INFORMACIJSKI SISTEMI

DR. MIRKO VINTAR

2001

(1. POGLAVJE) MESTO IN VLOGA INFORMATIKE V SODOBNI ORGANIZACIJI

Sodobna, računalniško zasnovana informatika naj v organizaciji poleg obravnave podatkov operativnih funkcij na transakcijski ravni, predvsem zagotavlja ustrezne podatke za pridobivanje informacij za podporo odločanju na nadzorni in upravljalni ravni poslovnega sistema . Zadovoljevati mora trenutne in bodoče informacije potrebe uporabnikov.

Opredelitev informacijskega sistema:
Informacijski sistem je skupek ljudi, postopkov in naprav, zasnovan za zbiranje, obdelavo, shranjevanje in distribucijo podatkov oz. informacij.
Informacijski sistem predstavlja organizacijsko-tehnološko okolje za upravljanje z informacijami in informacijskimi tokovi obravnavanega poslovnega sistema.

Vodja informatike na poslovno-organizacijskem področju v organizaciji bo moral uspešno obvladati naslednjo problematiko:
1. strateško načrtovanje informatike organizacije,
2. zagotavljanje konkurenčne prednosti organizacije,
3. prilagajanje organiziranosti novim potrebam,
4. uveljavljanje vloge informatike,
5. uporabniško programiranje,
6. uveljavljanje ugotovitve, da so podatki pomemben dejavnik organizacije,
7. informacijsko infrastrukturo,
8. ugotavljanje učinkovitosti in uspešnosti informatike v organizaciji,
9. povezovanje informacijskih tehnologij in
10. zniževanje stroškov informatike.

Ključni cilj zasnove IS je učinkovitost celotne org. in ne posameznih funkcij in aktivnosti (metodološki pristopi).

Zagotavljanje konkurenčne prednosti
Le natančno in dinamično opredeljene informacijske potrebe vseh nivojev upravljanja organizacije ob ustreznih organizacijskih, finančnih in kadrovskih predpostavkah odpirajo možnost uporabe sodobni inf. tehnologiji v smislu doseganja konkurenčne prednosti organizacije.

Težave zaradi katerih v nekaterih podjetjih niso uspeli zagotoviti prednosti, ki jih nudijo IS so naslednje:
1. težave glede opredeljevanja priložnosti uporabe inf. tehnologije za doseganje konkurenčne prednosti organizacije, ki je odvisna od okolja v katerem se organizacija nahaja, dejavnosti organizacije, značilnosti organizacije, konkurenčne strategije organizacije, vlogo informacijske tehnologije v organizaciji,
2. nepoznavanja procesa uvajanja obravnave uporabe inf. tehnolog. (nejasen in nedorečen proces),
3. nepoznavanje vpliva sprememb, ki so pogojene s sodobno informacijsko tehnologijo,
4. nerazumevanje zmožnosti sodobne inf. tehnolog. s strani vodstvene strukture org.,
5. odsotnost strategije, ki bi opredeljevala priložnosti uporabe inf. tehnolog. za doseganje konk. pred. poslovnega sistema

Načrtnost razvoja informatike (Nri)
Za zagotovitev (Nri) v org. moramo v vsakem okolju najprej opredeliti in zagotoviti razumevanje nekaterih ključnih spoznanj, kot so:
1. vloga in pomen načrtovanja informatike organizacije, podatkovne zasnove, pristop k razvoju in sodelovanje uporabnikov pri razvoju informatike org. ter smotrnost nadaljnje uporabe dosedanjih metodologij gradnje IS ob novih tehnoloških možnostih in dinamiki spreminjanja inf. potreb uporabnikov,
2. postopek ugotavljanja in potreben nivo podrobnosti razčlenitve inf. potreb v fazi izdelave načrta razvoja informatike organizacije,
3. medsebojni vpliv tehnološkega razvoja informacijskih orodij, kompleksnosti informatike in pristopov k njenem razvoju

Uprava in informacijska družba:

1. Inf. tehnologija (IT) kot glavna gonilna sila sprememb v upravi (uvajanje IT v upravo);
2. IT le sredstvo za dosego cilja ali pa vpliva tudi na same cilje uprave (IT vpliva konkretno na storitve, ki nam jih uprava nudi, z internetom bi izboljšali obstoječe storitve in zvišali nivo kvalitet storitev, potencial IT skušamo čim bolj izkoristiti);
3. Spremembe v organiziranosti uprave in v njenih storitvah (organizacijo uprave je potrebno spremeniti tako na mikro kot makro ravni);
4. Spremembe v miselnosti glede narave procesov informatizacije (od avtomatizacije k informatizaciji; celoten poslovni proces neke organizacije podpremo s sodobnimi sredstvi- IT, s čimer omogočimo informiranost subjektov poslovnega sistema)

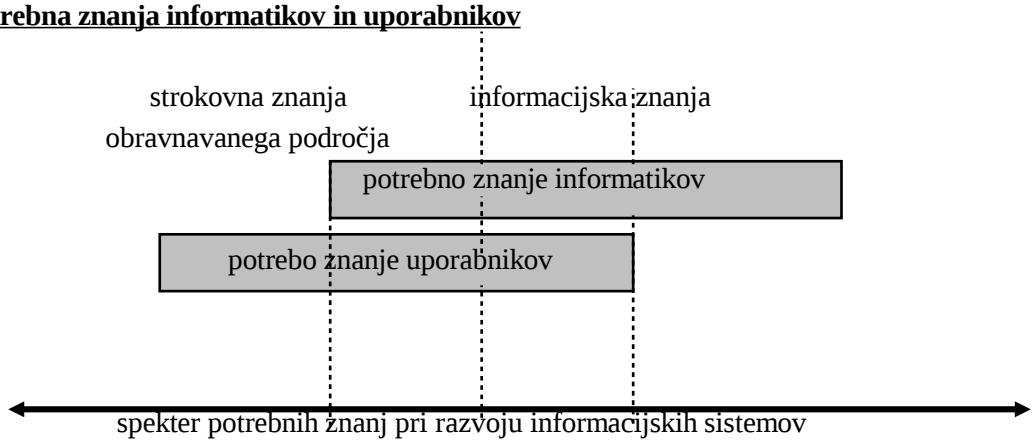
Ključne točke informatizacije uprave (procesa informatizacije organizacij):

1. Uvajanje informacijske tehnologije v vse faze zbiranja, obdelave, shranjevanja in posredovanja inf.;
2. Prenova poslovnih procesov na osnovi inovativne uporabe informacijske tehnologije (ko začnemo graditi IS za novo področje je treba izhajat iz prenove tega področja, prenovljen poslov. proces je izhodišče za nadaljnjo gradnjo);
3. Preureditev inf. tokov ter njihova prilagoditev možnostim informacijske tehnologije (ko bo uprava spremenila svoje poslovanje in v internet vključila segmente sodobne tehnolog. se lahko inf. tokovi med državljani in državo spremenijo), potrebno je bistveno spremeniti način dela;
4. Prilagoditev ali sprememba organizacijske strukture v katero se uvaja sodobna tehnolog. (informatizacija zahteva spremembo inf. strukture, organizacija npr. ministrstev ni več optimalna glede uslug IT);
5. Prilagoditev metod menedžmenta uporabi sodobnih inf. virov (v vodstvih org. oz upravi se še premalo zavedamo pomena inf. virov), danes se vse bolj nadzira tudi uprava, tako da občani lahko kontrolirajo ali uprava uspešno dela;

Vloga informacijskih sistemov v modernih organizacijah:
1. informatika postaja strateškega pomena
2. konkurenčna sposobnost organizacije (ko se v upravi vse funkcije pravilno izvajajo dosežemo konkurenčnost. Pogoj je še cena in hitrost izvajanja)
3. strateško načrtovanje informacijskih sistemov

Ključni dejavniki uspešnega razvoja informacijskih sistemov
1. strateško načrtovanje informatike (virtualno podjetje – nov organizacijski pojav)
2. opredelitev ključnih dejavnikov uspeha in vloge informacijskih sistemov
3. sodelovanje in podpora vodstva
4. sodelovanje uporabnikov
5. projektni tim
6. projektni pristop
7. uporaba informacijskih orodij (programi ki nam pomagajo ob vsakdanjih opravilih in jih moramo znati uporabljati)

Potrebna znanja informatikov in uporabnikov



Informatizacija kot priložnost za prenovo poslovanja uprave

1. od avtomatizacije k informatizaciji
2. vpliv novih tehnologij na poslovanje
3. temeljna izhodišča prenove poslovanja
4. prenova poslovanja na makro ravni (rušimo pristojnosti znotraj uprave)
5. prenova poslovanja na mikro ravni (dogovorimo se o potrebnosti določenega zaporedja procesov)
6. Integracija procesov

Avtomatizacija - ročne aktivnosti so se nadomeščale z avtomatiziranimi postopki.
Informatizacija - celotno poslovanje podpremo z IT, ga prenovimo in maksimalno izkoristimo potenciale, ki jih nudi.

Temeljna izhodišča prenove poslovnih procesov:

1. Pristop po načelu "nepopisanega lista papirja" (k prenovi poslov. procesov je treba pristopiti čimbolj neobremenjeno, pozabiti na to kar že je,neko področje je treba razvijati čisto na novo), ta pristop na žalost ni mogoč;;
2. Preseganje obstoječih organizacijskih struktur in procesna orientacija (pri prenovi poslov. procesov se je potrebno izogibat procesnim strukturam in se osredotočit na delovne procese, ki jih izvajamo);
3. Potreba po radikalni spremembi v pogledu učinkovitosti poslovanja (korenito skrajšanje postopkov, zmanjšanje stroškov poslovanja oz. resna prenova vseh poslov. procesov);
4. Obravnava informacijske tehnologije kot vzvoda in sredstva za spremembo (IT je treba videt kot vzvod radikalne spremembe);
5. Sprememba organizacije in organizacijske kulture kot nujnega spremljevalca sprememb (od avtomatizacije k informatizaciji)

Najpomembnejše razlike med procesom avtomatizacije in informatizacije org.:

Karakteristika pristopa	Avtomatizacija	Informatizacija
Način uvajanja	od spodaj navzgor	od zgoraj navzdol
Vpliv na organizacijo	majhen, predvsem na operativno poslovanje	velik, spreminja same postopke
Potrebna tehnologija	samostojni računalniki, lokalne mreže	lokalne in globalne mreže, internet, intranet
Iniciator sprememb	nižji in srednji management	vrhovni management
Odgovornost za spremembo	nižji in srednji management	vrhovni management
Obseg sprememb	majhne, predvsem v načinu izvajanja, ročna opravila se nadomeščajo z avtomatiziranimi	velike, možna je popolna prenova poslovnih procesov
Baze podatkov	parcialne po poslovnih funkcijah	integrirane za celotno org.
Upravljanje informacijskih virov	decentralizirano po organizacijskih enotah	decentalizirano ali centralizirano
Vloga inf. tehnologije v organizaciji	vpliv je čutiti predvsem na operativni in tehnični ravni	IT pridobiva strateško vlogo, vse vitalne funkcije organizacije so odvisne od uporabe IT
Spremembe v organizacijski strukturi	običajno jih ni	tudi zelo velike, odvisno od narave organizacije in njenega vodstva
spremembe v normativni ureditvi	niso nujne	koristne, včasih celo pogoj za uspeh projektov informatizacije
Vpliv na management	delen	Velik

(2. POGLAVJE) METODOLOGIJE NAČRTOVANJA IN GRADNJE INFORMACIJSKIH SISTEMOV

2.1.OPREDELITEV OSNOVNIH POJMOV

METODA- postopek ali tehnika za izvedbo posameznega segmenta ali faze razvojnega cikla IS. **METODOLOGIJA** je po definiciji skupek postopkov, tehnik, metod, ki jih uporabljamo pri reševanju nekega problema.
Metodologija gradnje informacijskih sistemov pod tem pojmom si vsaj v praksi največkrat predstavljamo organizacijsko–tehnično znanje, ki ga uporabljamo pri zasnovi in izdelavi računalniških rešitev.

Na področju **metodologije projektiranja in gradnje informacijskih sistemov** še ne poznamo neke urejene formalne teorije, ki bi bila osnova za razvoj postopkov in metod projektiranja in gradnje IS. Doslej razvita teorija nam omogoča formalizacijo nekaterih segmentov razvojnega procesa IS, ne pa celote.
Razlogi za takšno stanje:

1. disciplina je zelo mlada (prva poročila o raziskavi v literaturi sredi šestdesetih let),
2. IS sodijo v kategorijo izredno kompleksnih sistemov,
3. IS sodijo v kategorijo konceptov, modelov oz. teorije predstavitve znanja (problem predstavitve modela)

Glavni elementi celovite metodologije:

- opredelitev ključnih razvojnih faz ter njihovega sosledja,
- vsebinski opis vsake faze z opredelitvijo ključnih aktivnosti,
- navodila za izvedbo aktivnosti,
- prikaz metod in tehnik za izvedbo posameznih aktivnosti,
- opredelitev zahtevanih rezultatov posamezne faze,
- opredelitev kriterijev za kritično ovrednotenje rezultatov posameznih faz (zato da vemo ali smo dosegli cilj ali ne),
- navodila glede organizacijskih kadrovskih ter tehničnih pogojev pri uporabi metodologije,
- opredelitev področja uporabnosti

Ključni problemi razvoja programskih rešitev (oz. informacijskih sistemov):

1. predolgi razvojni cikli IS,
2. nizka produktivnost zaposlenih, zato veliki stroški
3. nepredvidljiva kvaliteta končnega programskega produkta,
4. izredno visoki razvojni in vzdrževalni stroški

Posledica vsega tega pa so visoke cene programske opreme, ki strukturi že predstavlja tudi do 80% stroškov vse opreme IS.

Razvoj konceptov modeliranja podatkov:

1. generacija: Primitivni modeli podatkov:
   - entitete so predstavljene z zapisi, ki se grupirajo v datoteke;
   - zaporedna, naključna, indeksno-zaporedna organizacija datotek,
   - povezave med datotekami

2. generacija: Klasični modeli podatkov:
   - hierarhični model,
   - mrežni model,
   - relacijski model

3. generacija: Semantični modeli:
   - model entiteta **–** povezava (E – R),
   - binarni model,
   - matematični model,
   - infološki model

4. generacija: Objektivno orientirani modeli:
   - v razvoju

2.4. VLOGA MODELIRANJA PRI NAČRTOVANJU IN GRADNJI IS

MODELIRANJE uporabljamo pri raziskovanju ali reševanju problemov na različnih področjih. Gre za prenos lastnosti, značilnosti raziskovanega predmeta na podoben predmet, narejen po določenih pravilih.
MODEL je vedno lahko le preslikava naših predstav o stvarnosti, ne pa stvarnosti same, kar povečuje možnosti za odstopanje med originalom in modelom. Model je vedno v nekem odnosu s svojim originalom. Med njima obstaja analogija in podobnost, kar se odraža v enaki strukturi, funkciji, obnašanju ali v vseh treh lastnostih.

