mining* je otkrivanje skrivenih veza, predvidivih sekvenci i tačnih klasifikacija među podacima. *Web mining* se deli na tri istraživačke oblasti, u zavisnosti od toga koji deo *Web*-a ispituju, a to su: otkrivanje sadržaja na *Web*-u (*Web Content Mining*), otkrivanje strukture veza na *Web*-u (*Web Structure Mining*) i otkrivanje obrazaca u korišćenju *Web*-a (*Web Usage Mining*).
12. Poslovna inteligencija (*Business Intelligence*) je tehnologija koja se zasniva na modelima orjentisanih na klijente i profit u cilju smanjenja operativnih troškova, povećanja profitabilnosti poboljšanjem produktivnosti, prodaje, usluga i donošenja boljih poslovnih odluka. Modeli poslovne inteligencije se zasnivaju na multidimenzionalnoj analizi i ključnim indikatorima performansi (*Key performance indicators* – KPI) preduzeća. Aplikacije poslovne inteligencije zasnovanih na modelima poslovne inteligencije se kreiraju softverima poslovne inteligencije koji obezbeđuju agregirane detalje o dobavljačima, klijentima, internim aktivnostima, poslovnim transakcijama menadžerima i svima onima koji treba da donesu bolje odluke. Alati poslovne inteligencije pomažu u prikupljanju, skladištenju, pristupanju i analizi korporativnih podataka u cilju podrške odlučivanju.
13. KPI su faktori koji utiču na uspešno poslovanje organizacije. KPI može da se odnosi na regionalne prodaje prodavaca, statistiku lanca snabdevanja dobavljača, produktivnost jedinica, zadovoljstvo klijenata, rast klijenata itd. Faktori koji utiču na izbor KPI su:

a. Merljivost – KPI mora da bude merljiv odnosno da bude u terminu brojki.
b. Reflektovanje organizacionih ciljeva – KPI mora da vodi poslovanje ka uspehu.
c. Mogućnost iniciranja poslovnih akcija – treba da pomognu menadžerima u iniciranju poslovne akcije kao rezultat analiza i mera KPI-a.

14. *Dashboard* poslovne inteligencije vizuelno predstavlja ključne organizacione podatke u realnom vremenu i na razumljiv način. *Dashboard* prikazuje KPI-jeve i omogućava menadžeru *drill-down* i druge opcije analize. Tipične karakteristike su Web interfejs, izveštaji, grafici i dijagrami i dr.

## Pitanja za razgovor i vežbe

1. Objasnite razliku između skladišta podataka (DW) i OLAP sistema.
2. Da li je moguće ažurirati podatke koji su uskladišteni u DW?
3. Objasnite OLAP arhitekture.
4. Navedite i objasnite tehnike data mining-a.
5. Koje su prednosti alata poslovne inteligencije (*Business Intelligence* - BI)?
6. Navedite opšte korake razvoja posovne inteligencije.
7. Navedite aktivnosti transformacije podataka u okviru ETL alata.

## Reference


[1] Adelman, S., L.T. Moss, *Data Warehouse Project Management*, Addison-Wesley, Boston, MA, 2000.

[2] Berson, A., S.J. Smith, *Data Warehousing, Data Mining and OLAP*, McGraw-Hill, New York, 1997.

[3] Moss, L.T., S. Atre, *Business Intelligence Roadmap – the Complete Project Lifecycle for Decision-Support Applications*, Addison-Wesley, 2003.

[4] Veljović, A., A. Njeguš, *Osnove relacionih i analitičkih baza podataka*, Megatrend univerzitet, Beograd, 2004.

[5] Visual Studio 2005 – Net arhitektura, prezentacija, Sinergija 2006, Microsoft.

# DEO IV: MODULI POSLOVNIH INFORMACIONIH SISTEMA

## Ciljevi poglavlja

Svako poslovno rešenje pokriva glavne poslovne procese organizacije. U ovom delu se analiziraju najprimenjiviji moduli današnjih ERP rešenja i to: prodaja i marketing, računovodstvo i finansije, proizvodnja i upravljanje materijalima, ljudski resursi, menadžment lanca snabdevanja i elektronsko tržište. Svi moduli su integrisani, što znači da različita odelenja u okviru organizacije koriste iste podatke u isto vreme. Takođe, omogućavaju povezivanje poslovnih procesa između organizacija širom sveta. Baza podataka je integrisana, tako da svi procesi koriste iste podatke. Na primer, modul prodaje i marketinga koji podržava proces prodaje i distribucije sa funkcijama za cene, obrađivanje porudžbina i isporuku, je direktno povezan sa modulom proizvodnje i upravljanje materijalima. Ovo omogućava integrisani proces na primer, provere kredite kupca, osiguranja materijala, proizvodnih kapaciteta koji zadovoljavaju pravovremenost narudžbine, izvršavanje narudžbine i automatski proces naplate. Ovaj modul omogućava i analilzu prodaje i isporuke.

Nakon proučavanja ovog dela, studenti će:

- razumeti glavne poslovne procese u organizaciji;
- prepoznati tokove i veze između poslovnih procesa koji podržavaju prodaju i marketing, proizvodnju, računovodstvo i finansije i ljudske resurse;
- razumeti veze u lancu snabdevanja.

# 11. ERP sistemi: Prodaja i marketing

Prodaja i marketing uključuju procese na operativnom nivou i procese kontrole upravljanja. Operativni procesi uključuju dnevne aktivnosti, kao što su predviđanje (*prospecting*), upravljanje kontaktima (*contact management*), telemarketing i direktni *mail*. Reprezentanti prodaje treba da kreiraju i održavaju liste predviđanja po lokacijama, po kategoriji proizvoda i prema potencijalnoj prodaji i da održavaju sistem za upravljanje kontaktima. Sistem za upravljanje kontaktima prati karakteristike klijenta, podatke o prethodnoj prodaji i kontaktima. Obično operativne funkcije prodaje i marketinga podržavaju sistemi za obradu porudžbina, koji prikupljaju podatke o porudžbini i POS (*point-of-sale*) sistemi, koji prikupljaju podatke sa mesta prodaje. Ovi sistemi su povezani sa sistemima za upravljanje zalihama koji ažuriraju nivoe zaliha prema podacima o prodaji.

Procesi upravljanja prodajom predstavljaju jednu od najvažnijih oblasti. Menadžeri prodaje su odgovorni za kreiranje teritorija i za raspoređivanje vremena prodajnog osoblja kako bi se generisala maksimalna dobit i usluga. Odluke koje menadžeri prodaje treba da donesu su sledeće:

- Kako bi trebalo urediti teritorije?
- Kako rasporediti vremena prodajnog osoblja?
- Koji klijenti su najprofitabilniji?
- Koji proizvodi su najprofitabilniji?

Informacije koje menadžeri prodaje koriste za donošenje odluka, su uglavnom zasnovani na analizama prethodnih prodaja. Sumarni izveštaji, izveštaji komparativne analize i drugi izveštaji su korisni alati u analiziranju trendova prodaje i određivanju kako najbolje rasporediti resurse. Na primer:

- poređenje prodaja, prihoda po proizvodima, klijentima i regionima u odnosu na benčmarking uspešnosti;
- poređenje produktivnosti svakog prodavca prema proseku za odeljenje;
- izveštavanje o najprofitabilnijim proizvodima po regionima;

- izveštavanje o proizvodima koji predstavljaju najveći procenat prodaje za svakog prodavca;
- izveštavanje o klijentima koji predstavljaju najveći procenat prodaje za svakog prodavca.

Jedan softver za upravljanje prodajom bi trebao da identifikuje slabe proizvode po regionima; da poredi performanse prodavaca po tipu proizvoda i tipu klijenta; da poredi performanse prodavaca u odnosu na ciljeve prodaje; analizira performanse prodavaca unutar regiona; identifikuje trendove kupovine klijenata; identifikuje moguće manjkove i viškove roba na zalihama itd. Analize prodaje bi trebale da pomognu u raspoređivanju prodajnog osoblja, u organizovanju regiona, u tome kako opslužiti potrebe klijenata i kako trenirati prodajno osoblje da efikasnije koristi svoje vreme kako bi se dostigla maksimalna dobit.

Proces predviđanja prodaje je veoma važan kako bi se odredile potencijalne potrebe klijenata u različitim tržišnim segmentima. Aktivnosti predviđanja prodaje uključuju segmentiranje tržišta u ciljne grupe potencijalnih klijenata i planiranje proizvoda/usluga prema potrebama klijenata. Prognoze bi se mogle praviti za celokupnu prodaju ili za prodaje po regionima, prema svakom proizvodu ili uslugama, za prodaju novih proizvoda/usluga i za prodaju prema prodavcima. Predviđanje prodaje koristi informacije iz prošlih prodaja, kao i informacije o konkurenciji, potrebama klijenata i demografskim trendovima.

Drugi važan marketing proces je reklama i promocija. Uglavnom se postavljaju sledeća pitanja:

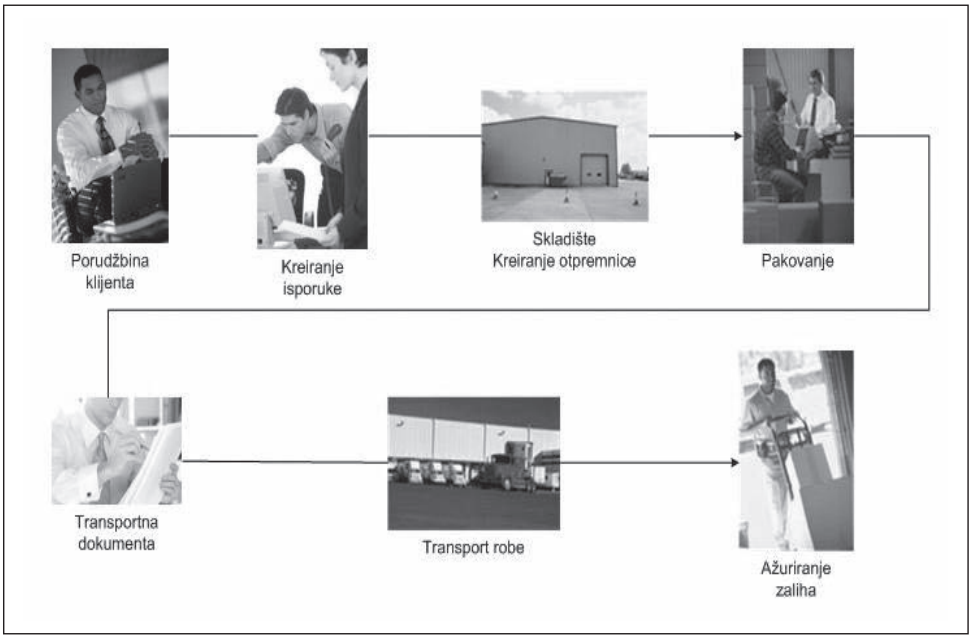
- Koje medije reklamiranja i kanale promocije bi trebalo koristiti?
- Koji kanali i mediji reklamiranja su najefikasniji kada se upućuje ciljnom tržištu?

Ključno pitanje koje se postavlja kod upravljanja marketingom je koju cenu dodeliti prozvodima. Da bi doneo odluku o ceni, marketing menadžer bi trebao da zna očekivanu tražnju za proizvodom, željenu profitnu marginu, proizvodne troškove proizvoda i proizvode konkurencije. Određivanje cena zavisi od strategije, modela određivanja cena koji se prave na osnovu podataka iz različitih izvora koji utiču na određivanje cena, a koje uključuju popis cena klijenta, očekivani raspoloživ prihod klijenta, količinu proizvedenih proizvoda, troškova rada i troškova sirovina.

## 11.1. Moduli prodaje i marketinga u ERP sistemima

Softver za prodaju i marketing obično podržavaju operativne i menadžment procese koji uključuju upravljanje kontaktima, upravljanje prodajama i analize/predviđanja prodaje. Razlika između modula Prodaja i marketing (*Sales and Marketing*) unutar jednog ERP sistema i tradicionalnih softvera za prodaju i marketing je u tome što ERP sistemi obezbeđuju integrisane sisteme za podršku marketingu, koji uključuju kontakt fajlove, fajlove za unos porudžbine, fajlove o bivšim prodajama i dr. ERP sistemi, takođe, obezbeđuju CRM (*Customer Relationship Management*) softver koji opslužuje prodavce sa informacijama o prethodnim iskustvima odnosa sa klijentima, podrazumevajući klijentove nabavke, njegove prioritetne proizvode i istoriju plaćanja.

Unutar jednog ERP modula, klijent postavlja porudžbinu, na osnovu koje sistem pravi raspored isporuke, nabavlja delove i sirovine od dobavljača i pravi plan proizvodnje. Zatim, modul proverava kreditni limit klijenta, ažurira prognoze prodaje i kreira sastavnicu materijala. Takođe se ažuriraju i dužnosti prodavca. Izračunavaju se troškovi proizvoda i profitabilnost. Na kraju se ažuriraju računovodstveni podaci, kao što su bilans stanja i uspeha, računi, glavna knjiga i druge finansijske informacije (Slika 11.1.).

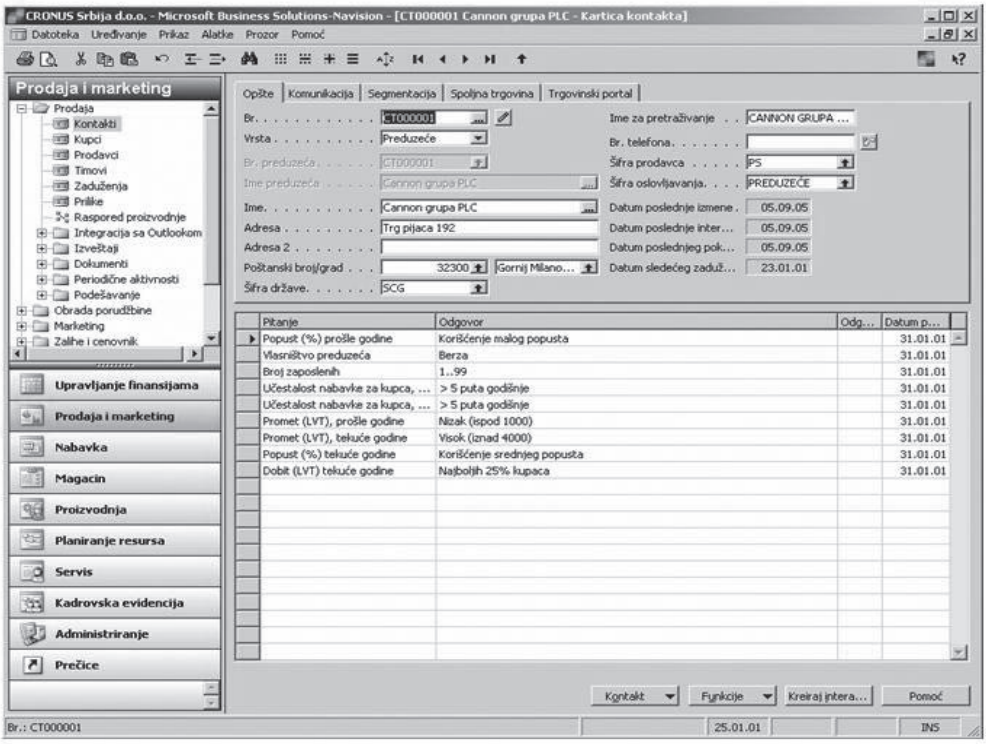



Slika 11.1. Proces prodaje i distribucije

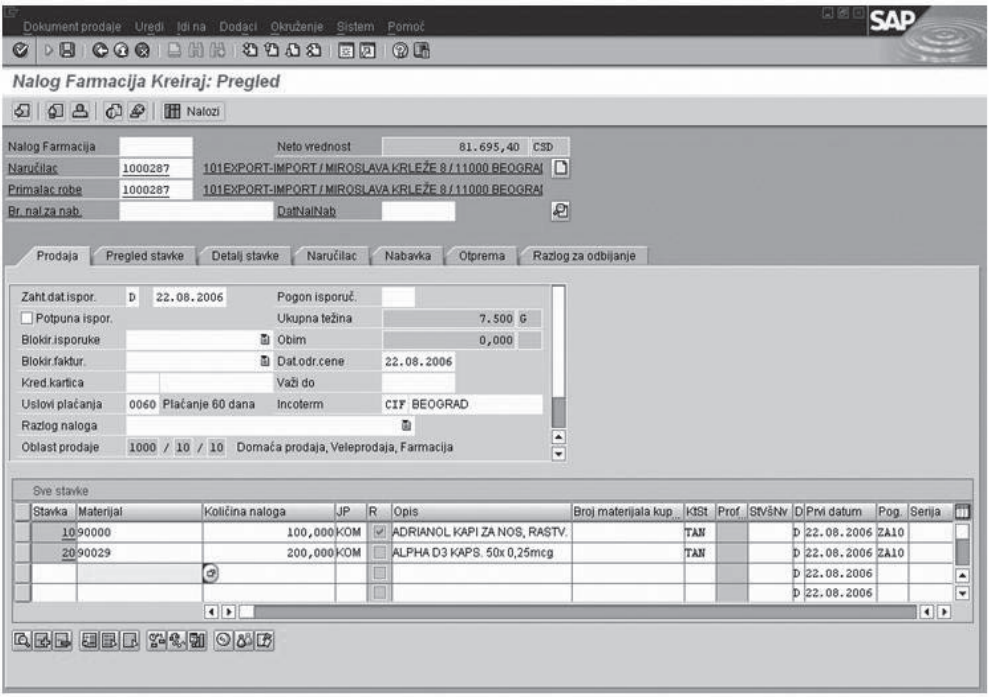
Svrha modula Prodaje i marketinga unutar ERP sistema je da identifikuje očekivane prodaje, obrađuje porudžbine, upravlja zalihama, rukuje isporukama, obezbeđuje naplaćivanje i prihvata i obrađuje uplate i isplate (tabela 11.1). Na narednim slikama mogu se uporediti ekrani *Microsoft Navision* ERP rešenja i SAP R/3 za deo prodaje i marketinga

Tabela 11.1. Funkcije podsistema prodaje i marketinga

| Podsistem | Funkcije |
|---|---|
| Prodajne aktivnosti | Praćenje kontakata klijenata; obezbeđivanje klijenta sa kvotama cena. |
| Obrada porudžbina | Određivanje prodajnih cena, evidentiranje količina poručenih proizvoda; određivanje popusta; proveravanje kredita klijenta. |
| Zalihe | Proveravanje baze podataka zaliha kako bi se uvidelo da li proizvodi mogu da budu isporučeni na vreme; ažuriranje baze podataka planiranja proizvodnje. |
| Isporuka | Proizvodi se biraju; pakuju i isporučuju. |
| Naplaćivanje | Kreiranje fakture; ažuriranje računovodstvenih podataka. |
| Plaćanje | Prihvatanje plaćanja i ažuriranje bilansa stanja. |




Slika 11.2. *Microsoft Navision* ekran




Slika 11.3. SAP R/3 ekran (Kreiranje naloga)

## 11.2. ERP i upravljanje odnosima sa klijentima (CRM)

ERP sistemi podržavaju tzv. *back-office* funkcije kao što su prodaja, računovodstvo, ljudski resursi i proizvodnja, čime obezbeđuju osnovu za napredne aplikacije kao što su CRM i SCM (*Supply Chain Management*). CRM predstavljaju sisteme koji su u interakciji sa klijentima i dobavljačima. Koristeći sledeće scenarije, mogu se uvideti kako CRM sistemi doprinose poslovanju.