MODEL ima dve nalogi:

1. da nam omogoči boljšo predstavitev, opredelitev in s tem razumevanje obravnavanega problema;

2. da poveča možnost predvidevanja

2.5. OPREDELITEV POMEMBNEJŠIH PRISTOPOV

Proces nastajanja in delovanja informacijskega sistema običajno poimenujemo **življenjski ciklus informacijskega sistema.** Ta ciklus zajema vse faze, od analize, načrtovanja, gradnje do izkoriščanja in spreminjanja informacijskega sistema. Ta ciklus je odvisen od metodologije razvijanja sistema

RAZVOJNI CIKEL IS: (v praksi si te faze ne sledijo tako lepo linearno)
1. definicija problema (opredelitev osnovne ideje in ciljev, ki bi jih z njo uresničili);
2. analiza in opredelitev zahtev (ugotovimo informacijske zahteve, kaj sistem potrebuje da bo učinkovito deloval), katere podatke potrebujemo in kateri postopki te podatke spreminjajo;
3. zasnova (naredimo načrt IS v tehničnem smislu);
4. gradnja (ni ostro postavljene meje med načrtovanjem in gradnjo. Ko se začne programiranje naj bi se začela faza gradnje), gradnja je v domeni informatikov;
5. uvedba; (poskrbimo za vse potrebno da bo IS zaživel);
6. preverjanje rešitev (izvajalci drugače naredijo, kot pa je naročnik želel, zato moramo rešitve preverit);
7. **vzdrževanje**

Karakteristični cikli IS (so odraz različnih pristopov oz. uporabljenih metodologij):

- linearni pristop,
- prototipni pristop,
- objektni pristop

LINEARNI PRISTOP (se je najprej razvil)
- zaporedni fazni proces;
- nobena faza se ne more začeti dokler ni v popolnosti dokončana predhodna faza;
- po vsaki fazi se izdela poročilo, ki služi kot osnova za začetek naslednje faze

namen poročil:
   - naročniku služi kot dokazilo o opravljenem delu,
   - vodji projekta kot osnova za planiranje dela,
   - projektnemu timu kot osnova za vpogled v posamezne detajle projekta

Značilnosti linearnega pristopa:

1. razvoj IS poteka po »kaskadnem principu« skozi natančno določene faze, ki si sukcesivno sledijo,
2. vsaka razvojna faza je natančno definirana in podrobno dokumentirana (zaradi vzdrževanja),
3. modelni mehanizmi (metode in tehnike) so, če lahko o njih sploh govorimo razmeroma preprosti in temeljijo na analitskih in dokumentacijskih tehnikah ter večinoma verbalnih opisih obravnavanega problema,
4. iz začetne analitične faze obravnavanega problema se praviloma izvrši neposreden prehod v izdelavo izvedbenega modela IS (detajlne tehnične rešitve), ki

je praviloma nenatančna in zato vir mnogih napak
5. univerzalen za vsako okolje, neodvisen od velikosti problema in neodvisen od uporabljenih orodij
6. dolgi in dragi cikli, slabo sodelovanje uporabnikov, nepredvidljiv konec

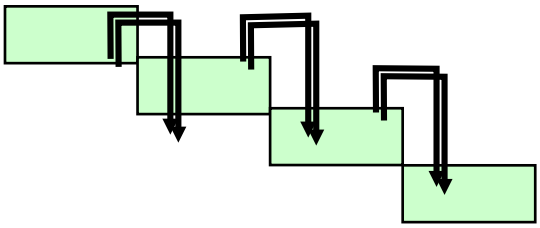
Prednosti: - omogoča dober pregled nad stanjem posameznega projekta, standardizacijo postopkov in dokumentacijo, nadzor nad sodelavci v projektnem timu

Slabosti:

1. v praksi se pokaže, da posameznih razvojnih faz ni mogoče tako natančno opredeliti,
2. niso mogoči čisti prehodi iz predhodne v naslednjo fazo,
3. izkaže se lahko, da predhodna faza ni bila opravljena dovolj temeljito ali pa da iz njene dokumentacije ni mogoče zanesljivo načrtovati posameznih segmentov sistema
4. to zahteva vračanje nazaj v predhodne faze, spreminjanje že narejenega in dokumentiranega, kar povečuje stroške in podaljšuje čas za realizacijo projekta

Vse to vodi k naslednjim slabostim:
1. predolgi razvojni cikli,
2. visoki razvojni stroški,
3. napake in pomanjkljivosti se odkrijejo šele na koncu,
4. sodelovanje uporabnikov je otežkočeno ali nemogoče.

kaskade



PROTOTIPNI PRISTOP
S prototipom se skuša ugotoviti možnost izdelave novega proizvoda, njegove lastnosti, proizvodne stroške, z njegovo pomočjo se izdela končni proizvod, sam prototip pa se zavrže.

Takšen način uporabe prototipa zasledimo pri linearnem pristopu gradnje IS :

1. preden izdelamo celoten sistem, s preprostim prototipom uporabnikom oz. naročniku pokažemo osnove funkcije sistema, vhode, izhode, pri tem zanemarimo detajle, različne kontrole;
2. za izdelavo prototipa lahko uporabimo druga orodja, jezike kot za končno rešitev,
3. pod prototipnim pristopom razumemo neke vrste evolutivni pristop
4. v tesni interakciji z uporabniki skušamo izdelati prototip sistema, ki ga potem dopolnjujemo in spreminjamo
5. postopno razvijanje v končni proizvod;
6. zahteva skrbno načrtovanje informatike na strateški ravni in podrobno opredelitev karakteristik IS na logični ravni;
7. izhodišče za razvoj IS pri prototipnem pristopu je podrobno opredeljen logični model IS

Prednosti prototipnega pristopa:

1. nastal je kot odgovor na slabosti linearnega pristopa z namenom skrajšati in poceniti cikel
2. S prototipnim pristopom se skuša doseči:
   - skrajšati čas, ki je potreben, da pridemo do prvih rezultatov;
   - omogočiti kreativno sodelovanje uporabnikov skozi celotno razvojno obdobje projekta;
   - zagotoviti, da se morebitne napake in pomanjkljivosti v projektu pokažejo v zgodnjih fazah kar dosežemo z vlogo uporabnikov

Slabosti:
1. zanašamo se na intuicijo;
2. ne sili v sistemiziranost in vodenje dokumentacije o opravljenem delu;
3. ni jasno določenih faz;
4. na koncu nimamo dobre dokumentacije, ki bi omogočala vzdrževanje rešitev.
5. priporočljiv je le za manjše sisteme

OBJEKTNI PRISTOP (napovedujejo se že 10 let pa še vedno niso množično uporabni)
Naj bi bil odgovor na slabosti predhodnih dveh. Razlikuje se od predhodnjih, da enovito obravnava podatke in postopke. Ta pristop ugotavlja da je informacijski sistem preslikava objektov, ki jih imamo v realnem svetu.

Objektni pristop temelji na 3 temeljnih konceptih:
1. objektih - vsebujejo podatkovne strukture in pripadajoče postopke na teh strukturah; objekti so največkrat objekti, ki nastopajo v realnem svetu (produkti, kupci..), pri kovencionalnem pristopu jih obravnavamo kot entitete;
2. sporočilih, t.j. sredstvu, s pomočjo katerega objekti komunicirajo med seboj pri izvajanju poslovnih postopkov;
3. tipih objektov - omogočajo realizacijo konceptov abstrakcije, ki jih poznamo iz podatkovnih modelov, kot so klasifikacija in generalizacija

Glavni cilji objektnega pristopa:
1. želja, da bi programsko opremo lahko razvijali podobno kot strojno iz množice standardiziranih sestavnih delov, kjer je vsak sestavni del po sistemu lego kock lahko uporabljiv večkrat, v različne namene,
2. elementarni objekti se lahko sestavljajo v komplementarne objekte po pravilih generalizacije to vodi v hierarhijo objektov;
3. objekt na najvišjem nivoju predstavlja celotno programsko rešitev

Prednosti objektnega pristopa:

1. večkratna uporaba istih objektov, kar bo znatno skrajšalo čas za razvoj novih rešitev ter zmanjšalo stroške.
2. ker uporabljamo standardizirane objekte, jih lahko večkrat uporabimo (objekte) zaradi česar se zmanjšajo stroški in skrajša se razvojni čas;
3. ker uporabljamo standardizirane objekte je tudi zanesljivost večja
4. ker so objekti povsem zaključene celote, se individualni objekti lhko spreminjajo, ne de bi to kakor koli zahtevalo spremembe v drugih objektih, kar zelo poenostavi vzdrževanje programskih rešitev
5. nove rešitve bodo sestavljene iz že obstoječih in preizkušenih gradbenih blokov (objektov) - to prispeva k zanesljivosti in kakovosti rešitev;

2.6. OSNOVNE ZNAČILNOSTI SODOBNE METODOLOGIJE

Na razvoj in uporabnost metodologij ima danes vpliv cela vrsta dejavnikov, kot so:

1. naglo naraščanje procesne moči vseh vrst računalnikov,

2. integracija poslovnih procesov ter poslovnih informacijskih sistemov,
3. distribuirano procesiranje in razvoj računalniških mrež,
4. bogata ponudba standardnih aplikativnih rešitev,
5. razvoj in naglo uveljavljanje računalniških orodij za razvoj in projektiranje IS

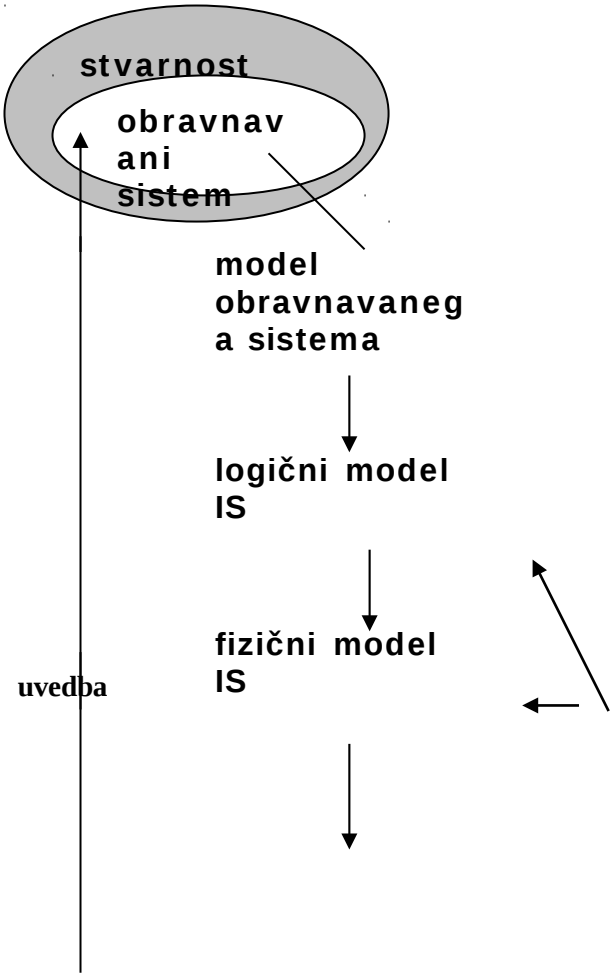
Karakteristike sodobne metodologije:

1. Zajemati mora celotni življenjski cikel IS in pripadajoče programske opreme, ne samo posameznih faz.,
2. Omogočati mora sistematični prehod iz ene v naslednjo fazo. Dana mora biti možnost spreminjanja vseh razvojnih faz in po potrebi vračanja nazaj in preverjanja predhodnih faz IS.,
3. Omogočati mora verifikacijo pravilnosti razvojnega cikla IS. Do odstopanj lahko pride na več relacijah. Rešitev se lahko ne ujema s samo specifikacijo IS, lahko se ne ujema z uporabnikovimi potrebami, lahko pa je že specifikacija napačna. Metodologija bi morala vsebovati neke formalne tehnike za verifikacijo vseh razvojnih faz IS.;
4. Omogočati mora timsko delo na projektu ter uporabo sodobnih metod organizacije in vodenja projektov. To pomeni, da mora omogočati dobro komunikacijo med vsemi sodelavci tima in sprotno verifikacijo opravljenega dela. ;
5. Biti mora uporabna za čim širši spekter računalniških projektov, kar omogoča v okviru organizacije, ki jo uporablja, ustrezno standardizacijo, razvoj pomožnih orodij, dokumentacije...;
6. Biti mora dovolj enostavna za priučitev - da naj bi jo bili sposobni razumeti in uporabljati povprečno sposobni ljudje, ki sodelujejo pri realizaciji posameznih projektov. ;
7. Omogoča naj uporabo čim širšega spektra avtomatiziranih orodij za povečanje produktivnosti posameznikov in celotnega tima.;
8. Omogočati mora dokumentiranje spremljanje razvoja IS skozi vso njegovo življenjsko dobo.

2.7. OPREDELITEV ZNAČILNIH RAZVOJNIH FAZ

Pod načrtovanjem Informatike razumemo načrtovanje celotne ali globalne informacijske infrastrukture v obravnavani organizaciji, kar je predpogoj za uspešen nadaljnji razvoj informacijskih sistemov posameznih poslovnih funkcij.

RAZVOJNE FAZE IN MODELI IS



Faza 1: Strateško načrtovanje informacijskega sistema .
Analiza OS in izdelava modela OS ter ugotavljanje informacijskih potreb

Faza 2: Logična opredelitev IS
Preslikava modela OS v formalni opis IS – izdelava logičnega modela IS; modeliranje podatkov in postopkov

Faza 3: Fizična opredelitev IS
Razvoj baze podatkov programskih modulov,...

Ta sistema ločimo zato, da se pri logičnem modelu ločimo od tehnologije, ki jo bomo uporabili pri sami izvedbi, to pa zato ker se tehnologija hitro spreminja.

izvedbeni model IS

Faza 4: Gradnja IS
Programiranje, testiranje, preverjanje

1. faza: v prvi fazi gre za **strateško načrtovanje informatike oz. IS,** ki mora biti utemeljeno na modelu obravnavanega sistema, ki prikazuje glavne poslovne funkcije sistema ter njihove informacijske potrebe. Model obravnavanega sistema prikazuje poslovne funkcije sistema ter njihove informacijske potrebe. Ta model predstavlja vse naše znanje o konkretni organizaciji. Prva faza je strateški načrt (ugotovimo potrebe, opredelimo funkcije ki se izvajajo v obravnavanem sistemu) in rezultat prve faze je model obravnavanega sistema. Obravnavani sistem je del realnega sveta, ki ga obravnavamo. Običajno se lotevamo le posameznih funkcij in ne celih organizacij, ker je to preveliko in prezahtevno.

Gre za:
- organiziranost obravnavanega sistema;
- funkcijo OS;
- analiza informacijskih tokov: (ključen vhoden podatek, hočemo izboljšat inf. tokove);
- analiza procesov in postopkov;
- opredelitev informacijskih potreb (kaj vse neko področje potrebuje, da bo lahko funkcioniralo);
- strateški načrt informatizacije (na tej osnovi izdelamo celovit načrt informatizacije OS).

2. faza: **opredelitev IS na logični ravni,** katere rezultat je logični model IS (nastaja postopoma) - gre za vsebinsko opredelitev IS, pri kateri zanemarimo vse elemente njegove konkretne tehnične izvedbe. Rešitev na logični ravni naj bo v čimvečji meri neodvisna od uporabljene tehnike oz. informacijske tehnologije. Osnova za drugo fazo je model obravnavanega sistema, rezultat druge faze pa je logični model IS
Značilnosti logičnega modela**:**
Preslikamo model obravnavanega sistema v logični model, pri tem nam pri klasičnem postopku nastaneta podatkovni in postopkovni model. Logični model se sestoji iz podatkovnega in postopkovnega modela. Logični model opredeljuje vse podatkovne in postopkovne značilnosti načrtovanega informacijskega sistema, s tem da se zavestno ne spuščamo v to kakšna bo fizična izvedba tega sistema.

3. faza: razvoj fizičnega modela IS, kjer že upoštevamo vse karakteristike ter zahteve in omejitve izbrane programske ter strojne opreme; V praksi bi lahko fizični model združili z logičnim.

Logičen model je neodvisen od uporabne tehnologije, fizični model pa je dejansko izpopolnjen logični model ob upoštevanju izvedbenih elementov, ki so vezani na izbrano strojno programsko opremo. V tej fazi torej opredelimo potrebno strojno in programsko opremo.

4. faza: **izvedbeni model IS,** ki ga predstavlja končna rešitev obravnavanega IS (programiranje, testiranje, preverjanje).

Kvaliteta načrtovanega IS je v največji meri odvisna od razumevanja delovanja realnega sveta.

Metode in tehnike ugotavljanja informacijskih potreb se pojavljajo kot samostojne metode ali pa kot sestavni del celovitejših metodologij gradnje IS. Razvrstili bi jih lahko v naslednje skupine:
1. avtomatizacija dela z zbrano dokumentacijo (shranjevanja in analiziranja)
2. modeliranje ciljnega sistema predvsem v grafični obliki, ki je sprejemljivejša bodočim uporabnikom
3. infološko utemeljene metode, ki so usmerjene predvsem v informacijske potrebe uporabnikom na različnih ravneh in problemskih področjih organizacije

Značilni infološki pristopi, ki so kot metode ali metodologije zaključene celote, so:
- ISAC,
- BSP,
- BICS,
- informacijska analiza (Information Analysis),
- CSF - pristop ključnih dejavnikov uspešnosti (Critical Success Factors)

ISAC (Information System Work and Analysis of Change)
1. metodologija izvira iz Švedske
2. je pristop, ki edini združuje analizo problemov in podatkov, s pomočjo diagramov aktivnosti pa v obliki globalnega modela podaja celovito in razumljivo sliko delovanja organizacije
3. metodološke slabosti izhajajo iz slabo opremljenega postopka analize problemov in zanemarjanja vpliva organiziranosti na te probleme
4. postopek preverjanja je lahko le ročen in ga ni možno avtomatizirati

Zajema naslednje faze oz. skupine aktivnosti:
1. **Analiza sprememb**, ki je namenjena ugotavljanju problemskih sklopov oz. področij, potrebnih izboljšav. Izvedene so v naslednjih aktivnostih:
- analiziranje problemov, trenutnega stanja in potreb po spremembah
- ugotavljanje možnih alternativnih rešitev
- izbira rešitev

Rezultati te faze so prikazani v obliki diagramov aktivnosti, tabel subjektov komuniciranja in potrebnih sprememb ter opisa aktivnosti, subjektov in nerešenih problemov.
2. **Analiza in zasnova IS**, ki zajema naslednje aktivnosti:
- podrobno razčlenjevanje aktivnosti
- analiziranje podatkov v smislu pripadnosti tipom entitet, njihovega izvora, ponora in medsebojnih relacij
- opredeljevanje tehnološke opreme

Rezultati te faze so vsebinsko podobni predhodni, vendar brez nerešenih problemov in z mnogo podrobnejšim prikazom podatkov, aktivnosti, ki pretvarjajo vhodne v izhodne podatke, ter tokov podatkov med posameznimi aktivnostmi.