Scenario 1: Na putu ste da posetite jednog od najboljih klijenta vaše kompanije, koji čine 5% od ukupne prodaje. Kada ste stigli uvideli ste da je klijent veoma ljut, jer je njegova čitava firma u 48-časovnom prekidu zbog toga što nije dobio isporuku jednog dela za visoko-brzinski laserski štampač u boji kako bi mogao hitno da odštampa neke marketinške materijale. Na neki način, niste prethodno uopšte bili obavešteni o ovom problemu i desilo se to da ste izgubili posao sa svojim klijentom.

**Scenario:**

Na putu ste da posetite jednog od najboljih klijenta vaše kompanije, koji čine 5% od ukupne prodaje. Na putu, uključili ste svoj laptop i primili ste hitnu poruku od klijenta da je došlo do kvara u opremi štampaču u boji. Kada ste stigli, vaša servisna ekipa je već tamo i zamenjuje polomljeni deo. Klijent je bio u mogućnosti da putem *eBusiness*-a naruči odgovarajući deo štampača koga je dostavila kurirska služba. Klijent priznaje da bi verovatno trebalo nadograditi (*upgrade*), odnosno nabaviti štampač većeg kapaciteta jer ga koristi prekomereno. Na ovaj način, ste povećali svoje prihode od svog klijenta.

Ova dva scenarija prikazuju kako znanje o potrebama klijenata može da utiče na kvalitet usluga, a samim tim i na zadržavanje klijenta. CRM softver ima svoje korene u softveru za prodaju koji je projektovan da omogući prodajnom osoblju:

- **upravljanje aktivnostima prodaje** - vodi prodajno osoblje kroz svaki korak procesa prodaje, koji uključuje generisanje prioriteta, kontaktiranje, rukovanje porudžbinama i osiguranje realizacije porudžbine.
- **upravljanje prodajom i teritorijom** – pomaže menadžerima prodaje u nadgledanju aktivnosti prodavaca, optimizovanju tima i praćenju celokupnog procesa prodaje.
- **upravljanje kontaktima** – pomaže prodajnom osoblju u organizaciji podataka o kontaktima, na način da mogu postavljati upite nad bazom podataka i razna pitanja, na primer: „Ko je klijentov agent prodaje?“ ili „Koji klijenti su dobili najnoviju promociju proizvoda X?“.
- **upravljanje prioritetima** – omogućava praćenje atributa proizvoda, kao što su interesovanje za proizvodom, budžet i konkurencija.
- **upravljanje konfiguracijom** – obezbeđuje podršku za određene proizvode.
- **upravljanje znanjem** – omogućava pristup resursima informacija koje uključuju: korporativne priručnike, slajdove prodajnog osoblja, telefone kompanija, šablone za pisanje ponude, podatke o industriji i konkurenciji, isečke iz štampe i zapisnike sa sastanaka prodaje. Ove informacije mogu biti dostupne putem intraneta kompanije.

CRM sistem može biti integrisan sa ERP sistemom. Glavne funkcionalnosti CRM sistema su prikazane na sledećoj tabeli.

Tabela 11.2. Funkcionalnosti CRM sistema

| Ključne funkcionalnosti | Karekteristike |
|---|---|
| Marketing jedan-prema-jedan | Odabir proizvoda, utvrđivanje cena i promocija ka klijentima. |
| Automatizacija prodaje | Olakšava upravljanje kontaktima i listom poziva klijenata. |
| Upravljanje prodajnom kampanjom | Organizacija marketing kampanja. |
| Automatizacija *call* centara | Omogućava postavljanje upita nad marketing bazom podataka proizvoda. |
| Elektronska prodaja (*eSelling*) | *Online* sistemi omogućavaju klijentima da pomoću korpe za kupovinu selektuju željene proizvode. |
| Usluge ka klijentima | Rukuje uslugama klijenata. Inicira se kada klijent zahteva uslugu, pravljenjem *online* porudžbine koju dalje prihvata tehnička služba. Na osnovu potrošenog materijala i provedenih radnih sati računovodstveno odelenje generiše odgovarajuća finansijska dokumenta. |

# 12. ERP sistemi: Računovodstvo i finansije

Donošenje poslovnih odluka o profitabilnosti proizvoda je znatno otežano ukoliko se finansijske informacije održavaju u posebnoj bazi podataka, odvojene od baze podataka nabavke, proizvodnje, marketinga i dr. U osnovi, računovodstveni sistemi podržavaju operativne i kontrolne funkcije. Na operativnom nivou, jedan računovodstveni sistem proizvodi transakcije (fakture, porudžbine i sl.). Proces započinje kada se unese porudžbina koja generiše ažuriranje zaliha kod sistema zaliha koji održava informacije o svakoj stavci na zalihama i povlači nabavku dodatnih zaliha ukoliko nivo zaliha padne do kritične tačke. Sistem porudžbine kreira porudžbine, proverava popunjena polja i određuje vreme kada se očekuje završetak ove transakcije. Porudžbina generiše pravljenje fakture u okviru sistema za obradu računa klijenata. Plaćanje se odvija kod sistema za plaćanje računa.

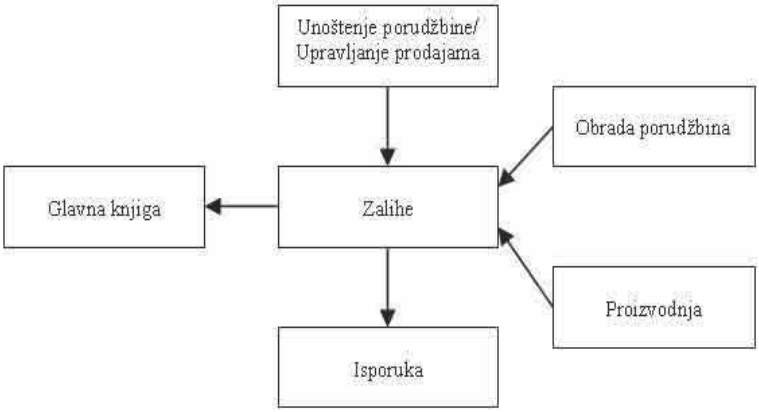
Procesi kontrole unutar računovodstva podrazumevaju budžetiranje, upravljanje gotovinom i investicijama. Budžetiranje podrazumeva poređenje trenutne alokacije budžeta u odnosu na prethodne godine, kao i poređenje trenutnih prihoda i rashoda u odnosu na prethodne godine. Na primer, ukoliko se uporede troškovi putovanja koji su u odnosu na prošlu godinu povećani za 25%, uviđa se znantno povećanje putnih troškova. Druga analiza bi mogla da sagleda da se prihod jednog regiona znatno povećao u odnosu na prethodne godine. Akcije koje bi se na osnovu ovih analiza mogle preuzeti, jesu, na primer, da se u regionu, gde je došlo do povećanja prihoda, ponudi nov proizvod.

Razlika između tradicionalnih računovodstvenih sistema i ERP modula koji podržavaju Računovodstvo i finansije je ta da se finansijske informacije dele u jednoj integrisanoj bazi podataka (Tabela 12.1).

Tabela 12.1. Računovodstvo i finansije u ERP sistemu

| Podsistem | Opis |
|---|---|
| Upravljanje kreditom | Automatski se ažuriraju bilansi računa tako da prodaja uvek ima ažurirane informacije o kreditnim limitima klijenata. |
| Profitabilnost proizvoda | Podaci se unose i skladište u jednoj integrisanoj bazi podataka, koja vodi ka uniformisanim rezultatima. |
| Zalihe gotovih proizvoda | ERP automatski ažurira finansijska dokumenta kada se gotovi proizvodi transportuju do magacina. |
| Troškovi zaliha | Obezbeđuje ažurne informacije o promenljivim troškovima, koje omogućavaju kompaniji da uspostavi cene koje će dovesti do profitabilnijih proizvoda. |
| Konsolidacija informacija | Integrisana baza podataka ima sposobnosti konvertovanja višestrukih valuta. |
| Upravljanje izveštavanjem | ERP baza je integrisana tako da su sve informacije konzistentne, kompletne i tačne. |
| Praćenje | ERP obezbeđuje međusobno povezana dokumenta koja omogućavaju da se lako dođe do izvornog dokumenta. |

Svaka transakcija u sistemu je predstavljena dokumentom koji beleži svaku poslovnu transakciju. Dokumenti su dostupni u realnom vremenu i mogu se pregledati sa bilo kog nivoa detalja. Transakcije koje se odvijaju u sistemu automatski ažuriraju glavnu knjigu (*General Ledger*). Glavna knjiga prikuplja podatke iz čitavog sistema (Slika 12.1).




Slika 12.1. Integracija drugih modula sa glavnom knjigom

Sa ERP rešenjem, menadžeri imaju pristup informacijama o troškovima aktivnosti (*Activity-based cost* – ABC). ABC informacije prate troškove (na primer, ljudi, mašina i drugih resursa) i primenjuje ove troškove na određene proizvode, usluge i klijente. Koristeći ABC sisteme, menadžeri mogu da odgovore na važna pitanja, kao što su na primer: Koliko košta pravljenje određenog proizvoda i koliko je profitabilan? ABC informacije su strateške jer utiču na odlučivanje o marketing i poslovnoj strategiji.

## 13. ERP sistemi: Proizvodnja i upravljanje materijalima

U tabeli 3.1, u trećem poglavlju ove knjige, prikazan je „zapadni" razvoj modela upravljanja proizvodnjom i kasnije kompletnim poslovanjem jedne firme. Paralelno sa razvojem ovog modela, u Japanu se razvijao sasvim drugačiji model upravljanja proizvodnjom/ poslovanjem, zasnovan, pre svega, na najboljoj poslovnoj praksi. Kompletna japanska filozofija je razvijana i usavršavana u tojotinoj (*Toyota*) laboratoriji, čiji su stručnjaci znali da procene šta je to što u praksi daje najbolje rezultate, na primer KANBAN, a takođe i da prihvate svaku novu ideju, bez obzira odakle ona dolazi - TQC (*Total Quality Control – Deming),* koja je u Tojoti doživela svoju prvu primenu.

Svu ovu „mešavinu" okupio je i uobličio Taiči Ono (glavni inženjer u Tojoti) i nazvao je JIT (*Just in Time*)[1]. Iako su MRPII i JIT u početku bile totalno različite filozofije upravljanja poslovanjem, oni su na kraju u ERP rešenjima integrisane. Trenutno je nezamislivo da bilo koji ERP softver nema u sebi KANBAN[2]. Jedan od takvih ERP rešenja je i SAP R/3, koji za upravljanje proizvodnjom ima dva različita modula: SAP PP koji u osnovi služi za diskretnu proizvodnju i koji u sebi sadrži KANBAN (*pull system*) i SAP PP-PI koji je klasično MRP II rešenje (*push system*).

ERP sistemi uvode „najbolje poslovne prakse" koje se jednostavno definišu kao „najbolji način izvođenja procesa". Da bi stekle konkurentnu prednost, organizacije moraju da poboljšavaju njihove poslovne prakse i dele informacije sa svojim poslovnim partnerima i klijentima. Implementiranje ERP rešenja omogućava kompanijama da izvrše reinženjering poslovne prakse ka „najboljoj praksi" i integrišu informacione resurse. Uspeh primene ERP rešenja će zavisiti od efektivnog menadžmenta, organizacionih promena, primene naprednih tehnologija i neophodnog nivoa znanja i veština. Međutim, mnoge kompanije ne iskorišćavaju sve potencijale implementiranog ERP rešenja. Mnogi projekti su propali i usled toga što su vođeni tehnološkim motivacijama umesto poslovnim opravdanjem.

Sa sveukupnog poslovnog gledišta, ERP sistemi dostižu brojne važne ciljeve, kao što su ubrzani protok informacija između svih poslovnih procesa, minimalno vreme odziva na zahteve klijenata i poslovnih partnera, odlučivanje na nižim nivoima, jedinstvene, pouzdane i blagovremene informacije donosiocima odluka, povećane interakcije kroz organizaciju, brže projektovanje i proizvodnju proizvoda, poboljšani menadžment poručivanja, smanjeno vreme ciklusa i troškova poslovnih procesa, bolje performanse rada i dr.

---

1 Pravovremena proizvodnja (Just-in-time - JIT) – proces proizvodnje je snabdeven odgovarajućim materijalima, u pravo vreme, na pravom mestu i u tačnom iznosu. Karakteristike JIT-a su: proizvodnja po narudžbini, u malim serijama, sa nula grešaka, sa najkraćim ciklusom izrade i bez zaliha.

2 Kanban (ident karta ili karta materijala) je sredstvo da se u proizvodnji postigne JIT. Cilj uvođenja Kanban sistema je da se materijal dobro planira, da se prati, kontroliše količina i da se, u pravo vreme, izvrši porudžbina novog materijala za potrebe proizvodnje. Na primer, kada stanje zaliha na skladištu dostigne nivo "momenta porudžbine" (signalna zaliha), odmah se oslobađa sledeći nalog za dopunu određene količine zaliha na skladištu.

Proizvodni sistemi ostvaruju sledeće ciljeve:

- kreiraju planove proizvodnje;
- nabavljaju sirovine;
- raspoređuju opremu, uređaje i radnu snagu;
- projektuju proizvode i usluge;
- proizvode željenu količinu, uz odgovarajući nivo kvaliteta u zahtevano vreme.

Procesi upravljanja proizvodnjom na operativnom nivou (Tabela 13.1) su:

- **Nabavka** – čija je funkcija da nabavi odgovarajuću količinu sirovina i/ili poluproizvoda.
- **Prihvatanje robe** – roba se pregleda i prihvata, a informacije o njenom statusu se prosleđuju do računovodstva, magacina i proizvodnje.
- **Kontrola kvaliteta (*Quality Control* – QC)** – identifikuje vendore koji su isporučili neispravnu robu ili nepouzdane materijale. QC nadgleda kvalitet proizvodnih delova od sirovina, preko proizvoda u proizvodnji do zaliha gotovih proizvoda. Kod TQM okruženja, naglasak je na predviđanje i prevenciju od defekata.
- **Sistemi upravljanja materijalima (*Materials Management*)** - obezbeđuju informacije o nivoima zaliha, o upotrebi tih materijala u proizvodnji i njihovoj lokaciji. Sistemi sastavnice (*Bill of Materials* – BOM) obezbeđuju listu neophodnih sirovina, poluproizvoda i komponenata kako bi se napravio finalni proizvod.
- **Upravljanje zalihama** - održavaju zalihe na odgovarajućem nivou.
- **Sistemi troškova** – prikupljaju i izveštavaju o resursima koji su se koristili u procesu proizvodnje kako bi se utvrdili tačni proizvodni troškovi, koji uključuju troškove osoblja, materijala, opreme i uređaja.

Tabela 13.1. Procesi planiranja i proizvodnje

| | |
|---|---|
| Operativni procesi | Nabavka |
| | Prihvatanje |
| | Kontrola kvaliteta |
| | Računovodstveni troškovi |
| | Menadžment materijala |
| | Menadžment/kontrola zaliha |
| Menadžment kontrola | MRP |
| | Just-in-time (JIT) proizvodnja |
| | Planiranje kapaciteta |
| | Raspoređivanje proizvodnje |
| | Dizajniranje (projektovanje) proizvoda |

Procesi menadžment kontrole (*Management Control*) u proizvodnji su:

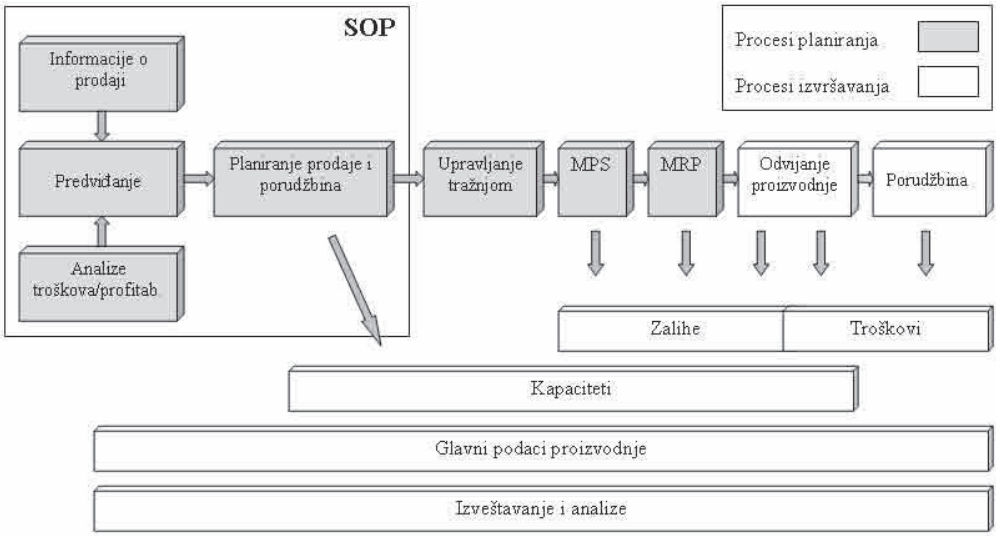
- **MRP (*Material Requirement Planning*)** - podržava procese: identifikovanja zaliha, određivanja glavnih vremena, izračunavanja bezbenosnog nivoa zaliha, određivanja najefektivnijih količina porudžbina i kreiranja porudžbina nabavke za odgovarajući nivo zaliha. MRP koristi ulazne informacije sa Glavnog plana proizvodnje (*Master Production Schedule* – MPS). MPS koristi prognoze prodaje kako bi identifikovao kada su potrebni određeni proizvodi.
- ***Just-in-time* (pravovremena proizvodnja)** – idealno proizvodno okruženje je JIT sistem gde postoji baš onoliko zaliha koliko je i potrebno za proizvodnju. Jedan način za ostvarivanje ovog cilja je uspostavljanje ekstraneta sa dobavljačima kako bi mogli da nadgledaju nivoe zaliha.
- **Planiranje kapaciteta** - određuje da li ima dovoljno proizvodnih kapaciteta (osoblje, prostor, mašine i drugi proizvodni uređaji) kako bi se ostvarili ciljevi proizvodnje. Planiranje kapaciteta zahteva detaljne informacije o ljudskim resursima, BOM, zalihama roba u procesu proizvodnje, zalihama gotovih proizvoda, veličinama serija, statusu sirovina, porudžbina i glavnog vremena[1] za odgovarajuće porudžbine.

  Rezultat je vremenski plan za svaki proizvod i svaku proizvodnu oblast. Planiranje kapaciteta osoblja ocenjuje broj i tip radnika, supervizora i menadžera koji su neophodni da bi se ostvario glavni plan proizvodnje.
- **Raspoređivanje proizvodnje** - raspoređuje određene proizvodne uređaje za proizvodnju gotovih proizvoda prema glavnom rasporedu proizvodnje. Ovo omogućava menadžerima da kontrolišu projekte i vremena završetka projekta i određuju uticaj problema na projektovane datume završetka.
- **Projektovanje i razvoj proizvoda** - integriše informacije o troškovima komponenti u dizajn/inženjering proizvoda kako bi smanjili troškove. Na primer, projektant proizvoda može da poredi troškove komponenata sa dva ili tri alternativna izvora i izgradi komponente koje su troškovno efektivne u dizajn proizvoda pre nego što proizvod krene u proizvodnju.

1 Glavno vreme (Lead time) je vreme za koje dobavljač prihvata i obrađuje porudžbinu i isporučuje je proizvođaču.

Idealni sistem za računarsku integrisanu proizvodnju (*Computer Integrated Manufacturing* – CIM) integriše sav softver i hardver neophodan za proizvodnju sa integrisanom bazom podataka proizvodnje. Neke komponente CIM-a su: projektovanje proizvoda (*Computer-aided Design* – CAD), projektovanje tehnologije (*Computer-aided Manufacturing* – CAM), upravljanje kvalitetom (*Computer-aided Quality Assurance* – CAQ), planiranje proizvodnje (*Computer-aided process planning* – CAPP), upravljanje materijalima (*Materials Management*), logistička podrška i dr. Ovaj koncept eliminiše uska grla u proizvodnji, smanjuje glavna vremena, troškove projektovanja, zalihe u proizvodnji i troškove radne snage i povećava produktivnost projektovanja i inženjeringa.