BSP (Business System Planning)
1. je metodologija, ki jo je v začetku 70-ih let IBM ponudil svojim uporabnikom
2. je dobro opremljena s številnimi priročniki in navodili za uporabo
3. uporabljena kot osnova različnih avtomatiziranih orodij za načrtovanje informatike (CASE orodij)
4. metodologija je celovita s pristopom od zgoraj navzdol se najprej loteva ciljev organizacije in poslovnih procesov (predstavljajo osnovo za zbiranje in analiziranje podatkov)
5. Ključni odločevalci (subjekti, ki s svojimi odločitvami bistveno vplivajo na obnašanje in s tem na rezultate izvajanja posameznih aktivnosti, skupin aktivnosti, poslovnih funkcij in organizacije kot celote) v organizaciji na ta način opredeljujejo svoje ključne dejavnike uspeha in nastopajoče probleme.

6. je ena najcelovitejših metodologij razvoja informatike, primerno za reševanje kompleksnih problemov
7. izredno dobro dokumentira rezultate
8. prehod v izvedbo je eden ključnih problemov
9. slabosti izhajajo tudi iz odsotnosti sistematičnih postopkov pri ugotavljanju posameznih aktivnosti in celotnega organizacijskega ustroja

CSF - pristop ključnih dejavnikov uspešnosti (Critical Success Factors)
1. ta metodologija zajema postopke, s pomočjo katerih lahko opredelimo posamezna ključna področja;
2. izhaja iz strukturnih dialogov med načrtovalcem in ključnimi odločevalci v organizaciji;
3. usmerjena je na ugotavljanje ključnih dejavnikov uspeha posameznikov in z njihovo pomočjo opredeljenih informacijskih potreb
4. ključne dejavnike predstavlja celovit in nivojsko strukturiran zbir individualnih dejavnikov
5. prednosti se kažejo v njeni učinkovitosti na področju strateškega načrtovanja in ugotavljanju potencialnih področij za zagotavljanje konkurenčne prednosti organizacije
6. ključni dejavniki uspeha odločevalcev so težje opredeljivi, bolj kot so le.ti v svojem organizacijskem nivoju odmaknjeni od najvišjega vodstva organizacije
7. na področju ugotavljanja dejanskih informacijskih potreb odločevalcev je smotrna uporaba metodologije CSF v povezavi s prototipnim pristopom

NAČRT RAZVOJA INFORMATIKE

Načrt razvoja informatike organizacije, ki zajema nivo strateškega načrtovanja informatike, kakor tudi gradnjo posameznih informacijskih sistemov, zajema naslednje faze in rezultate:

1. Strateško načrtovanje informatike: V tej fazi pričakujemo naslednje rezultate:
- opredeljeni cilji organizacije in ključni dejavniki uspeha,
- pregled problemov doseganja ciljev,
- opredeljeni ključni dejavniki uspeha organizacije na področju informatike

2. Razvijanje informacijske strukture: Pričakovani rezultati:
- prikaz organizacijske strukture,
- model poslovnih funkcij in aktivnosti,
- model tokov podatkov,
- globalni model podatkov,
- izhodišča razvoja IS v organizaciji:

3. Modeliranje podatkov, razvijanje baze podatkov in uporabniških rešitev: Pričakovani rezultati: - model podatkov organizacije,
- katalog podatkov,
- prikaz vsebine ključnih projektov,
- fizični model podatkov,
- uporabniške rešitve.

(3. Poglavje) MODELIRANJE INFORMACIJSKIH SISTEMOV

3.1.TEORETIČNA IZHODIŠČA:

Pri modeliranju realnega sveta smo soočeni s problemi preslikave stvarnosti v nek model:
1. model ni nikoli odraz stvarnosti same, ampak nastane na osnovi predstav o stvarnosti, ki si jo ustvari opazovalec - prihaja do popačenj;
2. model nastane na osnovi predstav o stvarnosti v očeh opazovalca, zato različni opisovalci pri opisovanju stvarnosti pridejo do različnih modelov, ki pa so vsi enako smiselni.

Za vernost preslikave stvarnosti v model vpliva:
1. pogled na svet tistega, ki to preslikavo dela
2. instrumentarij, ki ga imamo na voljo

stvarnost → preslikava v očeh opazovalca → model

predstave o stvarnosti → pogled na svet + instrumentarij → model

Načrtovanje in gradnja IS je timsko delo, pri čemer je pomembno tesno sodelovanje strokovnjakov izvajalcev in pa uporabnikov. Modeli služijo za komunikacijo med člani tima, zato morajo biti čimbolj poenoteni.
Pod **inštrumentarijem** razumemo dobro definirana orodja, tehnike ali koncepte, ki jih uporabljamo pri izdelavi modela. Pogojno lahko rečemo da velja:

pogled na svet + inštrumentarij → metodologija

Metodologija, ki jo uporabljamo za reševanje nekega problema, je vedno odraz pogleda na svet in inštrumentarija, ki ga imamo na voljo. Pri modeliranju IS se inštrumentarij, ki ga uporabljamo, sestoji predvsem iz različnih konceptov, s katerimi skušamo ponazoriti lastnosti IS. Množica konceptov, s katerimi operira metodologija, predstavlja gradbeni material, s katerimi lahko zgradimo model.

Zgradba IS in predstavitev znanja o IS
Modeliranje IS zahteva predstavitev ter opredelitev naslednjih karakteristik IS oz. njegovih komponent:
1. Strukture postopkov: Pod to strukturo razumemo hierarhično strukturo, do katere pridemo postopoma. Začnemo s procesi na najvišjem nivoju in jih s pomočjo funkcijske dekompozicije razstavljamo, dokler ne pridemo do elementarnih procesov.
2. Kontrolna struktura: Definira postopkovne karakteristike sistema, povezane med njegovimi elementi in dovoljena stanja sistema.
3. Podatkovni tokovi: So krvni obtok IS. Prek teh tokov IS oskrbuje poslovni sistem z informacijami. Kažejo pa nam tudi prenosno funkcijo posameznih procesov ter njihove vhode in izhode.

4. Podatkovne strukture: So hrbtenica vsakega IS. Kažejo nam karakteristike in razmerja med podatki, ki tvorijo bazo načrtovanega sistema.
5. Predstavitev povezav med procesi in podatki: Pomembna je zaradi nadzora nad integriteto in konsistenco načrtovanega sistema.
6. Vhodno/izhodne maske in poročila: Predstavljajo povezave, prek katerih posamezne komponente sistema komunicirajo med seboj oz. s svojo okolico.

Glavni gradbeni elementi pri modeliranju IS:
Pri modeliranju poljubnega sistema je potrebno običajno predstaviti naslednje lastnosti sistema:
1. objekte sistema (podatkovne in procesne);
2. povezave med objekti, ki tvorijo strukturo sistema,
3. statične lastnosti sistema,
4. dinamične lastnosti sistema;
5. omejitve (integritetne, strukturne, dinamične)
Pri modeliranju IS je teže predstaviti dinamične lastnosti kot statične ker zahtevajo drugačen pristop pri modeliranju.

Da lahko izvedemo modeliranje moramo poznati:
1. podatke ki jih potrebujemo
2. postopke ki se izvajajo
3. povezave med podatki in postopki

Težava, s katero se srečamo pri modeliranju pa je tudi, da želimo v različnih razvojnih fazah izpostaviti različne lastnosti načrtovanega sistema, torej da imamo v življenjskem ciklusu opraviti z različnimi modeli. V zgodnjih razvojnih fazah je model na konceptualni ali logični ravni, ki je neodvisna od konkretne izvedbe na izbrani stopnji in programski opremi, torej stopajo v prvi plan "fizične" lastnosti načrtovanega sistema.
Glavni problem pa je dokumentiranje sistema v posamezni razvojni fazi ter prenos znanja iz faze v fazo.Te probleme je mogoče rešiti le z uporabo CASE orodij ter sistemskih knjižnic, ki morajo biti sestavni del teh orodij.

Modeliranje IS je sestavni del razvojnega procesa ima pa še druge funkcije. Model IS je tudi sredstvo, ki mora omogočati naslednje:
- komunikacijo med sodelavci skupine, ki dela na razvoju IS,
- komunikacijo z uporabniki bodočega IS,
- zagotavljati dokumentiranje značilnosti načrtovanega IS na ravni zasnove in ne na ravni izvedbe, kar omogoča enostavnejše vzdrževanje in prilagoditve sistema.

Rešitev modeliranja IS na konceptualni oz. logični ravni bi morala biti sestavljena iz naslednjih elementov:
1. Modeliranega mehanizma z naslednjimi lastnostmi:
- biti mora čimbolj natančen in intuitiven
- biti mora dovolj bogat za predstavitev vseh pomembnih lastnosti načrtovanega IS
- biti mora neodvisen od fizičnih specifičnosti posameznih računalniških sistemov
- omogočati mora formalno specifikacijo vseh zgoraj navedenih elementov IS in uporabo avtomatiziranih orodij pri nadaljnjem razvoju IS

2. ustreznega formalizma za opis konceptualnega ali logičnega modela;
3. grafične notacije za predstavitev modela;
4. programska orodja za podporo izdelave modela in njegovo grafično predstavitev

3.1.2. OPREDELITEV OSNOVNIH KONCEPTOV ZA MODELIRANJE STATIČNIH IN DINAMIČNIH LASTNOSTI IS

Ključni problem obravnave informacijskih sistemov je obvladovanje kompleksnosti, s katero smo soočeni že pri nekoliko obsežnejših sistemih. Ta problem je mogoče reševati samo z abstrakcijo, to je tako, da se osredotočimo samo na z vidika obravnave pomembne karakteristike obravnavanega sistema ter zavestno zanemarimo vse ostalo. To pa zahteva vpeljavo ustreznih abstraktnih konceptov, s katerimi skušamo predstaviti lastnosti obravnavanega sistema.
Modeliranje IS temelji na uporabi abstraktnih konceptov. Večje število konceptov pri izdelavi ter specifikaciji modela IS omogoča boljšo predstavitev lastnosti načrtovanega IS.

Seznam konceptov, ki jih bomo uporabljali (koncepti za modeliranje IS):
- <entiteta>,
- <atribut>,
- <povezava>,
- <vrednost>,
- <postopek>,
- <operacija>,
- <začetni pogoj>,
- <končni pogoj>,
- <dogodek>,
- <sporočilo>

Nabor konceptov je tak, da omogoča specifikacijo IS z njegovimi statičnimi (podatkovnimi) in dinamičnimi (postopkovnimi) lastnostmi.

3.2. MODELIRANJE PODATKOV

podatki in podatkovni modeli
PODATEK
- je opis, zapis nekega pojava ali dejstva in je lahko predstavljen v številčni, tekstovni ali grafični obliki.
- je surovina, ki se pretaka skozi IS in se v njem obdeluje, shranjuje, v končni fazi so tudi njegov produkt. Modeliranje podatkov in podatkovni modeli so se razvijali vzporedno z razvojem informacijske tehnologije.
- podatki s katerimi opisujemo pojave, dejstva, se nanašajo na neke objekte oz. entitete (to so lahko subjekti, objekti v ožjem smislu ali pojmi), ki nastopajo v okviru obravnavanega dela stvarnosti in pomembni z vidika obravnave.
- podatek ima statične lastnosti, dinamične lastnosti (ali se lahko spreminja ali ne – EMŠO se ne sme – in pod kakšnimi pogoji), integritetne omejitve (ne moreš imeti r.d. 30.02.)

podatkovni model:

- nam omogoča delni opis obravnavanega izseka stvarnosti oz. konkretnega sistema, ki ga obravnavamo;
- je zbirka konceptov, s katerimi skušamo izraziti statične in dinamične lastnosti podatkov v okviru IS.
- Vsak objekt, ki ga predstavljamo s pomočjo podatkovnega modela, bi moral biti predstavljen s četverko: <ime_ objekta, lastnosti_objekta, vrednosti_lastnosti, čas>

Modeliranje podatkov gre v razvojnem procesu IS skozi več razvojnih faz. Zaključena faza je fizična zasnova podatkovne baze v izbranem krmilnem sistemu baze podatkov, kar omogoči začetek polnjenja in uporabe baze podatkov. Najbolj prikladno bi bilo, da bi skozi vse faze modeliranja podatkov v okviru razvojnega cikla IS lahko uporabljali isti podatkovni model. Pri modeliranju podatkov je najpomembneje, da je uporabljani model čim preprostejši, da ga razumejo tudi netehnično izobraženi uporabniki, hkrati pa mora omogočati verno predstavitev semantike podatkov.

Osnovni gradbeni elementi modeliranja podatkov:
Z modeliranjem podatkov skušamo Čimbolj verno predstaviti neki izsek realnega sveta ter njegove lastnosti. Pri tem se nam kot osnovni modelirni koncepti predstavljajo:
1. entiteta
2. atribut
3. povezava
4. vrednost

❖ **ENTITETA:** - je neka reč (objekt, subjekt ali pojem), ki obstaja v realnem svetu, je pomembna z vidika načrtovanega IS. Entitete so lahko fizične ali abstraktne narave; Podatki v okviru konkretnega sistema, ki ga modeliramo se nanašajo na entitete.
❖ **ATRIBUTI ENTITET**: - je opisna lastnost nekega tipa entitete, ki jo lahko pripišemo celotni množici primerkov danega tipa; - entitete opisuje končna množica atributov, - tip entitete je agregat njegovih atributov.
❖ **POVEZAVA:** - je logična ali vsebinska zveza med dvema ali več tipi entitet, ki je pomembna z vidika obravnavanega IS; - v splošnem lahko opisuje zvezo med n-tipi entitet

3.2.1. KONCEPTI ABSTRAKCIJE PRI MODELIRANJU PODATKOV

ABSTRAKCIJA: - je način razmišljanja oz. reševanja problema, pri katerem zavestno zanemarimo podrobnosti in se osredotočimo na splošne, skupne lastnosti pojavov, objektov ali pojmov, odvisno od narave problema, ki ga rešujemo;
Tak pristop je še posebno pomemben na področju IS, ker gre za sisteme, v katerih nastopajo predvsem objekti abstraktne narave

koncepti abstrakcije:

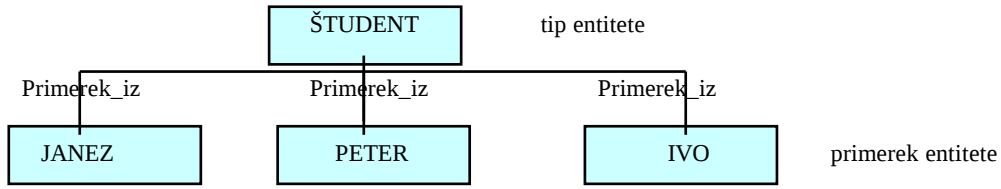
nam pri modeliranju podatkov omogočajo predstavitev strukture podatkov, njihovo kategorizacijo, hierarhično razvrščanje, poglobljeno semantično obdelavo.

V podatkovnih modelih zasledimo naslednje koncepte abstrakcije:
1. klasifikacija
2. generalizacija
3. agregacija
4. asociacija

KLASIFIKACIJA:

- je koncept abstrakcije, pri katerem se množici primerkov entitet, ki imajo neko skupno lastnost, priredimo tip entitete, ki izraža skupno lastnost;
- isti množici primerkov entitet priredimo različne tipe;
- vsak tip entitete izraža skupno lastnost;
- vpeljati moramo razliko med **primerkom** (npr. Janez, Ivo...) in **tipom** tega primerka (študent, občan, delavec...) (slika spodaj)
- klasifikacije vpeljuje razmerje **primerek_iz** med primerkom in tipom objekta, ki mu primerek pripada.

Grafični prikaz koncepta klasifikacije



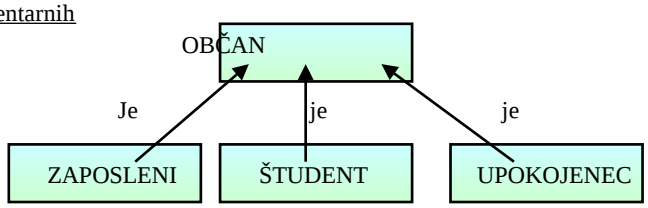
GENERALIZACIJA:

- je postopek abstrakcije pri katerem dvem ali več tipom entitet priredimo nek skupen, splošnejši, generaliziran tip entitete, ki ponazarja vse lastnosti entitet, ki jih ta generaliziran tip predstavlja. Značilno je dedno pravilo = da se lastnosti posplošenega tipa dedujejo po hierarhiji navzdol.