Sa ERP sistemima, podaci iz prognoze o prodaji ulaze u sistem planiranja prodaje i operacija (*Sales and Operation Planning* – SOP) koji određuje zahtevane resurse proizvodnje, nivo proizvodnje i razvija proizvodne planove (Slika 13.1). Glavni plan proizvodnje (MPS) određuje vremena završetka i neophodne količine gotovih proizvoda. MRP, dalje, razvija detaljan plan materijala i određuje šta treba da se poruči i u koje vreme. Planirane porudžbine se šalju proizvodnji u vidu radnih naloga. Svaki radni nalog uključuje listu zahtevanih materijala sa sastavnice (BOM) i proizvodne operacije koje treba izvršiti. Proces proizvodnje utiče na kapacitet, troškove zaliha, izveštavanje i analize.




Slika 13.1. Planiranje i izvršavanje proizvodnje

Web tehnologije omogućavaju dobavljačima da vide zahteve za ponudama (*Requests for Proposals*), tehničke specifikacije i zahteve za nabavkom, što omogućava brži odziv na zahteve klijenata.

## 13.1. SAP R/3 ERP rešenje

Prikazaće se jedan scenario koji opisuje optimalno rešenje za proizvodnju lekova u jednoj farmaceutskoj kompaniji.

**Scenario 1:**

Optimalno rešenje za farmaceutsku industriju je implementacija sledećih modula:

- Planiranje proizvodnje i kontrola (*Production Planning and Control* – PP-PI)
- Upravljanje kvalitetom (*Quality Management* – QM)
- Menadžment materijala (*Materials Management* – MM)
- Menadžment skladišta (*Warehouse management* – WH)
- Prodaja i distribucija (*Sales and Distribution* – SD)
- Finansijsko računovodstvo (*Financial Accounting* – FA)
- Kontrola (*Controlling* – CO)
- Održavanje pogona (*Plant Maintenance* – PM)

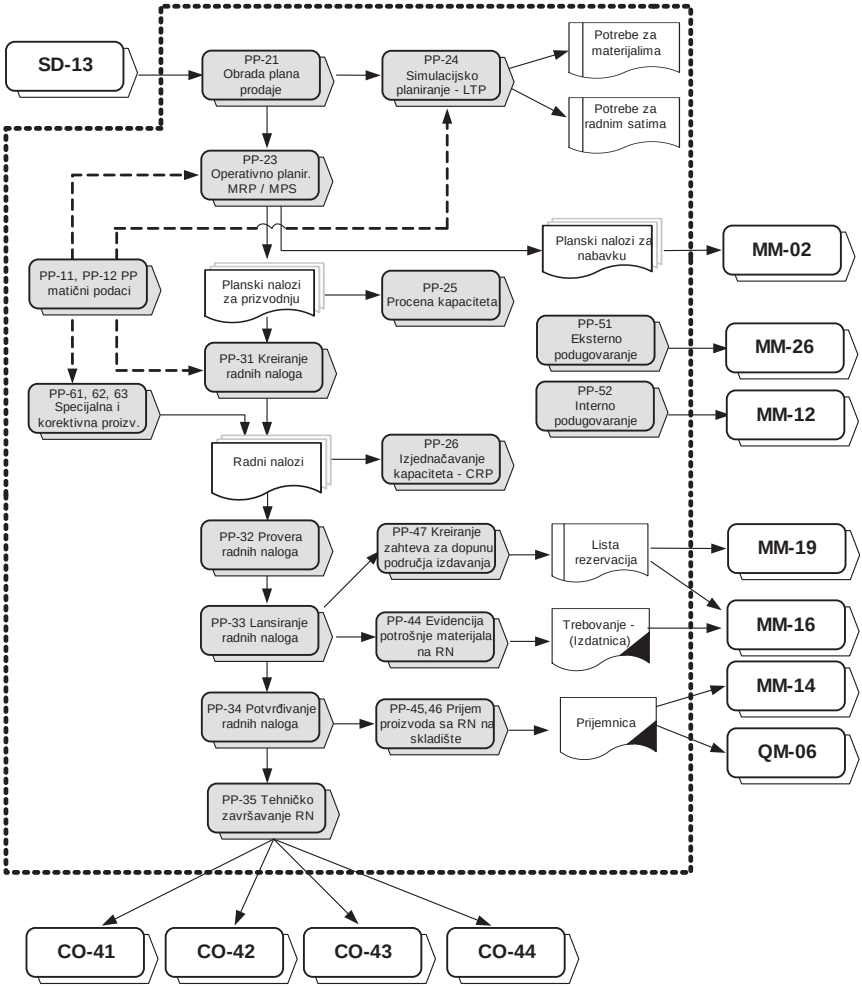
Tip proizvodnje u farmaceutskoj industriji je proizvodnja za zalihe (*make-to-stock*). Za farmaceutsku industriju u proizvodnji je korišćen modul PP-PI, modul upravljanja proizvodnjom za procesnu industriju. Osnovni procesi mogu se ukratko opisati na sledeći način:

- Obradom planova prodaje nastaju planovi proizvodnje.
- Operativnim planiranjem (MRP) nastaju planovi nabavke i planski nalozi za proizvodnju.
- Posle provere svih raspoloživih resursa kreiraju se radni nalozi.
- Nakon izjednačavanja kapaciteta CRP (*Capacity Requirement Planning*) i provere radnih naloga prema prioritetima proizvodnje lansiraju se radni nalozi.
- Lansiranjem radnih naloga kreira se i potrebna dokumentacija za izdavanje sirovina i ambalaže sa logističkih skladišta, odnosno poluproizvoda sa pogonskih skladišta.
- Potvrđivanje radnih naloga inicira praćenje prijema poluproizvoda na pogonska, odnosno gotovih proizvoda na logistička skladišta.

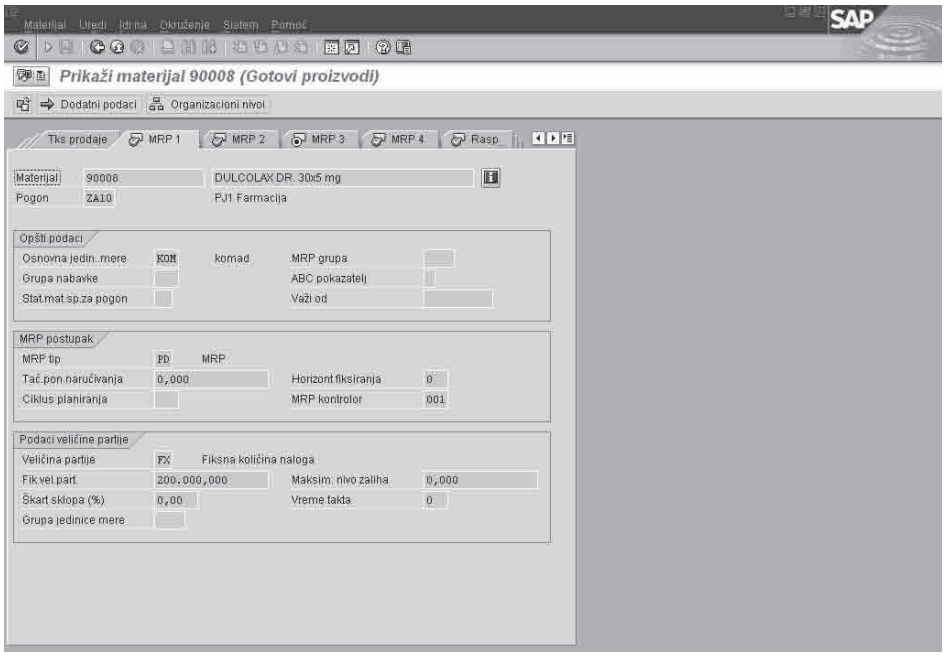
- Tehničko završavanje radnih naloga je završetak ovog uprošćenog modela.

Osnovni procesi u upravljanju proizvodnjom, njihova struktura međuzavisnosti i veze sa drugim modulima (prodaja, kontrola, upravljanje materijalima, kontrola kvaliteta) dati su na slici 13.2.

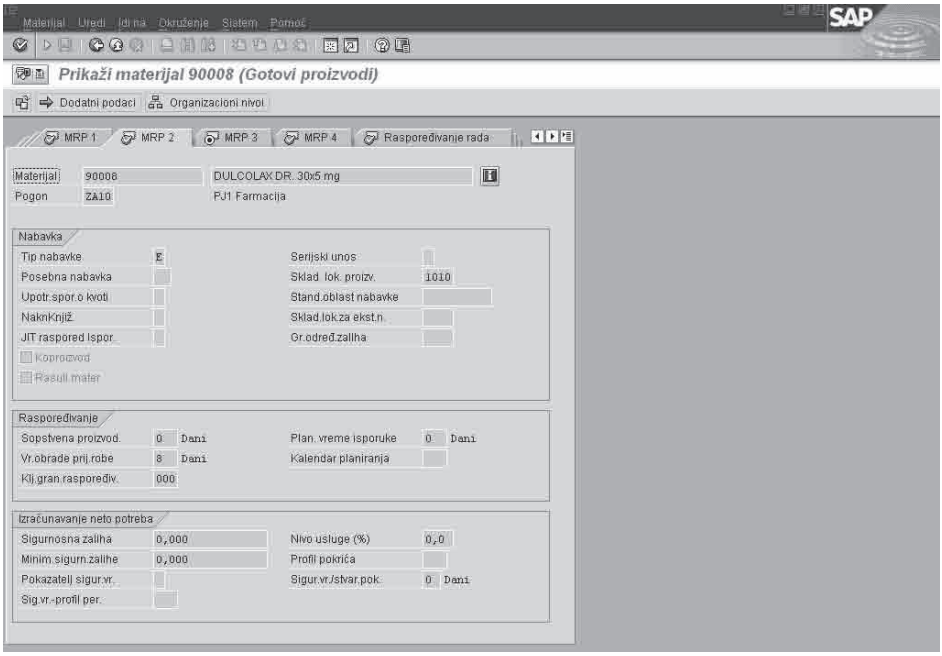
Ekranski prikazi SAP R/3 ERP rešenja za upravljanje proizvodnjom su dati na slikama 13.3, 13.4, 13.5, 13.6 i 13.7.



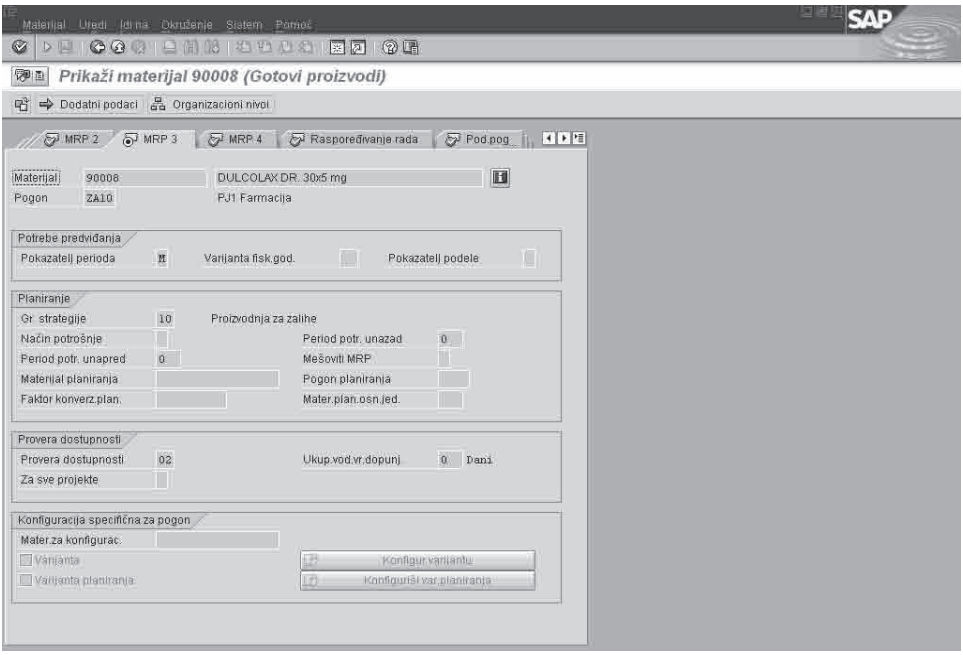

Slika 13.2. Osnovni procesi upravljanja proizvodnjom za farmaceutsku industriju



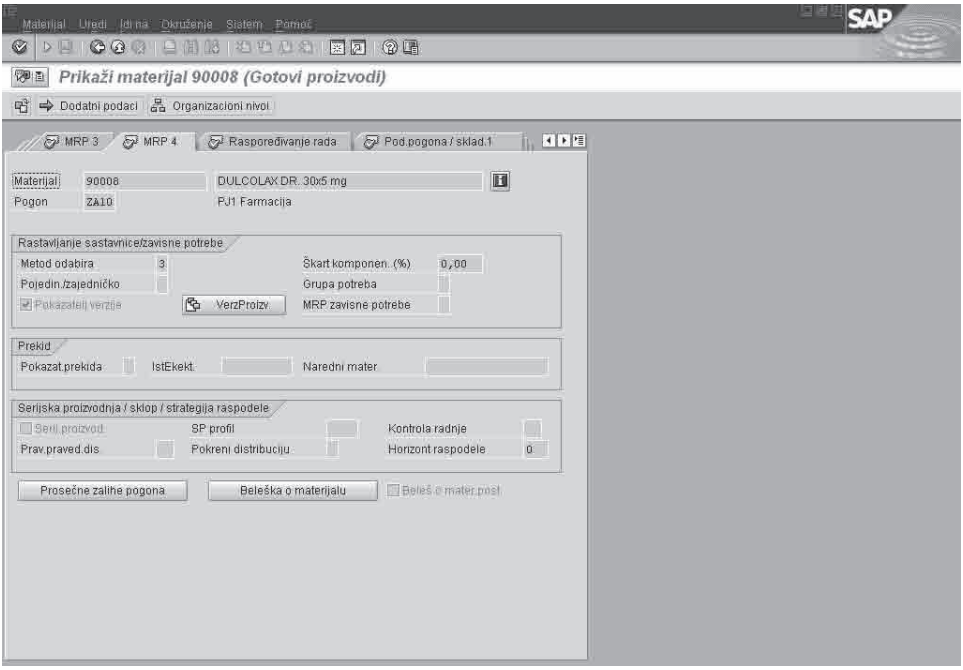

Slika 13.3. Ekranski prikaz MRP1



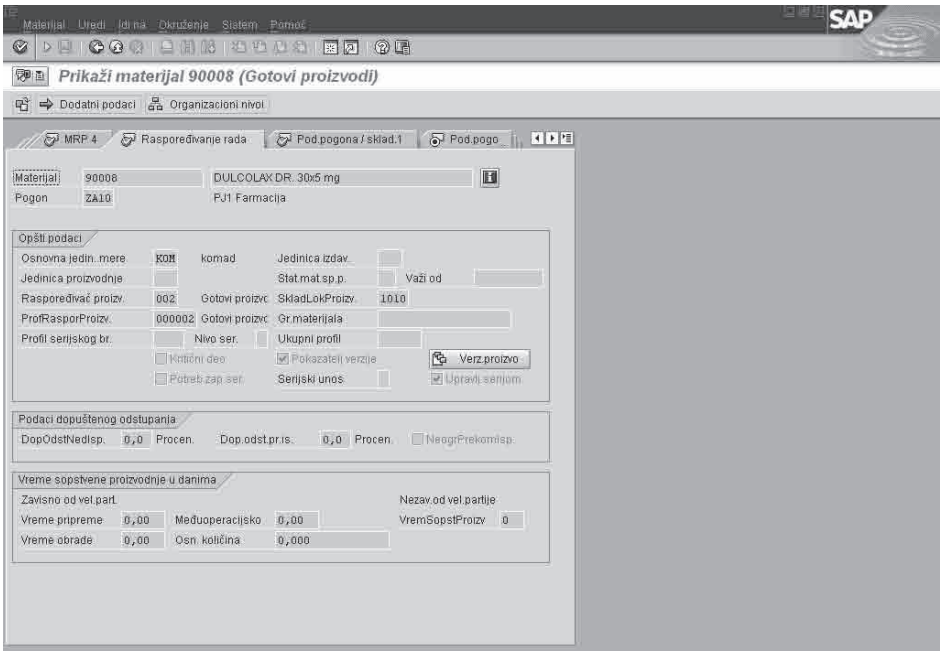

Slika 13.4. Ekranski prikaz MRP2




Slika 13.5. Ekranski prikaz MRP3



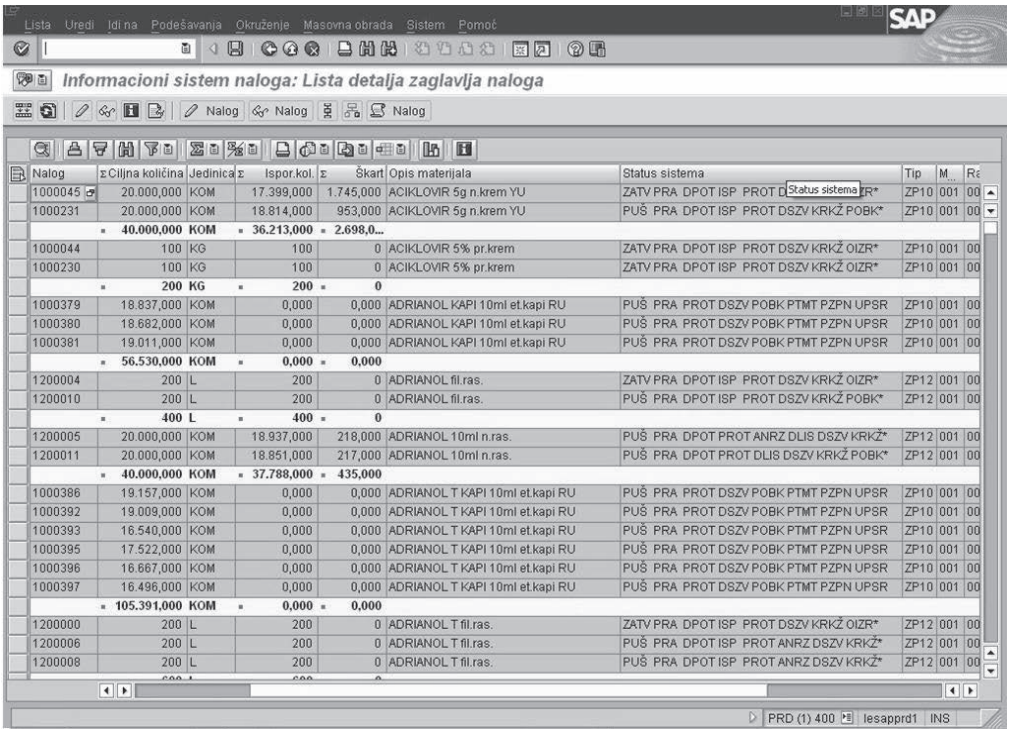

Slika 13.6. Ekranski prikaz MRP4




Slika 13.7. Ekranski prikaz raspoređivanja rada

SAP nudi puno rešenja za farmacetsku industriju koja druga ERP rešenja nemaju, što ga i izdvaja u odnosu na druge. Na primer, proračun aktivnih supstanci u farmaceutskoj industriji. S obzirom da aktivne sirovine veoma retko stižu u istim procentualnim odnosima (što im i samo ime kaže), dolazi do potrebe da se u proizvodnji stalno mora računati receptura za određenu seriju proizvoda iz početka. SAP R/3 i za ovo ima rešenje zato što je moguće definisati varijabilnu sastavnicu koja zavisi od kvaliteta i sadržaja aktivnih supstanci. Za svaku posebnu seriju takve sirovine, sistem automatski proračunava potrebe za svim ostalim komponentama u sastavnici.