Generalizacija vpeljuje razmerje **je** med elementarnimi tipi in posplošenim tipom na višji ravni.

Tipi entitet DELAVEC, ŠTUDENT, UPOKOJENEC imajo skupne atribute: <EMŠO, ime, priimek, naslov...>; specifične atribute: ZAPOSLENI <šifra_org, poklic, šifra_del_mesta...>; ŠTUDENT <šifra_šole, letnik, status...>; UPOKOJENEC <šifra_zavarovanja, leto_upokojitve...>.
Po dednem pravilu ima ZAPOSLENI poleg svojih lastnosti še vse lastnosti tipa OBČAN. Posplošeni tip OBČAN pa nima nekih svojih atributov, ki se ne bi dedovali po hierarhiji navzdol.

Primer generalizacije elementarnih tipov v posplošene tipe



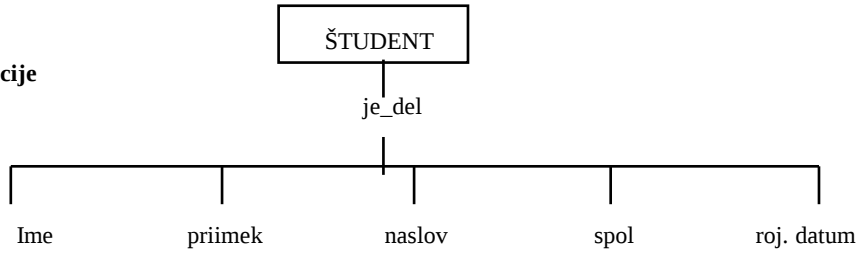
AGREGACIJA:

- je oblika abstrakcije, pri katerih se objektom, ki predstavljajo dele sestavljenega objekta, priredi sestavljeni objekt na višjem nivoju;
- uporablja se lahko na več nivojih = vzpostavlja se relacija **je-del** med objekti in njihovim agregiranim objektom;
- agregacija nas pripelje do hierarhije ki jo lahko poimenujemo je-del hierarhija. Ta izraža strukturo nekega sestavljenega objekta;

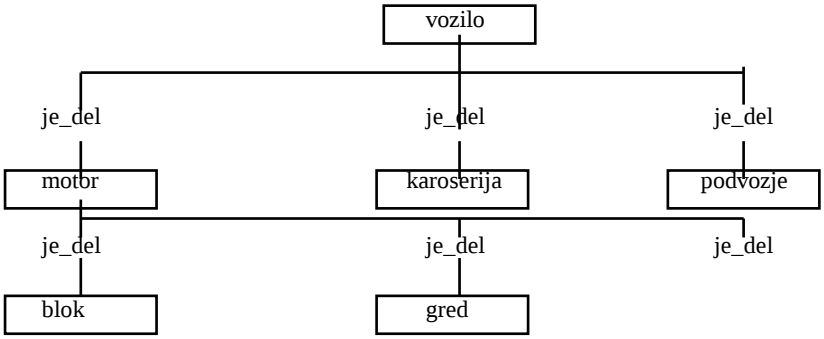
V semantičnih modelih se uporabljata dve obliki agregacije:

1. kartezična: tu razumemo objekt kot agregat njegovih lastnosti; (atributom priredimo tip entiteto)
2. pri drugi razumemo objekt kot agregat drugih objektov (prikažemo sestavne dele nekega objekta)

Primer kartezične agregacije



Uporaba koncepta agregacije na ravni objektov



ASOCIACIJA:

- je oblika abstrakcije podatkov, pri kateri se podmnožici primerkov nekega tipa priredi na višjem nivoju asociirani tip, ki predstavlja skupne lastnosti asociiranih primerkov;
- pripelje nas v razmerja **je-član** med primerki nekega tipa in asiciiranim tipom na višji ravni;
- za asociiran tip velja dedno pravilo = veljajo vse skupne lastnosti asociiranega tipa, poleg tega pa ima še določene svoje lastnosti, ki so skupne celotni množici primerkov določenega tipa

3..2.2. RAZVOJ PODATKOVNIH MODELOV

Podatkovni model naj bi omogočal specifikacijo naslednji značilnost objektov, ki nastopajo v stvarnosti:

1. statičnih oz. strukturnih lastnosti, kot so specifikacija objektov, lastnosti objektov in povezav med objekti;
- to so tiste lastnosti podatkov, ki se s časom razmeroma malo spreminjajo, so stalne;
- množica izvedbenih pravil izraža statične lastnosti podatkovnega modela, vsebovana je v jeziki opisov podatkov;
- množica določa dovoljene strukture podatkov v okviru modela, zato te lastnosti lahko poimenujemo **strukturne** lastnosti;
- izvedbena pravila so osnova za specifikacijo množice shem, vsaka shema določa konkretno podatkovno strukturo z omejenimi pravili

2. dinamičnih lastnosti, to je specifikacijo operacij na objektih, lastnosti operacij in povezav med operacijami (formiranje transakcij);
- opredeljujejo spremembe na podatkih, ki so odvisne od časa oz. od sprememb realnega sveta;
- množica dopustnih operacij določa dinamične lastnosti podatkovnega modela = vsebovana v jeziku za upravljanje baze podatkov;
- množica operacij določa dovoljene akcije na bazi podatkov, s katerimi se spremeni stanje baze podatkov

3. integriranih omejitev nad objekti in operacijami (dovoljenih stanj baze podatkov in prehodov med stanji)

Modele razdelimo v 4 skupine:

1. primitivni modeli,
2. klasični modeli,
3. semantični modeli,
4. specializirani modeli.

Primitivni in klasični modeli so bili usmerjeni izključno v fizično zasnovo in specifikacijo podatkovne baze. Šele z nastankom prvih semantičnih modelov se pozornost vedno bolj usmerja na logične ali semantične aspekte modeliranja podatkov.

1. PRIMITIVNI PODATKOVNI MODEL:

- predstavljajo podatkovne objekte v obliki zapisov, grupiranih v datoteke (ima 3 nivojsko strukturo: polje, zapis, datoteka); (Datoteka je množica zapisov).
- možnosti povezovanja objektov oz. vzpostavljanja povezav med njimi so zelo skromne (indeksne in invertirane tabele);
- predstavljajo začetno stopnjo modeliranja podatkov;

- uporabljajo se v okviru IS, kjer količine podatkov niso velike.

2. KLASIČNI PODATKOVNI MODELI: Sem uvrščamo:

- **relacijski model**: temelji na matematičnem konceptu relacije, množice n-teric - omogočajo predstavitev tipov entitet in tipov povezav med relacijami
- **hierarhični model** in
- **mrežni model**: sta nadgradnja dotedanjih datotečnih modelov, objekti so predstavljeni s segmenti oz. zapisi, med katerimi so vzpostavljene povezave 1: n, kar nas pripelje do drevesne strukture, v kateri vozli predstavljajo segmente ali zapise

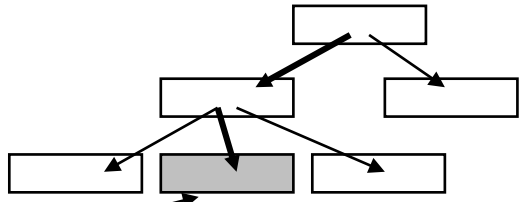
Hierarhični podatkovni model:

- razvili so se intiutivno, bolj skozi praktično uporabo kot teoretično,
- osnovna struktura je narobe obrnjeno drevo;
- na najvišjem nivoju imamo samo 1 zapis, temu zapisu podrejeno različno število zapisov
- povezave so sestavni del modela in so vnaprej opredeljene.;
- uporaba tega modela zahteva da predvidimo na kakšen način se bodo podatki uporabljali (možno pri majhnih IS).

Osnovni koncepti hierarhičnega modela:
V hierarhičnem modelu nastopata dva temeljna modelirna koncepta, **zapisi** ter povezava **oče-sin**. **Zapis** je zbirka polj, ki vsebujejo podatke o primerkih entitet ali povezav med njimi.
Povezava **oče-sin** je povezava tipa 1:N med dvema tipoma zapisov. Zapis na strani 1 se imenuje oče, zapis na strani N pa sin. En primerek povezave oče-sin se torej sestoji iz enega primerka zapisa tipa oče ter več primerkov zapisa tipa sin. Grafično se hierarhična shema prikazuje kot hierarhični diagram, v katerem so tipi zapisov predstavljeni kot pravokotniki, povezave oče-sin pa kot daljice, ki povezujejo zapis tipa oče z zapisom tipa sin. (slika spodaj)

Lastnosti hierarhične sheme baze podatkov:
1. Drevesna struktura;
2. imamo vozlišča, na najvišjem nivoju en vozel;
3. vsak zapis ima vedno enega samega nadrejenega in večje število podrejenih;
4. povezave so vnaprej opredeljene;
5. povezave so poti po katerih iščemo zapise;
6. vstopna pot je dejansko ena sama.
7. na najvišji ravni v hierarhiji je en sam tip zapisa, ki se imenuje koren, ta nima nadrejenih zapisov tipa oče.
8. do določenega zapisa lahko pridemo le v točnem določenem zaporedju, čim želimo dostop do zapisov po nekem drugem sosledju, ki ga v hierarhični shemi ni bilo vnaprej predvideno, se začnejo kazati slabosti tega modela.

Primer hierarhičnega modela:



Točno določena pot do podatka.

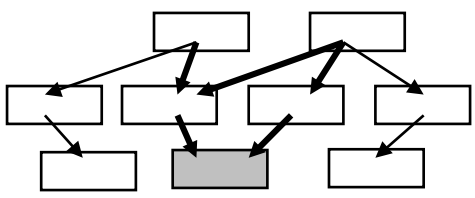
Mrežni podatkovni model:

- na najvišjem nivoju nimamo enega zapisa, ampak več poljubnih;
- več vstopnih točk do iskanih podatkov od katerih ima vsak lahko poljubno število podrejenih in nadrejenih;
- bolj fleksibilna raba podatkov ;
- slabosti so podobne kot pri hierarhičnem modelu; dostop do podatkov je možen le na način, ki je bil vnaprej predviden in vgrajen v shemo mrežnega modela.

Koncepti mrežnega modela**:**

1. tip zapisa
- podatki se v tem modelu shranjujejo v zapise, ki vsebujejo skupino polj z medsebojno povezanimi vrednostmi podatkov
- zapisi se kvalificirajo v tipe zapisov, kjer vsak tip zapisa določa strukturo skupine zapisov v katerih so shranjene informacije istega tipa
- vsak tip zapisa ima svoje ime, svoje ime ima polje (atributi), ki sestavljajo določen tip zapisa

2. tip povezave
- opredeljuje povezave 1:N med tipi zapisov
- Vsak tip povezave se sestoji iz 3 elementov:
    1. imena tipa povezave,
    2. zapisa tipa lastnik,
    3. zapisa tipa član



primer ko ima vsak zapis več podrejenih ali nadrejenih zapisov in ko je dostop do določenega podatka možen po več poteh.

Relacijski podatkovni model:

- podoben modelu entiteta - povezava (E-R);

Razlike:

- relacijski model ima enoznačno matematično podlago, njegovi koncepti so jasno formulirani, omogočajo uporabo relacijske algebre pri izvajanju operacij na modelu;
- E-R model je nastal bolj intituitivno, brez enotnega matematičnega formalizma, v praksi ima vedno več interpretacij
- Prednost relacijskega modela je da povezave niso vnaprej določene in se vzpostavijo odvisno od trenutnih potreb.

Osnovni koncepti relacijskega modela:

- relacije so zasnovane okrog entitet, ki jih identificiramo v okviru OS;
- relacija je dvodimenzionalna tabela, ki se sestoji iz določenega števila stolpcev in vrstic;
- število stolpcev odvisno od števila atributov, ki jih ima neka entiteta;
- vrstice predstavljajo primerke entitet;
- ni vnaprej določenih povezav do podatkov;
- povezave se vzpostavijo v času obdelave podatkov, glede na uporabnikovo zahtevo
- povezave se tvorijo z indeksi, (tu gre za sprotno povezovanje struktur, pristop je počasnejši, problematično je ažuriranje, zahteva zmogljivo strojno opremo)
- Ena relacija nam predstavlja en tip entitete. Celotna tabela je relacija
- Število atributov določa število stolpcev.
- Število primerkov tipa entitet določa število vrstic.

primer relacijske tabele:

ŠTUDENT			
IME	Vpis_št.	Letnik	Spol
Ivo	13011	1	M
Janez	13014	2	M
Ana	13018	1	Ž

Primerjava med :

hierarhični	**mrežni**
ustrezen za prikaz realnega sveta	CODASYL – ANSI
zapis je zbirka polj o primerkih entitete in povezavah	zapis je skupina polj
povezava oče:sin 1:N	zapisi se klasificirajo v tipe zapisov z dololčeno strukturo in imenom
vsak zapis ima enega nadrejenega in poljubno število podrejenih	tip povezave se sestoji iz imena tipa povezave, zapisa tipa lastnik, zapis tipa član
listi nimajo očeta	edini omogoča sestavljene atribute in ponavljajoče se grupe
edini uspeli model je od IBM (IMS)	

Za oba modela velja, da je dostop do podatkov mogoč samo po predvidenih poteh.

3.2.3 SEMANTIČNI (pomenski) PODATKOVNI MODEL (to ni izvedbeni model)

Novi modeli praviloma uvajajo širše in bogatejše koncepte modeliranja podatkov, ki omogočajo boljši prikaz "pomena" oz. semantike podatkov v fazi modeliranja krmilnih sistemov in izdelave konceptualnega ali logičnega modela IS. To modeli prikazujejo vlogo posameznih podatkov.

Novejše modele lahko razvrstimo:

1. neposredna razširitev klasičnih modelov,
2. matematični,
3. statistično-semantični hierarhični,
4. dinamično- semantični- hierarhični,
5. objektno orientirani modeli.

Izmed doslej znanih semantičnih modelov je najbolj uveljavljen E-R model. E-R model je najbolj prikladen za modeliranje na logični ravni, saj ni omejen z restrikcijami komercializiranih KSBP.

MODEL ENTITETA - POVEZAVA (E-R):

- sodi v kategorijo direktnih naslednikov klasičnih modelov, nastal je leta 1976;
- združuje lastnosti mrežnega in relacijskega modela;
- prednost je enostavnost konceptov in možnost enostavne transformacije;
- dober je pri izražanju statističnih in strukturnih lastnosti podatkov;
- šibak je za izražanje dinamičnih lastnosti
- E-R model je bil razvit za modeliranje IS na logični ravni, ni pa ga mogoče uporabiti za fizični ali izvedbeni model ampak ga je potrebno spremeniti v enega od klasičnih modelov – ponavadi v relacijskega.
- To je edini semantični model, ki se je doslej uveljavil v praksi in to za modeliranje podatkov po logični ravni

Teoretične osnove E-R modela:

- nima enoznačne matematične formulacije, kot jo ima relacijski model;

- temeljni konstrukti pa imajo matematično osnovo, ki jo predstavljajo znani matematični kompleksi (množica, kompleks in relacija)

Osnovni koncepti E-R modela:

1. **entiteta**: (obstaja v realnem svetu ali v naših predstavah in je pomembno za obravnavani IS. To je lahko neki objekt, subjekt, ki fizično obstaja npr: VOZILO, HIŠA, DRŽAVLJAN ..., lahko pa je organizacijski ali nek drug konceptualni pojem npr.: PODJETJE, ODDELEK, SEMINAR ... Vsaka entiteta ima določene lastnosti, ki jih imenujemo atributi);
2. **povezava**;
3. **atributi** : (opisujejo oziroma določajo lastnosti entitet. Tako npr. Entiteto DRŽAVLJAN opisujejo atributi: priimek, ime, spol, naslov ...).

Omogoča uporabo 3 konceptov abstrakcije: - klasifikacija; - generalizacija; - agregacija

Tipi entitet:

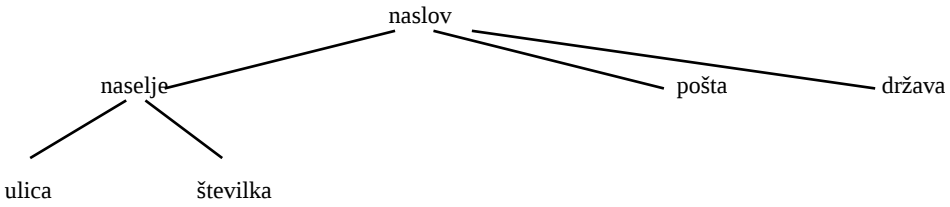
- tipe entitet predstavljajo primerki z enakimi lastnostmi;
- vsa imena tipov entitet morajo biti enolična;
- predstavljeni so z imenom (DRŽAVLJAN) in seznamom atributov (ime, Spol ...);
- entitetna množica vsebuje vse atributne vrednosti atributov tipa entitete

Osnovni E-R model pozna samo elementarne tipe entitet. V novejših modelih zasledimo možnost združevanja v posplošene tipe v smislu koncepta generalizacije.