Izveštajni sistem R/3 je takođe originalno rešenje. Pored velikog broja predefinisanih izveštaja, R/3 nudi mogućnost da se veoma jednostavno kreiraju izveštaji prema želji i potrebama korisnika bez dodatnog programiranja (Slika 13.8). Međutim, ukoliko ni to ne zadovoljava sve potrebe, R/3 ima snažno razvijene alate u okviru svog *framework*-a.




Slika 13.8: Ekranski prikaz izveštaja naloga [2]

# 14. ERP sistemi: Ljudski resursi

Poslovni procesi koji podržavaju upravljanje ljudskim resursima (*Human Resources* – HR) mogu se kategorizovati u operativne i procese odlučivanja. Procesi na operativnom nivou uključuju kreiranje i održavanje informacija o zaposlenima, njihovim pozicijama i aplikacijama za posao. Druge važne aktivnosti na operativnom nivou uključuju administriranje plata, menadžment performansi, razne potvrde i izveštaje. Na osnovu ovih informacija menadžeri odlučuju kako efektivno rasporediti ljudske resurse. Obučavanje zaposlenih i profesionalni razvoj zaposlenih su kritični za održavanje odgovarajućih veština kako bi se i dalje poboljšavala produktivnost i odžala lojalnost i moral kod radnika.

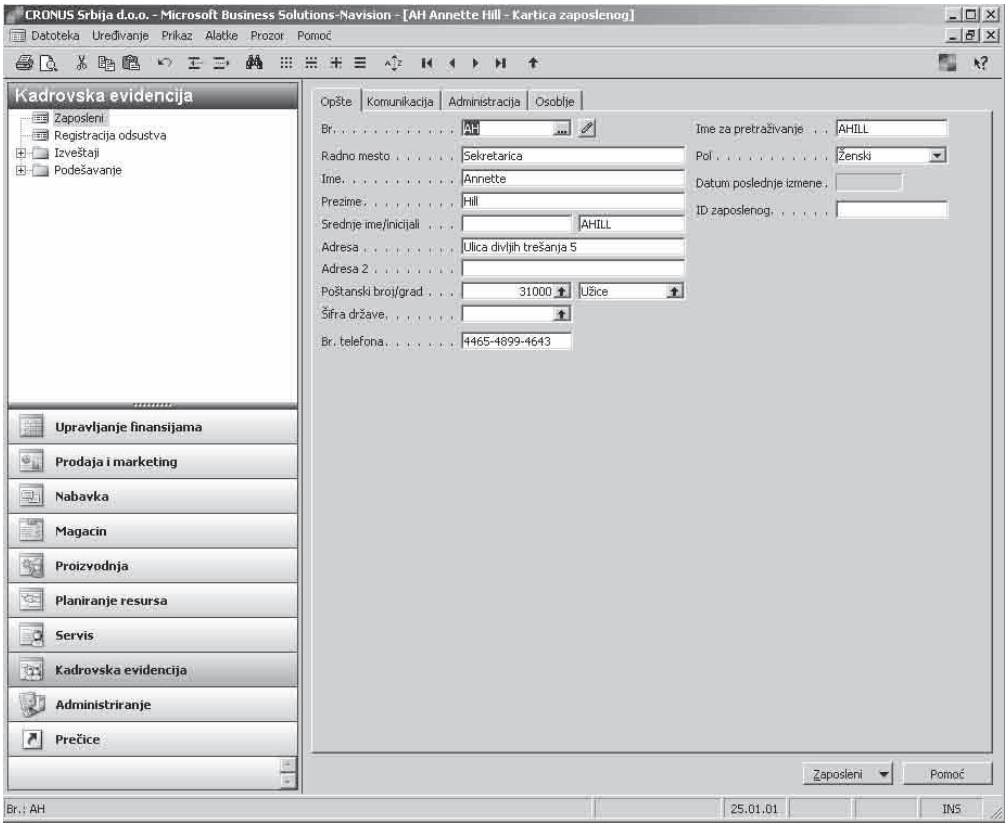
HR modul unutar jednog ERP sistema povezuje HR aplikacije sa finansijskim sistemima. Neke od komponenti HR modula su HR Menadžment, Plate, Menadžment rada i vremena, Web izveštavanje o ličnim i poslovnim podacima i sl. (Tabela 14.1).

Tabela 14.1. HR procesi podržanih ERP sistemima

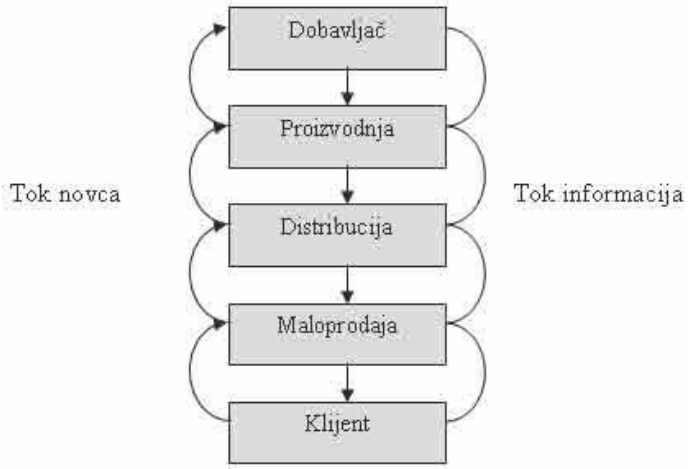
| Podsistem | Opis |
|---|---|
| Informacije o zaposlenima | Održava lične informacije kao što su prethodni poslovi, plata i sl. |
| Veštine | Održava informacije o specifičnim veštinama, radnom iskustvu i sl. |
| Pozicija | Definiše svaku poziciju u organizaciji, obezbeđuje opštu kategoriju poslova, opise i specifikacije. |
| Izbor aplikacija | Održava informacije o regrutovanju, proceni i izboru kandidata. |
| Naknade | Na primer, povećanja plate, istorija plata, procena rezultata posla i dr. |
| Menadžment performansi | Održava podatke o performansama i produktivnosti zaposlenih |
| Sistemi za trening i razvoj | Prati informacije o kursevima, instruktorima, polaznicima, registracijama i sl. |
| Vladini izveštaji | Obuhvata članove o diskriminaciji, o jednakim pravima, bezbednosti na poslu i zdravlju i sl. |
| Plate | Generiše plate i razne iskaze. |

# 15. Menadžment lanca snabdevanja i elektronsko tržište

Tržišna konkurencija, promena potreba klijenata, promene tržišta, kraći životni ciklusi proizvoda i globalna konkurencija karakterišu trenutno poslovno okruženje. Svrha upravljanja lancima snabdevanja (*Supply Chain Management* – SCM) je da se dostigne integrisano planiranje kroz aktivnosti lanca snabdevanja (Slika 15.2). SCM podrazumeva planiranje i upravljanje tokom dobara i usluga, informacija i novca kroz lanac snabdevanja, počevši od nabavljanja sirovina do finalnih proizvoda u rukama klijenata.

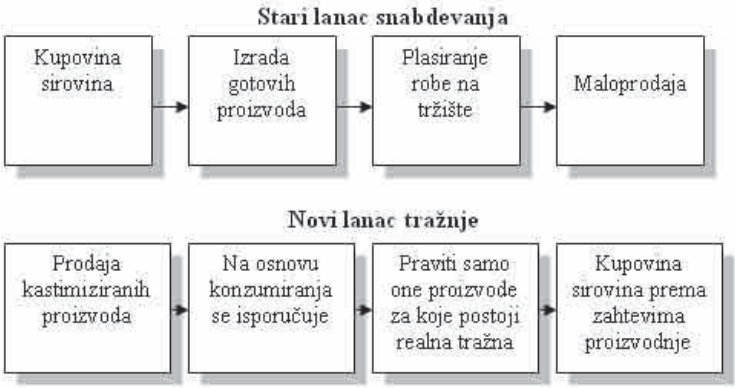



Slika 15.1. SAP ekran (Prikazivanje opštih podataka)



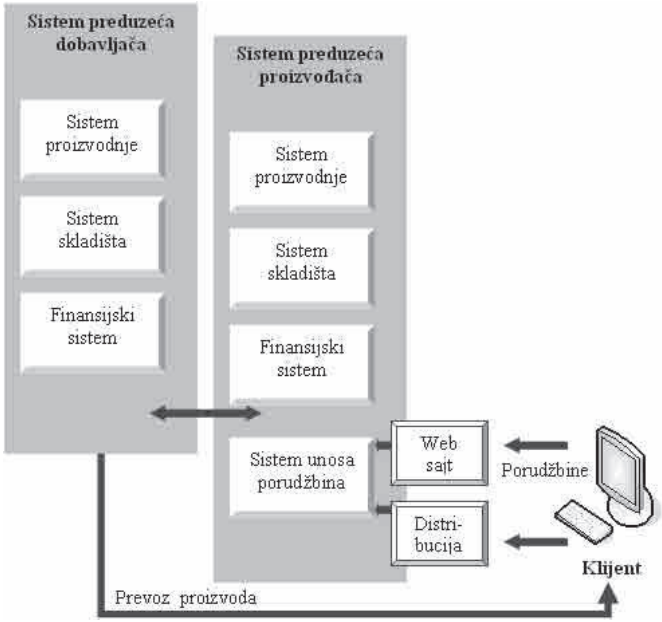

Slika 15.2. Upravljanje lancima snabdevanja

Tranzicija od ranijeg lanca snabdevanja ka novom lancu tražnje je prikazana na slici 15.3.