Vrste atributov:

Nekateri atributi se lahko razstavijo na manjše celote, od katerih ima lahko vsaka svoj neodvisen pomen. Tako atribut NASLOV lahko razstavimo na naslednje dele: Ulica, Hiš._št, Naselje, Naziv_pošte, Poštna_št. Atributi ki os sestavljeni iz več elementarnih atributov se imenujejo sestavljeni atributi. (slika spodaj)

Elementarni: - predstavljajo eno samo lastnost entitete, ki jih ni mogoče sestaviti na manjše dele **sestavljeni atributi:** - predstavljajo skupino atributov in jih lahko sestavimo na elementarne atribute;



Ključni atributi entitete:

Primarni ključ: Vsaka entiteta ima praviloma atribut, katerega vrednosti omogočajo enolično identifikacijo posameznih primerkov entitet. Tak atribut imenujemo ključni atribut ali primarni ključ; Pri tipu entitete DRŽAVLJAN je primarni ključ EMŠO. Primarni ključ je obvezen. Vedno je en sam, lahko pa je več kandidatov (davčna_št., EMŠO). Kadar ni kandidatov, vpeljemo umetni atribut. Primarni ključ naj bo čimbolj kratek in numeričen.

Sekundarni ključ: Po njem iščemo podatke. Ne zagotavlja enolične identifikacije. Iskanje lahko postreže z več zadetki. Sekundarnih ključev je v neki entiteti lahko več, lahko so celo vsi. sekundarne ključe določijo končni uporabniki.

Sestavljeni ali speti ključ: Včasih moramo sestaviti več atributov skupaj, da dobimo primarni ključ, t.j. enolično identifikacijo vsakega primerka. Tak ključ imenujemo sestavljeni ali speti ključ. Pri tipu entitete ŠTUDENT atribut priimek ni dovolj za primarni ključ. Če pa spnemo Priimek + Ime + Roj._datum, pa dobimo dovolj zanesljiv primarni ključ. Ta ključ je alternativa umetnemu atributu. Običajno se nahaja v »presečnih entitetah«.

Tuji ključ: Nekatere entitete imajo lahko več atributov, ki imajo lastnosti primarnega ključa, tuji ključ je atribut, ki predstavlja primarni ključ neke druge entitete. Tuji ključi realizirajo povezave med entitetami. Lahko jih je več v eni entiteti. V slovarju entitet mora biti natančno toliko ključev, kolikor je narisanih povezav med entitetami. Tudi povezave 1:1 moramo realizirati s tujim ključem. To je edini atribut ki se lahko ponavlja.

Vsak tip entitete mora imeti vsaj en tak atribut, ki omogoča enolično identifikacijo njenih primerkov.

Lastnosti atributov:

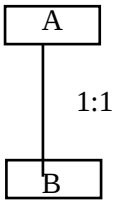
1. vrednost atributa: vsak A zavzame pri vsakem primerku entitete določene vrednosti, - če primarni ključ, zavzame različno vrednost (ime ne more biti primaren ključ, ker se vrednosti ponavljajo);
2. domena atributa: je množica vseh vrednosti, ki jih posamezen atribut lahko zavzame; (npr. Atribut Starost pri entiteti ŠTUDENT lahko zavzame vrednosti med 17 in 27. pri atributu Vpisna_številka lahko zavzame vrednost i med 00001 in 99999, spol M ali Ž)
3. večvrednostni atribut: kadar pri posameznem primerku entitete lahko nastopi več vrednosti, - enovrednostni: tu nastopi ena sama vrednost (spol M ali Ž, ne more biti oboje)

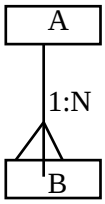
Lastnosti povezav:

1. Ločimo med tipi in primerki povezav. Primerek nam vzpostavlja dejanske povezave med posameznimi primerki tistih entitet, ki so med sabo povezane. Tip povezave občan je lastnik vozila;
2. Ime povezave: vsaka povezava ima ime, ki pomeni simbol, poudarja značaj razmerja med tistimi tipi entitet, ki so med sabo povezani; Ime označuje vsebino povezave.
3. stopnja povezave: nam pove število tipov entitet, ki so med sabo povezani oz. ki so v nekem razmerju; prevladujejo binarne povezave (povezave med dvema tipoma entitet);
4. kardinalnost: - kardinalnost opredeljuje število primerkov povezav, v katerih lahko neki primerek entitete sodeluje.

Poznamo 3 sisteme kardinalnosti pri binarnih povezavah:

- **1:1** (ena proti ena) en primerek tipa entitete A sodeluje v povezavi z enim primerkom tipa entitete B; (to razmerje velja za tip povezave VODI med tipoma entitet PREDSTOJNIK in ODDELEK)
- **1:N** (ena proti več): en primerek tipa entitete A sodeluje v povezavi z enim ali več primerki tipa entitete B;
- **N:M** (več proti več): en ali več primerkov tipa entitete A sodeluje v povezavi z enim ali več primerki tipa entitete B; (tip povezave UČI med tipoma entitet UČITELJ in ŠTUDENT)





Tip povezave predstavlja množico **primerkov povezav** med primerki entitet.

Tip entitete označimo s pravokotnikom, povezavo označimo s črto.

Obvezne in neobvezne povezave:

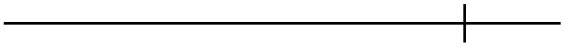
- obvezna, kadar mora biti primerek entitete A povezan z vsaj enim primerkom entitete B (entiteti GOSPODINJSTVO in ČLAN - povezava JE_ČLAN = gospodinjstvo lahko obstaja, kadar ima najmanj enega člana); povezava mora obstajati vsak trenutek
- neobvezna, primerki entitete A lahko participirajo v povezavi s primerki entitete B, ni pa nujno (entiteta ZAPOSLENI - povezava VZDRŽEVANEC, vendar ne vedno, ker mnogi zaposleni nimajo otrok, ki bi jih morali vzdrževati); ni treba da povezava obstaja vsak trenutek;

Rekurzivne povezave: so povezave, kjer sodeluje en sam tip entitete (povezava nadzira: pove kateri nadrejeni nadzira zaposlenega);

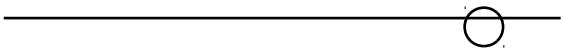
Eksistenčna odvisnost je obstoj entitete je odvisen od obstoja drugega tipa entitete; Taka odvisnost se kaže med entiteto ODDELEK in entiteto ŠOLA, saj entiteta ODDELEK lahko obstaja samo, če obstaja entiteta ŠOLA;

Identifikacijska odvisnost: kadar nek tip entitete nima sam nobenega takega atributa, ki bi nam zagotavljal enolično identifikacijo v tem primeru uporabimo primarni ključ tistega tipa entitete na katerega je ta tip identifikacijsko navezan (pacient, diagnoza)

Grafični prikaz obvezne povezave



Grafični prikaz neobvezne povezave



SLABOSTI E-R MODELA:

1. manjka mu enoznačna matematična formulacija, enoten standard glede pomena njegovih osnovnih konceptov in glede grafične predstavitve E-R diagramov,
2. ne omogoča povsem enotnega pogleda na podatke,
3. ne omogoča povsem enotnega modelirnega postopka,
4. ni univerzalni model, ki bi podpiral vse faze modeliranja podatkov,
5. od konceptov abstrakcije, ki so bili doslej definirani za potrebe modeliranja podatkov oz. predstavitve znanja, omogoča le klasifikacijo, agregacijo in z razširitvami generalizacijo;
6. ni standardiziran niti v pogledu uporabljenih konceptov niti grafične notacije

- **moč E-R** modela je v tem, da ni usmerjen v izvedbo baze podatkov niti v predstavitev baze podatkov, temveč v konceptualno predstavitev podatkov, kako jih vidi uporabnik

Grafična notacija E-R modela

Za vsak podatkovni model je pomembno, da ga je mogoče grafično predstaviti. Za modele ki jih uporabljamo pri modeliranju podatkov na logični ravni, pa je to še posebej pomembno, saj bi tak model bolje razumeli tudi uporabniki, ki jim je načrtovani sistem namenjen. Za model E-R obstaja preprosta grafična notacija. Na spodnji sliki so prikazani osnovni simboli.

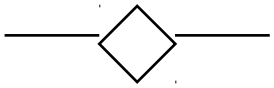
Tip entitete

Šibki tip entitete

Tip povezave

ime povezave napišemo na črto.

Tip povezave
(druga možna notacija)

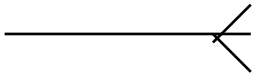


ime povezave vpišemo v romb
za povezave rišemo samo ravne črte s čimmanj križanj

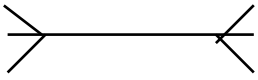
kardinalnost tipa povezave

1:1

1:N



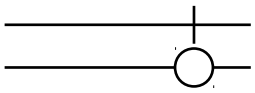
N:M



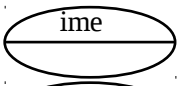
Obveznost povezave

obvezne

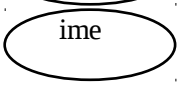
neobvezne



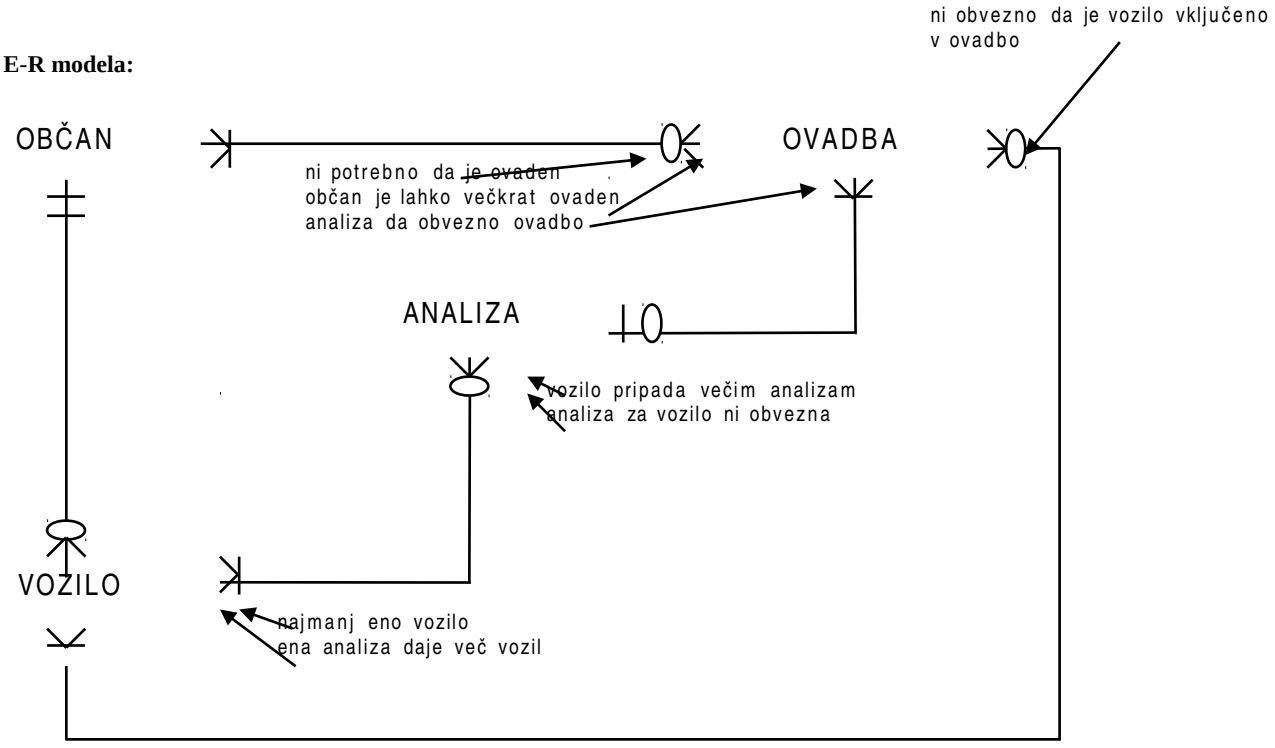
atributi

ključni 

vpiše se ime atributa

ostali 

Razvoj E-R modela:

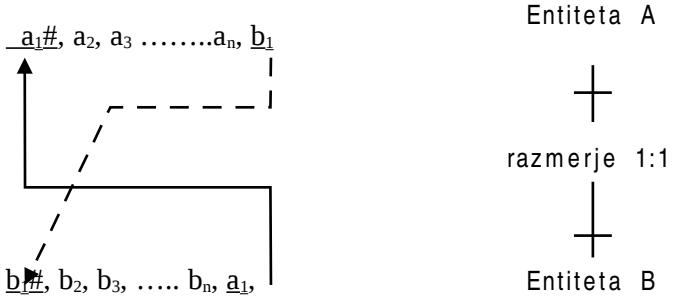


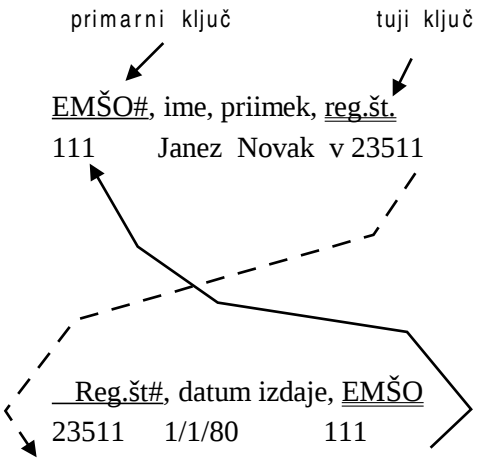
Analiza: tip, barva, letnik, kraj, datum, čas
Ovadba: datum ovadbe, ime, priimek
Vozilo: reg._št., tip, barva, letnik

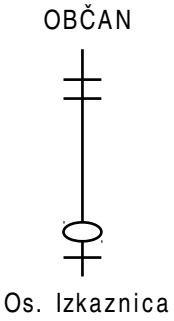
Tipične napake v E/R modelu:

1. Kartoteka, baza podatkov … niso entitete. Ima entitete naj odraža o čem zbiramo podatke
2. E/R diagram ni organizacijska struktura
3. Ne opazujemo samo enega pojava. delamo IS za 2 milijona ljudi za 10 let.
4. V E-R ne rišemo postopkovnih povezav, ampak tiste, ki so bistvene s podatkovnega vidika.

Vgradnja povezav s pomočjo tujih ključev:



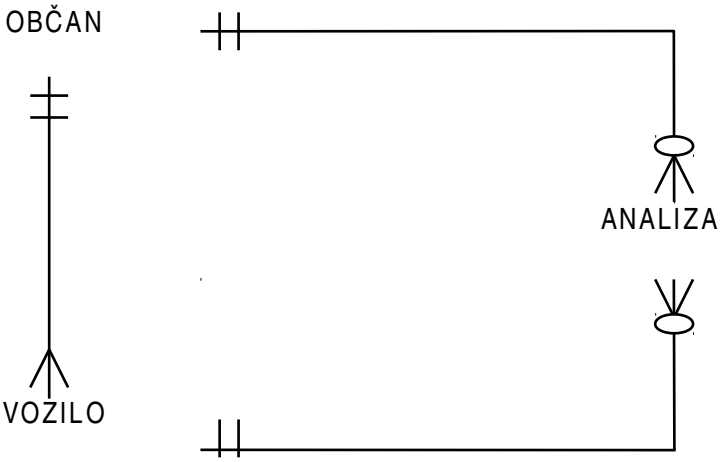




OBČAN: EMŠO#, ime, naslov

VOZILO: reg.št, barva, letnik, tip, EMŠO

Tuji ključ mora biti na strani grabljic



1. navaden atribut, ga normalno zapišemo
2. **primarni ključ#**
3. sekundarni ključ
4. speti ključ#
5. tuji ključ

Slovar entitet za zgornji primer:

Št. entitete	Naziv	Atributi
E-01	OBČAN	**EMŠO#**, ime, priimek, naslov, datum rojstva
E-02	VOZILO	**reg.št#**, tip, barva, letnik, št.šajsije, št. motorja, datum zadnje registr.
E-03	ANALIZA	BA, tip, letnik, poročilo, datum, laborant, **št.poročila#**, EMŠO, reg.št

Vsaka povezava ki ima grabljice mora imeti tuji ključ. Analiza ima dve grabljice in mora imeti dva tuja ključa.

Razlika med E/R modelom in informacijskimi potrebami:

1. potrebujemo dokumente, ki vsebujejo vse podatke za poslovanje
2. E/R model pa zahteva, da vsak atribut nastopa samo enkrat

samo znaki

datum

Slovar atributov za zgornji primer:

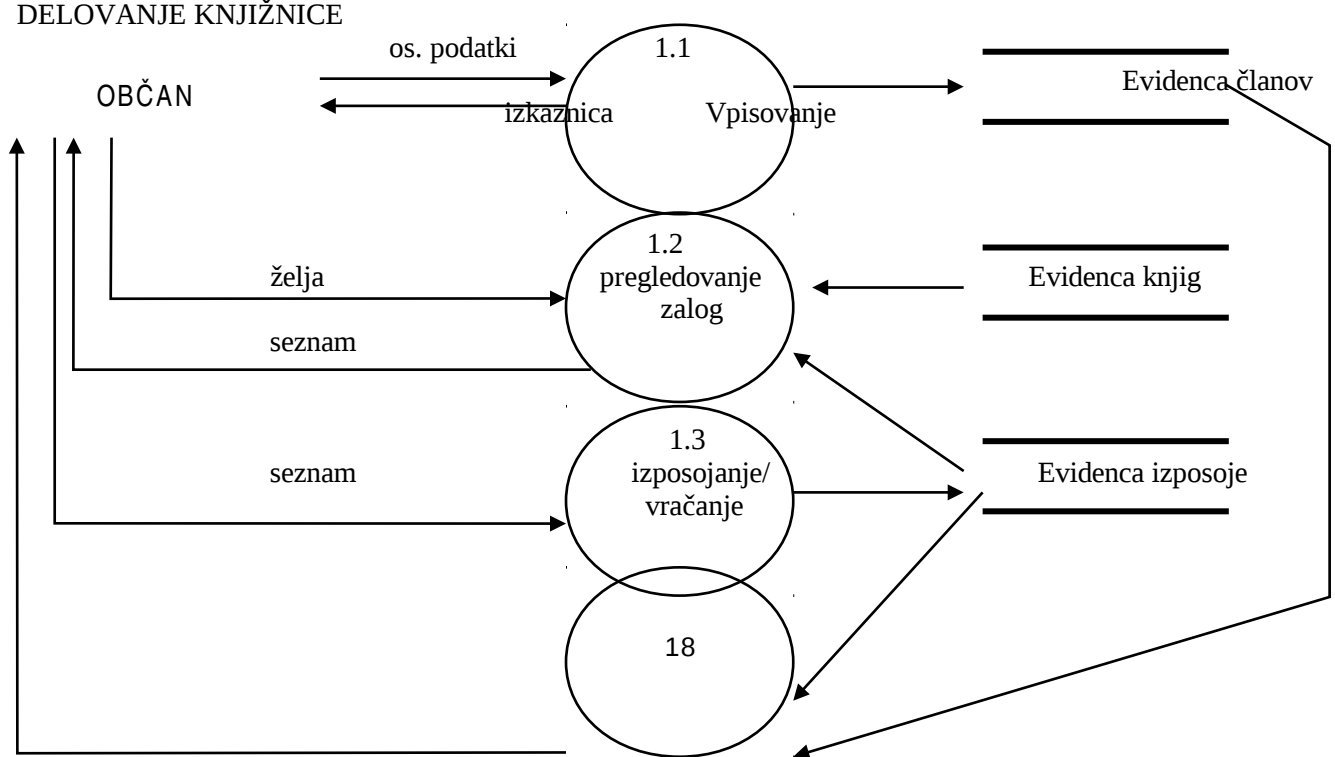
Oznaka atr.	ime atributa	standardno ime	tip	dolžina	stand. vrednosti
A-01	enotna mat. št. občana	EMŠO	N	13	modul 11
A-02	ime občana	ime	A	25	
A-03	priimek občana	priimek	A	25	
A-04	datum rojstva	datum rojstva	D	8	datum>1880
A-21	registrska številka vozila	reg.št.	AN	8	1-2(KR,LJ,MB,GO, NM,SG,KK,MS,CE,PO)
A-22	tip vozila	tip	AN	10	
A-23	barva vozila	BA	AN	10	
A-24	leto izdelave vozila	leto	D	4	

Slovar povezav za zgornji primer:

Oznaka povezave	povezani entiteti	ime povezave	kardinalnost
P-01	OBČAN:VOZILO	lastnik	1o:Mn
P-02	OBČAN:ANALIZA	osumljenec	1n:Mn
P-03	ANALIZA:VOZILO	preiskava	Mn:1n

kardinalnost:
1 – samo en primerek entitete
M – več primerkov entitete
O – obvezno najmanj en primerek mora obstajati v entiteti
N – neobvezno: obstoj pojava ni nujen

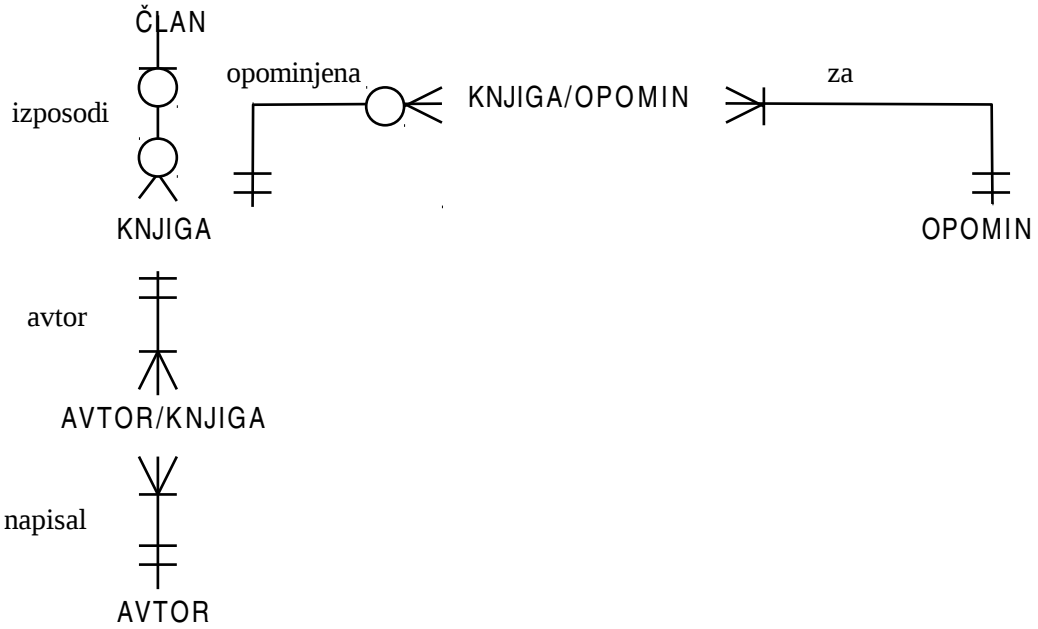
PRIMER:
DELOVANJE KNJIŽNICE



1.4

opomin opominjanje

Inf. potrebe:
Os. podatki: ime, priimek, naslov, telefon
Izkaznica: št. izkaznice, ime, priimek
Evidenca članov: št. izkaznice, os. podatki
Želja: avtor, knjiga, zvrst
Evidenca knjig: avtor+, naslov, leto, založba, zvrst, inv. številka
Evidenca izposoje: št. izkaznice, ime, priimek, inventarna številka, datum izposoje
Opomin: ime, priimek, naslov, inv. številka, naslov knjige, datum izposoje,
Seznam: avtor+, naslov, zvrst, inv. številka



E1	ČLAN	ime, priimek, naslov, **št. izkaznice#**
E2	KNJIGA	**inv.št#**, zvrst, naslov knjige, datum izposoje, leto, založba, št.izkaznice#
E3	AVTOR	**ID#**, ime
E4	AVTOR-KNJIGA	ID + inv. številka#
E5	OPOMIN	datum opomina, inv. številka, **številka opomina#**
E6	KNJIGA-OPOMIN	št.opomina + inventarna št.#

Objektno orientirani podatkovni modeli

- pri tem pristopu se baza podatkov sestoji iz množice objektov, kjer vsak objekt predstavlja neko fizično entiteto, koncept, idejo ali organizacijski pojem realnega sveta;
- objekti realnega sveta so predstavljeni z ekvivalentnimi objekti baze podatkov, kar omogoča ohranitev identitete objektov ter večjo primerljivost med realnim svetom in njegovim modelom v bazi;
- eden od ciljev objektnega pristopa pri modeliranju podatkov je obdržati ustrezno korenspondenco med realnimi objekti in njihovimi reprezentanti v bazi;
- objektni pristop dovoljuje, da v isto podatkovno strukturo posega množica med seboj lahko tudi povsem neusklajenih programov, vsak s svojo programsko kodo;
- nosilna ideja je vpeljava novega modelirnega koncepta objekt, ki je strukturno in dinamično celovito opredeljen.

Objektno orientirani modeli temeljijo na naslednjih osnovnih značilnostih in konceptih:;

1. **abstrakcija ter ogrevanje**: - abstrakcija je najučinkovitejši način za premagovanje kompleksnosti, s katero se vse bolj srečujemo na področju gradnje IS; Pod abstrakcijo razumemo pristop, pri katerem se osredotočimo na tiste lastnosti obravnavanega sistema, ki so z obravnavanega zornega kota pomembni, ostale pa zanemarimo. Pri objektnem pristopu gre za abstrakcijo na ravni funkcij - gre za funkcijsko diskompozicijo (razstavljanje na manjše, lažje obvladljive sklope) in za abstrakcijo na ravni podatkov. Na ravni podatkov gre za klasifikacijo razvrščanja primerkov v tipe, generalizacijo/specializacijo (uvajanje posplošenih tipov ter podtipov)
2. **objekti:** - so osnovni gradbeni bloki objektno orientirane baze podatkov; Objekt je opredeljen z ustrezno podatkovno strukturo in množico dovoljenih operacij na tej strukturi, ki jo imenujemo **množica metod.** Dostop do objektov in njihova obdelava sta mogoča prek ene od opredeljenih metod;
3. **hierarhija objektov**: - objekti so lahko združujejo popravilih generalizacije v posplošene objekte višjega tipa - to vodi do hierarhije objektov;
4. **dedno pravilo**: - lastnosti objektov se dedujejo navzdol po hierarhiji. Elementarni tipi objektov dedujejo atribute in metode posplošenih razredov objektov ;
5. **sestavljeni objekti**: - iz elementarnih objektov je mogoče sestavljati sestavljene objekte

Pričakovane prednosti objektno orientiranih modelov:

1. lokalizacija sprememb: poljubna sprememba strukture objekta ali njegovih metod je povsem lokalne narave ter ne zahteva posegov v druge objekte;

2. standardizacija objektov: objekt je skupaj z metodami povsem celovito opredeljen gradbeni blok sistema; ena podatkovna struktura se lahko obdeluje samo z eno in isto programsko kodo - olajša vzdrževanje konsistentnosti in integritete podatkovne baze;
3. predvidljivo obnašanje objektov: ker so objekti vseobsegajoči, je obnašanje vsakega objekta vnaprej določeno z fiksno množico metod, ki omejuje možne operacije v bazi podatkov

Podatkovni slovar

- Opisuje in razčlenjuje lastnosti podatkov OS

SLOVAR ENTITET: Opredeljuje vse tipe entitet, ki nastopajo v okviru OS, za vsak tip opredeljuje njegove atribute in med entitetami posebej označuje tiste atribute, ki nastopajo kot ključi. Za vsak tip entitete imamo oznako, ki mora biti enolična (E-01), sledi naziv entitete in nato seznam atributov
SLOVAR ATRIBUTOV: Podrobno opredeljuje lastnosti atributov enega tipa entitete. Atributi so lahko: numerični, alfabetski, alfanumerični. Slovar izdelamo za vsak tip entitete. Vsebuje podroben opis atributa, ki nastopajo. Oznaka atributa; ime atributa; standardno ime; tip; dolžina, standardne vrednosti
SLOVAR POVEZAV: Opredeljuje tipe entitet, ki sodelujejo v neki povezavi. Za vsako povezavo imamo oznako povezave, katere entitete sodelujejo v povezavi, kardinalnost. Oznaka povezave, povezane entitete, ime povezave, kardinalnost.

3.3. MODELIRANJE POSTOPKOV

Ko snujemo IS se moramo osredotočiti na 2 sklopa karakteristik:

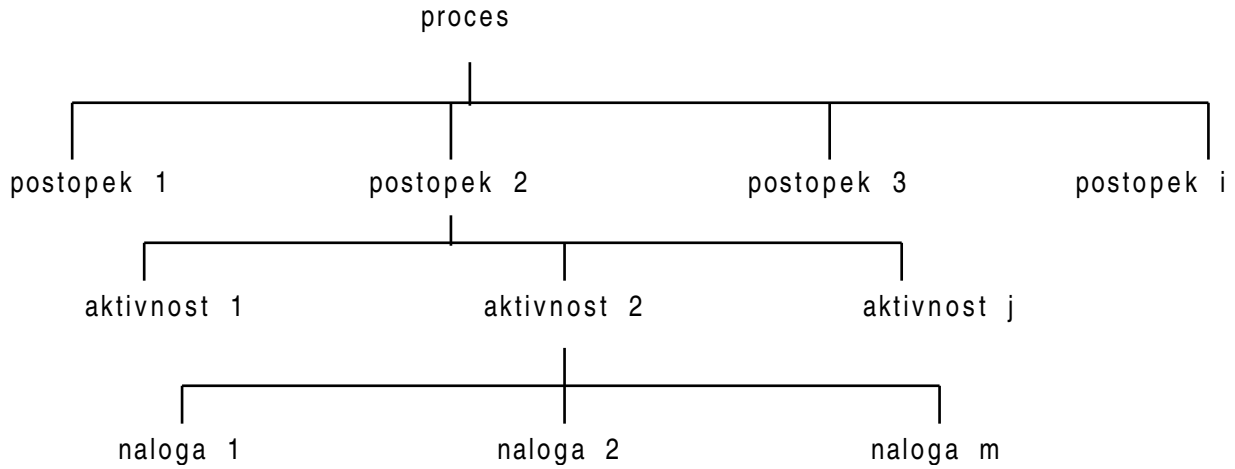
PODATKOVNI SKLOP – proučiti moramo podatke, ki nastopajo v okviru obravnavanega sistema.

POSTOPKOVNI (PROCESNI) SKLOP – moramo proučiti postopke. Procesi (postopki) vnašajo v naš sistem dinamiko (npr.: bančni IS – stanje na TR – stanje ostane nespremenjeno, dokler ne dvignemo denarja na bankomatu – izvrši se celotna transakcija dviga denarja).
Postopki nam nenehno spreminjajo stanje IS. Postopki IS služijo kot podpora poslovnim funkcijam organizacije ali podjetja. Postopki so del poslovnega procesa, ki se odvija v nekem okolju. Z modeliranjem postopkov modeliramo sam proces.

poslovna funkcija predstavlja skupino poslovnih aktivnosti, s katerimi se zadovoljuje en vidik delovanja organizacije ali podjetja. Poslovna funkcija sistema (organizacije, podjetja) se izvaja skozi postopke, ki so lahko avtomatizirani ali pa ročni.

Postopek je definirana poslovna ali proizvodna aktivnost, ki predstavlja logično zaključeno celoto, katere izvedba je opredeljena z vhodno/izhodnimi podatki in pravili za njihovo obdelavo. Postopki informacijskega sistema služijo kot podpora poslovnim funkcijam organizacije ali podjetja. Z modeliranjem postopkov in njihovo opredelitvijo posredno opredeljujemo tudi poslovna pravila oziroma pravila obnašanja obravnavanega sistema.

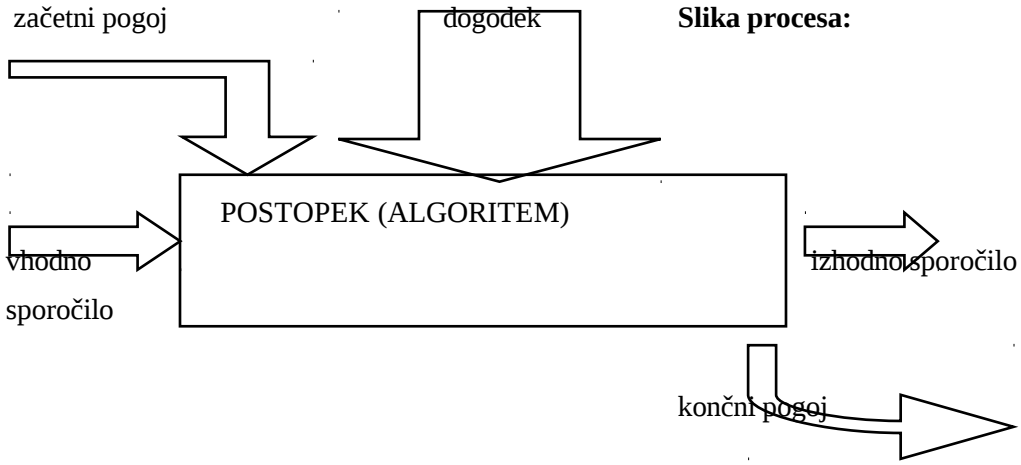
Postopek napram procesu: izraza sta sinonima, uporabljamo ju v hierarhičnem smislu, tako da je proces širši pojem, ki je sestavljen iz posameznih postopkov.