Slika 15.3. Prelaz sa starog lanca snabdevanja na novi lanac tražnje

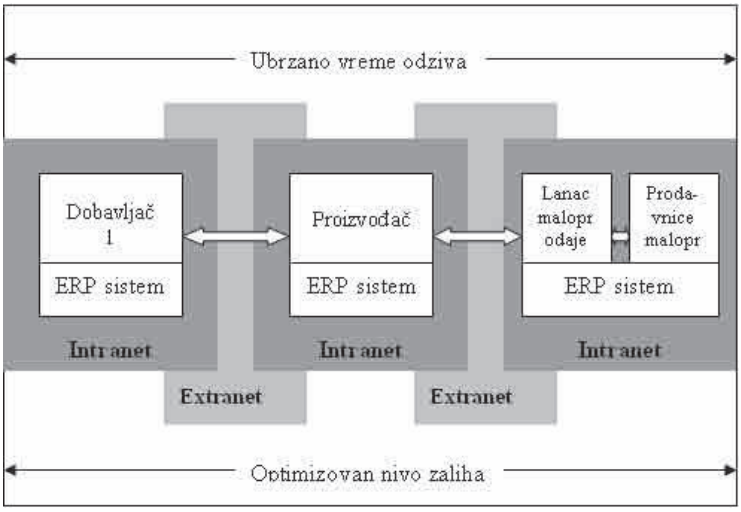
Sistemi upravljanja lancima snabdevanja kontinualno povezuju aktivnosti nabavke, proizvodnje i distribucije proizvoda od dobavljača ka kupcima. Sa jednim SCM sistemom i kontinualnim popunjavanjem dobara na skladištu, eliminišu se zalihe, a proizvodnja započinje tek onda kada se dobije porudžbina (Slika 15.4).




Slika 15.4. Sistemi upravljanja lancima snabdevanja

Da bi se kompanije brzo odazvale zahtevima klijenata, menadžeri moraju da imaju informacije o aktivnostima POS terminala, prognozama potreba, projekcijama rasta, planovima proizvodnje, bilansu zaliha i kretanju proizvoda kroz lanac snabdevanja. Namena takve informacije je da se raspored proizvodnje prilagodi potrebama klijenata.

Kompanija Hewlett Packard je razvila Web zasnovani sistem upravljanja lancom snabdevanja sa sistemom porudžbine koji započinje tako što klijent kreira porudžbinu *on-line*. Porudžbina se iz sistema unosa porudžbine prosleđuje proizvodnji i sistemu isporuke. Odatle se porudžbina prenosi do jednog od nekoliko firmi dobavljača koji verifikuje porudžbinu sa HP-om kako bi osigurala da se proizvod, u ovom slučaju PC može proizvesti. Porudžbina se dalje prosleđuje sistemu kontrole proizvodnje koja izdaje bar kodovane etikete fabrici gde se montira proizvod. Istovremeno, poručeni delovi se prosleđuju do magacina dobavljača i sistemu za upravljanje zalihama. Radnici montiraju i pakuju računar i prevoze ga do klijenta. Isporuka se nagdleda i prati preko HP-ovog sistema za upravljanje lancima snabdevanja, koji se direktno povezuje sa distributerima. Proteklo vreme od unosa porudžbine do isporuke je 48 sati. Sa ovim sistemom, dobavljač i HP su eliminisali zalihe, smanjili vreme ciklusa sa jedne nedelje na 48 sati i umanjili greške. HP je proširio ovaj sistem na globalni B2B sistem za praćenje porudžbina, izveštavanje i podršku. Trenutno njihov sajt radi na 15 jezika u 180 zemalja (Hewlett Packard, 2002).




Slika 15.5. Elektronski lanac snabdevanja

ERP sistemi su osnova za elektronsko tržište (*eMarketplace*). Prednosti elektronskog lanca vrednosti (*eSupply chain*) su smanjena vremena ciklusa, prisnija veza sa partnerima, rast prihoda, smanjenje troškova i vremena proizvodnje, optimizovano upravljanje zalihama, efikasnija distribucija i kolaborativniji proces poslovanja.

## Rezime

1. Moduli prodaje i marketinga u ERP sistemu su projektovani da podrže unošenje porudžbina, generisanje zaliha, obrađivanje isporuka, fakturisanje i obradu plaćanja. Ovaj modul je osnova CRM sistema. CRM moduli podržavaju menadžment prodaje, kontakt menadžment, menadžment vođenja i konfiguracije.
2. Namena modula računovodstvo i finansije je da podrže funkcije finansijskog računovodstva i upravljačkog računovodstva. Informacije o finansijama se kreiraju, održavaju i ažuriraju u svrhe izveštavanja. Informacije upravljačkog računovodstva su interne i projektovane su da podrže donošenje upravljačkih odluka.
3. Moduli upravljanja proizvodnjom i materijalima pokazuju kako je proizvodni plan zasnovan na predviđanju prodaje. Jednom kada je razvijen plan prodaje, menadžmenta tražnje određuje količine i datume koji su neophodni za proizvodnju gotovih proizvoda.
4. Namena SCM (*Supply Chain Management*) je da poveže sve aktivnosti kroz lanac nabavke i to od nabavke sirovina do prodaje proizvoda klijentima. Kod novijih lanaca snabdevanja postoji prisnije partnerstvo između proizvođača i trgovaca na malo koji dele informacije o prodaji i nivoima zaliha. Uvođenje elektronskog poslovanja stvara virtuelne vrednosne lance koji olakšavaju interaktivne odnose i deljenje informacija između dobavljača, proizvođača, prodavaca i klijenata. ERP je osnova za elektronsko poslovanje (*eBusiness*) jer obezbeđuje osnovne podatke za Web transakcije. ERP je takođe osnova za sisteme poslovne inteligencije koji menadžerima pružaju informacije neophodne za donošenje strateških odluka.

## Pitanja za razgovor i vežbe

1. Objasnite razliku između *back office* i *front office*-a.
2. Šta omogućava CRM softver?
3. Da li se više isplati razvijati u kući ili nabaviti gotovo ERP rešenje?
4. Šta se podrazumeva pod terminom *just-in-time* proizvodnja?
5. Kako funkcioniše CIM (računarska integrisana proizvodnja)?
6. Kako ERP sistemi poboljšavaju SCM? Navedite primere za sledeće:
    a. Povezivanje dobavljača i proizvođača
    b. Povezivanje klijenata i proizvođača
    c. Povezivanje proizvođača i prodavaca

## Reference


[1] Bocij. P., D. Chaffey, A. Greasley i S. Hickie, *Business Information Systems – Technology, Development and Management for the e-business*, drugo izdanje, Prentice Hall, Harlow 2003.

[2] Njeguš, A., Z. Đurđević, „Upravljanje proizvodnjom u farmaceutskoj industriji uz pomoć SAP R/3 rešenja“, 13. festival informatičkih dostignuća – INFOFEST, Budva, 2006.

[3] SAP Library,

http://help.sap.com/saphelp_46c/helpdata/en/73/69f5c755bb11d189680000e829fbbd/frameset.htm.

[4] Sumner, M. *Enterprise Resource Planning*, Prentice-Hall, New Jersey, 2005.

[5] Uitgeverij, B., *ERP with Navision – a product of Microsoft Business Solutions*, Microsoft, Amsterdam, 2004.

CIP - Каталогизација у публикацији - Народна библиотека Србије, Београд

007:004(075.8)
004(075.8)

ЊЕГУШ, Ангелина, 1971-
Poslovni informacioni sistemi / Angelina Njeguš. - 9. izd. - Beograd : Univerzitet Singidunum, 2021 (Beograd : Caligraph). - IX, 342 str. : ilustr. ; 24 cm

Tiraž 1.200. - Bibliografija uz svako poglavlje.

ISBN 978-86-7912-582-8

а) Информациони системи б) Информациона технологија

COBISS.SR-ID 45642761

© 2021.

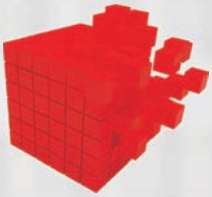
Sva prava zadržana. Nijedan deo ove publikacije ne može biti reprodukovan u bilo kom vidu i putem bilo kog medija, u delovima ili celini bez prethodne pismene saglasnosti izdavača.

9 788679 125828

www.singidunum.ac.rs

Angelina Njeguš

# POSLOVNI INFORMACIONI SISTEMI



BUSINESS INFORMATION SYSTEMS

Informacione tehnologije (IT) ne spadaju više u resurse organizacije, već postaju poslovno okruženje. Informacioni sistemi (IS) formiraju jedan integralni deo moderne organizacije i poslovanja. Informacioni sistemi se koriste kao podrška svim aspektima organizacionih funkcija i aktivnosti. Međutim, koristi od novih tehnologija se mogu postići jedino ukoliko su usmereni ka organizacionim ciljevima. Moderno poslovno okruženje zahteva i novi tip menadžera koji kombinuje menadžerske veštine sa ekspertizom u oblasti IT-a. Takav menadžer treba da:

· definiše strategiju IS bilo za radnu grupu, odeljenje ili kompaniju;

· identifikuje potencijalnu potrebu za IS kako bi poboljšao performanse kompanije;

· izabere i nabavi novi IS od odgovarajućih vendora;

· nadgleda razvoj i implementaciju novih sistema;

· upravlja IS kako bi se obezbedila efektivnost u pružanju kvalitetne informacije krajnjim korisnicima.

Knjiga „Poslovni informacioni sistemi“ treba da posluži kao vodič u odabiru pravih poslovnih informacionih rešenja za organizaciju, njihovom adekvatnom razvoju i efektivnom upravljanju. Knjiga je zamišljena da bude jedini izvor za studente dodiplomskih studija Fakulteta za poslovnu informatiku Univerziteta Singidunum, koji proučavaju poslovne informacione sisteme, bez potrebe da nabavljaju različite knjige koje pokrivaju teme kao što su Sistemska analiza i projektovanje, Strategije razvoja IS, Enterprise Resource Planning (ERP), Analiza zahteva i definisanje arhitekture rešenja, Modeliranje informacionih sistema, Data warehousing, OLAP sistemi, Data Mining i dr.

Univerzitet Singidunum