Z modeliranjem postopkov skušamo doseči naslednje cilje:
1. postopkovni modeli nam omogočajo boljši vpogled v strukturo in delovanje posameznih poslovnih funkcij ali sistema kot celote;
2. omogočajo lažjo komunikacijo med načrtovalci sistema ter uporabniki, ki delajo na obravnavanem področju;
3. omogočajo razstavljanje poslovnih funkcij oziroma delov poslovnega sistema na manjše sklope vse do elementarnih postopkov, kar razrešuje problem kompleksnosti, s katerim se običajno srečujemo pri načrtovanju IS in ki smo ga nekajkrat že omenili;
4. omogočajo opredelitev ključnih značilnosti načrtovane računalniške rešitve.

Modeliranje postopkov ima več funkcij:
1. delo ki ga moramo nujno opraviti, če želimo razviti IS za obravnavano področje
2. poslovanje pogledamo s kritičnim očesom in identificiramo odvečne postopke
3. je osnova za racionalizacijo poslovanja

Modeliranje postopkov je vezano na svoje modelirne koncepte, to so:
Postopek, operacija, dogodek, začetni_ pogoj, končni_ pogoj, sporočilo itd.



PROCES je strukturna množica aktivnosti, katerih rezultat je nek proizvod ali storitev s tržno vrednostjo. Zajema vhode in izhode, ki predstavljajo neko dodano vrednost za uporabnike. Praviloma se sestoji iz več postopkov in posega na več funkcijskih področij.

POSTOPEK predstavlja smiselno zaključeno množico operacij na podatkih, s katerimi se sistem pripelje iz enega v drugo stanje. Opredeljen je z algoritmom in vhodno-izhodnimi podatki.
Postopek je temeljni koncept, s katerim modeliramo dinamiko IS. Postopki so lahko **elementarni** ali **kompleksni** (sestavljeni postopki). Kompleksni postopki nastopajo praviloma na višjih hierarhičnih ravneh delovnega sistema. Izvedba postopka povzroči, da sistem preide iz enega stanja v drugo, hkrati pa pomeni tudi uresničitev neke naloge, ki je sestavni del določene poslovne funkcije. Naloga je opredeljena z algoritmom, to je množico logično povezanih pravil, ki opredeljujejo operacije po podatkih.

OPERACIJA je poseg na primerku entitete, s katerim se primerki čitajo, kreirajo, spreminjajo in brišejo, ali na vrednostih njegovih atributov, s katerim se te vrednosti spreminjajo. Operacije delimo na **elementarne** - nanašajo se na primerke entitet (kreiranje, čitanje, spreminjanje, brisanje) in **procesne –** nanašajo se na vrednosti atributov entitet. Operacije se izvajajo samo v okviru postopkov.

DOGODEK je neke vrste impulz, do katerega pride bodisi zaradi sprememb ali posegov iz okolice IS ali zaradi notranjega stanja IS. Povzroči izvedbo enega ali več postopkov, s katerimi se spreminja stanje sistema.

2 vrsti dogodkov:
- tisti, ki nastopajo od zunaj (eksterni); povzroča jih okolica
- tisti, ki so rezultat nečesa, kar se je zgodilo znotraj sistema (interni)

Dogodek sproži nek postopek, proces. Noben postopek se ne sproži sam od sebe. Za vsakim postopkom stoji računalniški program, ki se ne bo sprožil, če ne bo nekega impulza. Vedeti moramo zakaj in kdaj se bo nek postopek izvedel.
Eksterni dogodki – kažejo nek impulz od zunaj.
Interni dogodki – npr. prekoračenje limita – sproži se postopek o prekoračenem limitu – izpisa opomina. To je notranji dogodek, ki je nastal kot posledica stanja v katerem se je sistem znašel.

ZAČETNI POGOJ opredeljuje vse pogoje, ki morajo biti izpolnjeni, da je zagotovljen pravilen potek izvajanja postopka. Začetni pogoj opredeljuje vse pogoje, ki morajo biti izpolnjeni, da se nek postopek začne in da je zagotovljeno pravilno izvajanje postopka.

Z začetnimi pogoji se opredeli naslednje:

1. katera vhodna sporočila morajo biti dana, da se izvajanje postopka lahko začne
2. vrednost atributov entitet, ki jih vsebujejo vhodna sporočila, ali ujemanje med temi vrednostmi in vrednostmi, shranjenimi v podatkovni bazi.

Funkcija opredelitve začetnega pogoja: da zagotavlja vse pogoje, da se bo nek postopek normalno odvijal in normalno končal – to lahko ugotovi končni pogoj, s katerim preverjamo ali se je postopek normalno zaključil.

KONČNI POGOJ, z njim ugotavljamo ali je bil nek postopek izvršen do konca in izvršen po pravilih tako, da ga štejemo za veljavnega. Funkcija je tudi v tem, če ugotovimo, da nek postopek ni bil izveden v skladu s pravili, ga lahko razveljavimo in ga vrnemo na začetek izvajanja postopka. Končni pogoj opredeli vse pogoje, ki morajo biti izpolnjeni po izvedbi procesa, da se lahko šteje, da je bil proces pravilno izvršen in normalno zaključen ter, da je zagotovljena integriteta IS.
Je varovalni mehanizem, ki pokaže ali je bil postopek izvršen na pravilen način.
Funkcija končnega pogoja: je v tem, da preprečuje nekonsistenco v podatkih – zagotavlja integriteto podatkov.
S končnimi pogoji se najpogosteje opredeli naslednje:
1. izhodna sporočila, ki morajo nastati kot rezultat izvedbe postopka
2. razmerja med vrednostmi atributov entitet, ki jih vsebujejo izhodna sporočila
3. vrednosti atributov entitet, shranjenih v bazi podatkov po izvedbi postopka
4. stanje baze podatkov po izvedbi postopka

SPOROČILO je zaključena celota informacij, ki vstopa v postopek ali izstopa iz njega. Je predmet obdelave v postopku in je potrebno za njegovo izvedbo, ali pa je rezultat njegove izvedbe.
Sporočilo je koncept, ki omogoča modeliranje informacijskih tokov med:
- postopki IS,
- IS in njegovo okolico.

Glede na postopke so sporočila lahko **vhodna** ali **izhodna**. S stališča IS kot celote pa lahko ločimo sporočila še na **interna** in **eksterna.** Interna sporočila omogočajo komunikacijo med postopki IS, eksterna pa med IS in njegovo okolico.

ELEMENTARNI IN SESTAVLJENI POSTOPKI

Postopke lahko obravnavamo na različnih ravneh kompleksnosti. Večinoma so sestavljeni iz podpostopkov ali elementarnih postopkov. Postopke razstavljamo s pomočjo funkcijske dekompozicije, dokler ne pridemo do elementarnih postopkov. **Elementarni postopek** je najmanjša za uporabnika smiselna aktivnost, katere izvedba predstavlja zaključeno celoto in je opredeljena z vhodno/izhodnimi podatki ter izvedbenimi pravili (algoritmi). Do elementarnih postopkov pridemo tako, da jih razstavljamo s pomočjo funkcijske dekompozicije.El. postopek je tisti postopek, ki predstavlja smiselno celoto in ga ni smiselno razstavljati naprej. El. postopke moramo smiselno analizirati, da lahko v naslednjih fazah gradnje IS avtomatiziramo postopke.

3.3.2 METODE IN TEHNIKE MODELIRANJA POSTOPKOV

Za modeliranje postopkov uporabljamo različne grafične tehnike.

Metode in tehnike, ki jih podpira večina sodobnih CASE orodij so:

- strukturni graf (structure chart)
- diagram toka podatkov (data flow diagram)
- prehodni graf (state transition diagram).

Za podrobnejši opis postopkov pa je potrebno še dvoje:

- sistemska knjižnica,
- specifikacija postopkov.

Dekompozicija sistema

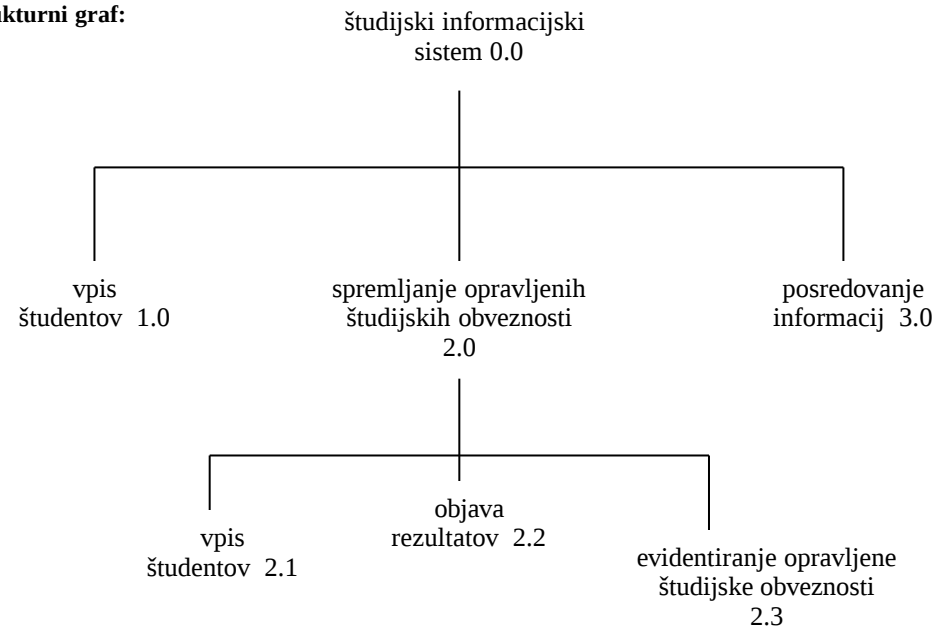
Reševanje kompleksnih problemov je možno le tako, da jih postopoma razstavimo na manjše in manjše ter s tem tudi na preglednejše in obvladljivejše celote. Zaporedna uporaba te metode se imenuje **funkcijska dekompozicija** in nas pripelje do hierarhične strukture sistema, katerega zaključki vej predstavljajo elementarne dle sistema, ki jih je mogoče podrobno obravnavati.

Osnovni cilj je razstaviti sistem na elementarne postopke, ki predstavljajo logično zaključene celote, ki jim je mogoče določiti vhode, izhode in pravila njihovega izvajanja.

STRUKTURNI GRAF je drevesni (hierharični) graf. je temeljna tehnika s katero predstavimo strukturo posameznih procesov iz katere so razvidni vsi postopki, ki na nekem področju nastopajo in njihova medsebojna hierarhija. Strukturni graf nam predstavlja vse postopke, ki nam nastopajo na področju za katerega načrtujemo IS, ter njihove medsebojne povezave, strukturo oziroma njihovo hierarhijo. Do hierarhičnega grafa pridemo po metodi funkcijske dekompozicije.

funkcijska dekompozicija - je proces, postopek, pristop povečanja obravnavanega poslovnega sistema, pri čemer začnemo od zgoraj navzdol pri čemer razstavljamo celoten poslovni sistem (proces) na manjše in manjše dele, dokler ne pridemo do celot, ki jih ni več mogoče naprej sestaviti torej do elementarnih postopkov.

Strukturni graf:



ELEMENTARNI POSTOPKI – postopki, s katerimi se IS vzdržuje. Elementarni postopki so tisti, ki so na koncu posameznih vej strukturnega grafa. Predstavljajo zaključeno celoto medsebojno povezanih aktivnosti oziroma operacij, ki je ni mogoče in ki je ni smiselno razstavljati na manjše dele. Vsi postopki znotraj strukturnega grafa morajo biti enolično označeni. Ena od možnosti je, da postopek, ki je na vrhu označeno z 0.0 in potem naprej označujemo z: 1.0, 2.0, 3.0, ... Vsi postopki na koncu vej so elementarni.

DIAGRAM TOKA PODATKOV – je druga temeljna tehnika, s katero skušamo čimbolj podrobno predstaviti podatkovne oziroma informacijske tokove v okviru posameznega postopka oziroma procesa in pa med njimi. Je zelo razširjena tehnika, ki ni standardizirana. Gre za karakteristične simbole iz katerih se diagram sestoji.

Diagram toka podatkov omogoča prestavitev naslednjih elementov sistema:

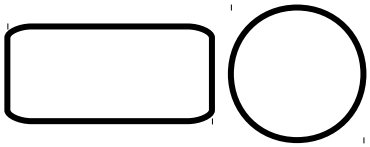
1. postopkov

2. tokov podatkov
3. zbirk podatkov
4. zunanjih entitet

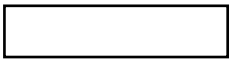
Postopek: Osnovna komponenta diagrama toka podatkov (DTP) je postopek, ki je običajno predstavljen s krogom ali ovalom. Pod postopkom razumemo del modeliranega sistema, ki izvaja neko aktivnost, s katero se transformira vhode v izhode. Vsak postopek ima svoje ime.
Višje v hierarhiji so postopki bolj sestavljeni (kompleksni) kot nižji v hierarhiji.
Vsi postopki,ki nastopajo na koncu vej so elementarni postopki, ki se bodo kasneje avtomatizirali (postopek, ki ni nadaljnje razstavljen je elementarni postopek). Vsi postopki so oštevilčeni.
Če želimo postopek razstaviti, ga moramo razstaviti na najmanj dva postopka.
Zbirka podatkov se uporablja za modeliranje podatkovnih zbirk, ki nam nastajajo v okviru modeliranega IS. Zbirka podatkov je preko podatkovnih tokov vedno povezana s procesi. Podatkovni tokovi, ki vodijo v zbirke podatkov, izstopajo iz postopkov. Tokovi ki vodijo iz zbirke podatkov pa vodijo v postopke. Torej iz tega sledi, da postopki niso povezani med seboj ampak so povezani preko zbirk podatkov. Zbirka podatkov je v diagramu toka podatkov predstavljena z dvema vzporednima črtama.
Zunanja entiteta: koncept zunanje entitete se bistveno razlikuje od koncepta entitete, ki smo ga doslej poznali. Z zunanjo entiteto skušamo predstaviti vse zunanje akterje, tako fizične kot pravne osebe, s katerimi naš načrtovani sistem komunicira. Pogosto so zunanje entitete s katerimi povzročajo dogodke, ki sprožijo postopke.
Od zunanjih entitet prihajajo vhodni podatki/dokumenti, vanje se pa stekajo izhodni podatki oziroma rezultati sistema. Zunanje entitete so ponavadi fizične ali pravne osebe s katerimi naš sistem komunicira in ki pogosto povzročijo izvajanje postopkov.
!!!! zunanje entitete ne smemo zamenjati z entiteto pri podatkovnem modelu!!!!!!
Tok podatkov: Podatkovni tokovi so v DTP predstavljeni s puščicami, ki so usmerjene v postopke ali pa iz njih. Pod tokom podatkov razumemo prenos oziroma potovanje podatkov/dokumentov med različnimi deli sistema. Mora imeti puščico, da se vidi smer v katerega dokumenti potujejo; če ima obe puščici podatki tečejo v obe smeri. Puščice torej predstavljajo sporočila, ki potekajo med postopki oziroma med postopkom in zunanjim svetom.
Informacijski tokovi med procesom in zunanjo entiteto so poimenovani, med procesom in zbirko podatkov pa niso poimenovani.

Slika osnovnih simbolov diagrama tokov podatkov:

Postopek,proces, aktivnost (ki se odvija v okviru IS katerega modeliramo)



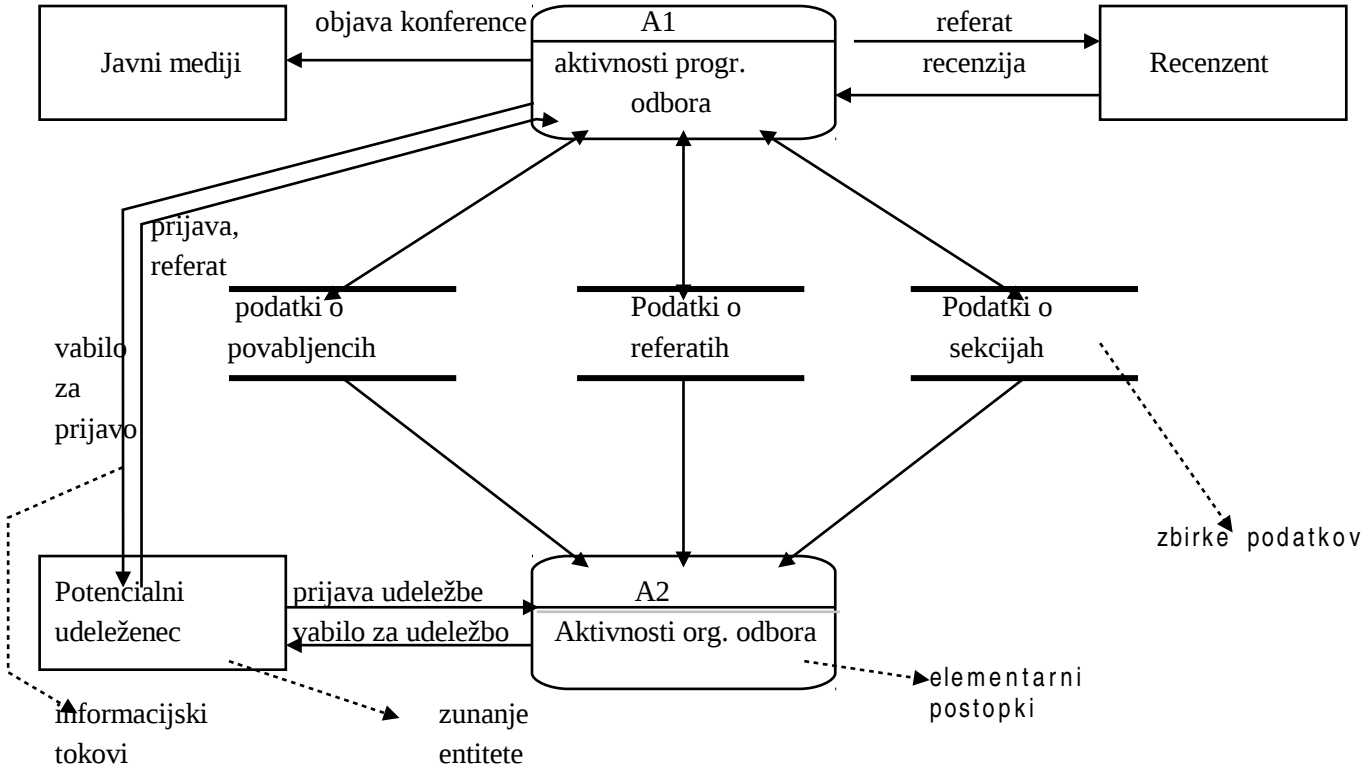
Zunanja entiteta



Zbirka podatkov

Tok podatkov (kaže s puščicami kako so simboli povezani)

Slika diagrama tokov podatkov:

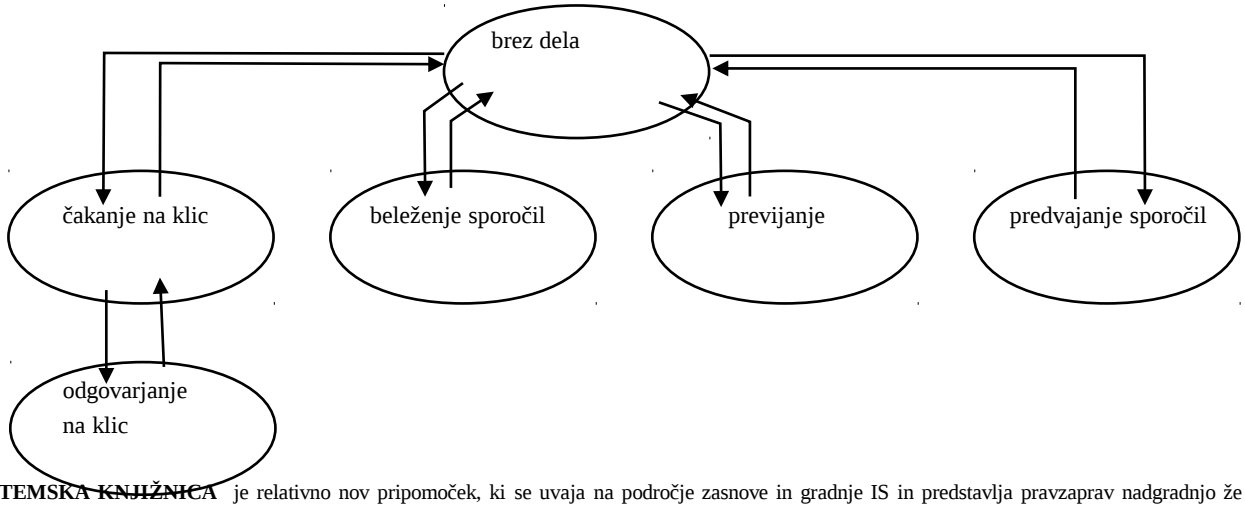


PREHODNI DIAGRAM – se ne uporabljajo tako pogosto kot diagram toka podatkov. Prav nam pride kadar je pomembno vedeti v katerih obravnavanih sistemih se lahko sistem znajde. Predstavlja nam možna stanja OS in zaporedja njihovih postopkov. Medtem ko nam DTP omogoča modeliranje postopkov oziroma podatkovnih tokov med njimi, pa nam prehodni diagrami omogočajo modelirati obnašanje sistema v odvisnosti od časa in prikaz dovoljenih stanj, v katerih se sistem lahko znajde.

Prehodni diagrami postanejo zelo pomembni kadar želimo s pomočjo načrtovanega IS zagotavljati ne le vse potrebne informacije ampak želimo tudi vpeljati upravljanje postopkov.

Prehodni diagram kaže vsa možna stanja v katerih se lahko znajde določen sistem in prehode med temi stanji.

Slika prehodnega diagrama za sistem elektronske tajnice:



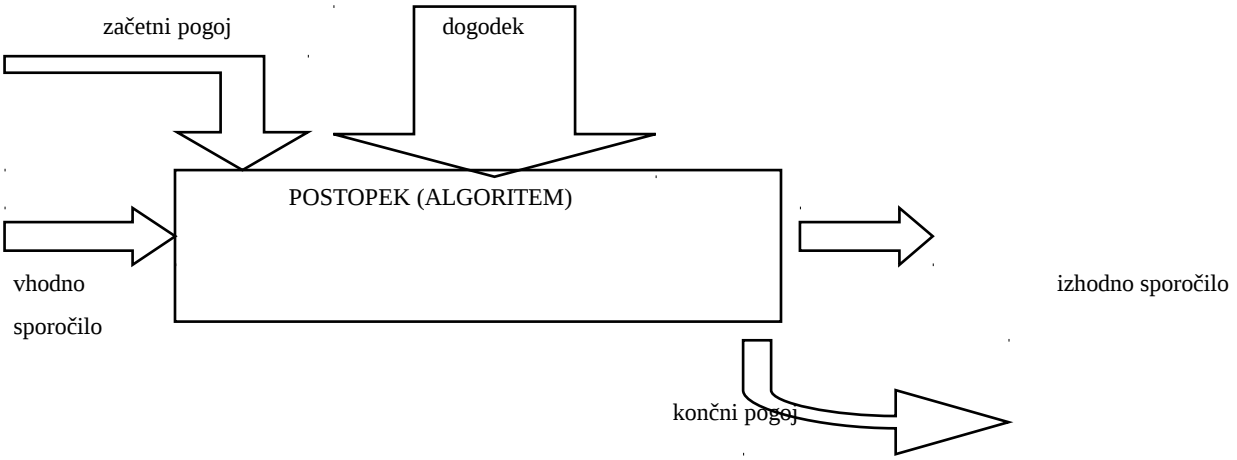
SISTEMSKA KNJIŽNICA je relativno nov pripomoček, ki se uvaja na področje zasnove in gradnje IS in predstavlja pravzaprav nadgradnjo že dolgo uveljavljenih podatkovnih slovarjev. Za razliko od slovarja podatkov, ki po definiciji vsebuje formalne opise podatkov, ki nastopajo v bazi podatkov IS, so funkcije sistemske knjižnice naslednje:

Funkcije sistemske knjižnice:

- vzdrževanje vseh opisov podatkov, ki sestavljajo bazo podatkov sistema (katalog podatkov);
- vzdrževanje vseh tekstovnih in grafičnih opisov poslovnih funkcij, sestavljenih in elementarnih postopkov, itd (opisi grafov);
- vzdrževanje vseh opisov povezav med postopki in podatki;
- vzdrževanje opisov vhodno-izhodnih mask in poročil.
- Podrobno opredeljeni postopki (algoritem,dogodek, začetni in končni pogoj (slika spodaj)

Grafične tehnike predstavijo procese do neke mere, dokumentiramo pa te procese v sistemski knjižnici. Sistemska knjižnica naj bi omogočala sistematično vzdrževanje in vodenje vseh podatkov o IS.

Sistemska knjižnica ni samo orodje za podporo modeliranju postopkov, temveč univerzalno orodje za podporo celotnemu razvojnemu procesu IS.



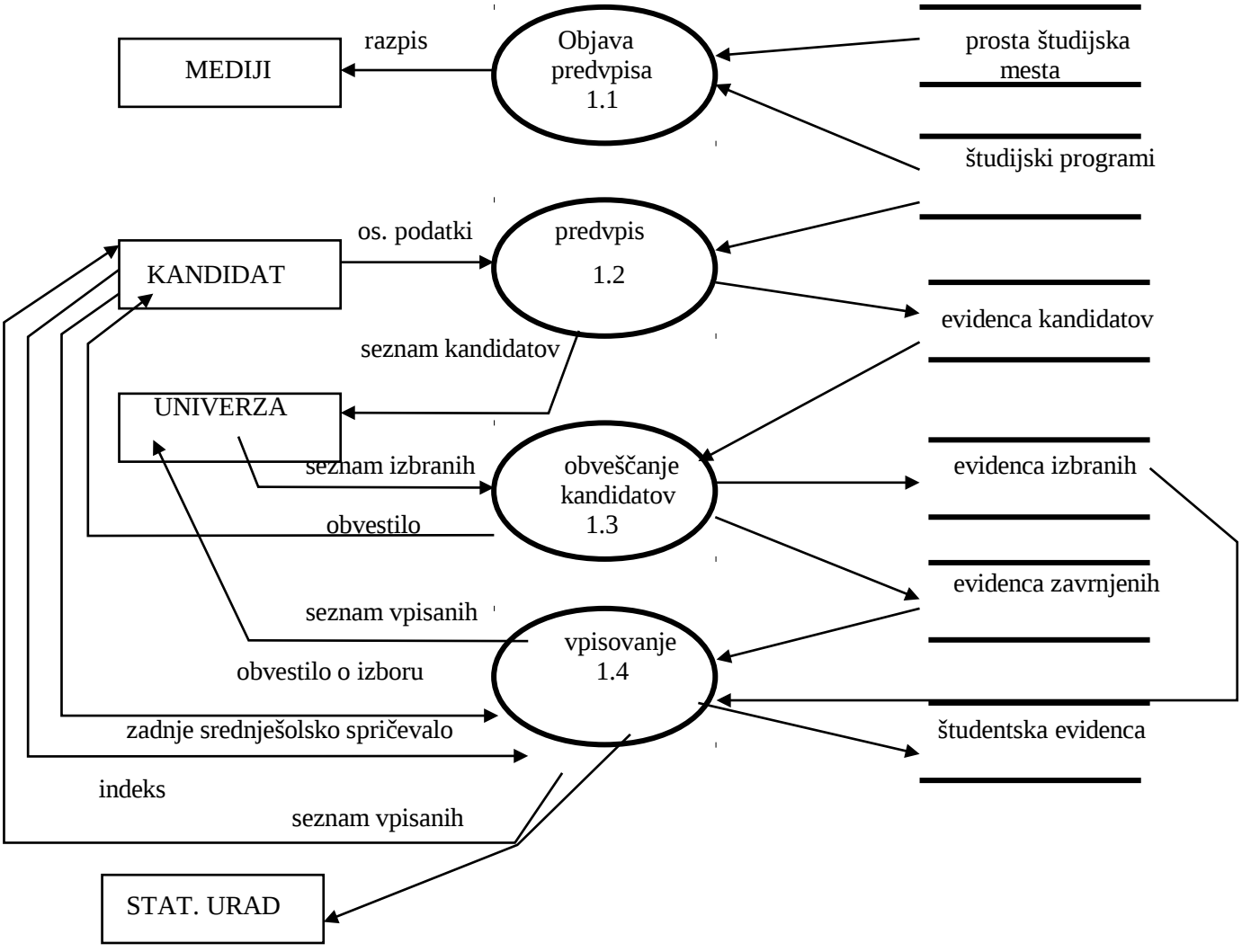
Opredelitev postopkov:

Za podrobno opredelitev elementarnega postopka je potrebno opredeliti naslednje:

1. pravila njegovega izvajanja (algoritem)
2. dogodek, ki sproži izvajanje procesa
3. začetne pogoje
4. končne pogoje
5. specifikacijo vhodno izhodnih/sporočil

Razvoj postopkovnega modela
Primer:

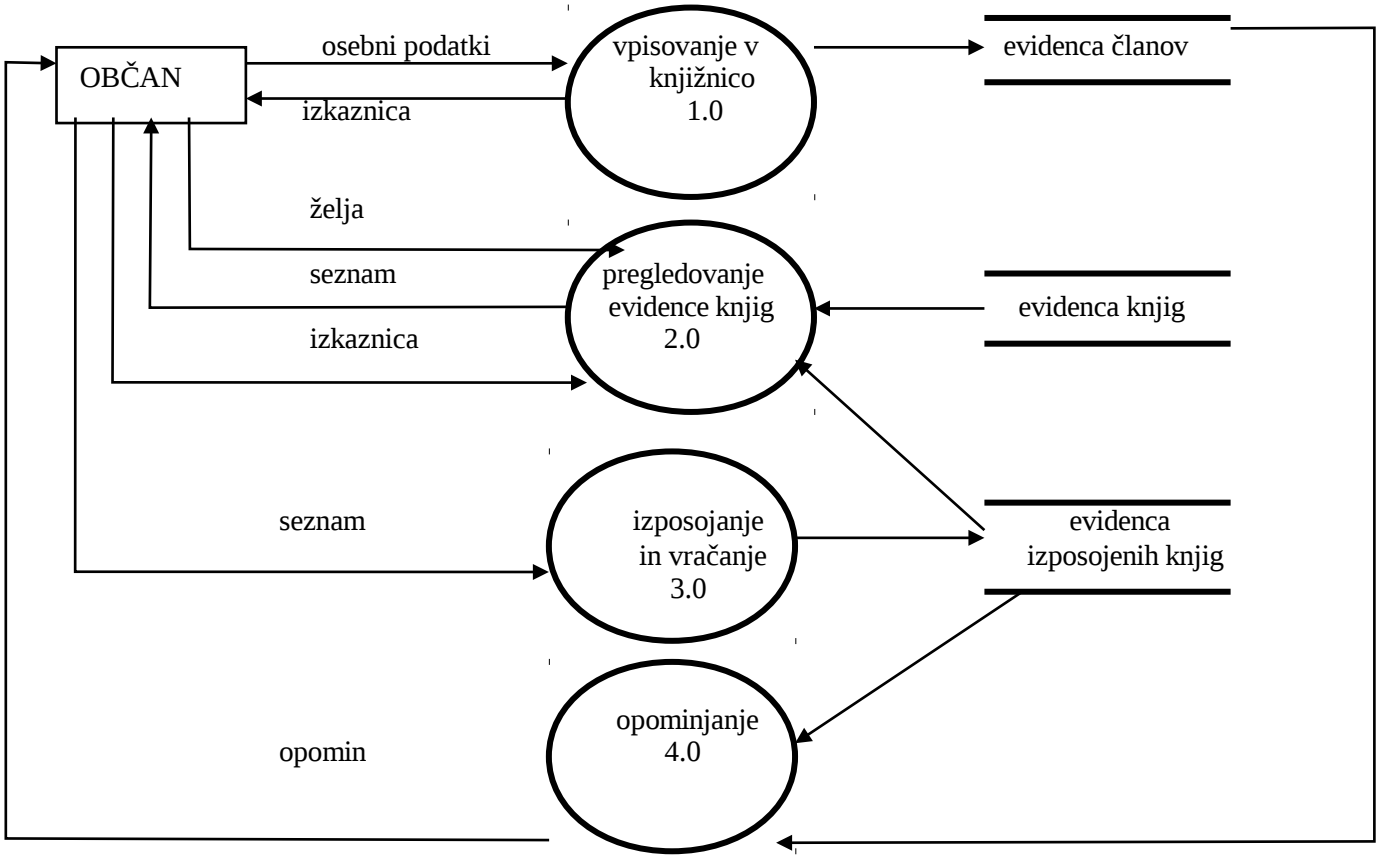
Vpis na univerzo:
1. VUŠ na podlagi prostih mest in na podlagi študijskih programov objavi predvpis v medijih.
2. Kandidati pošljejo na predvpis osebne podatke. Šola pošlje seznam kandidatov na Univerzo.
3. Univerza izdela izbor, ki ga pošlje šoli. Šola obvesti kandidate o izboru.
4. Kandidati pri vpisu predložijo obvestilo o izboru in zadnje spričevalo. Seznam vpisanih pošlje šola na Univerzo in Zavod za statistiko.

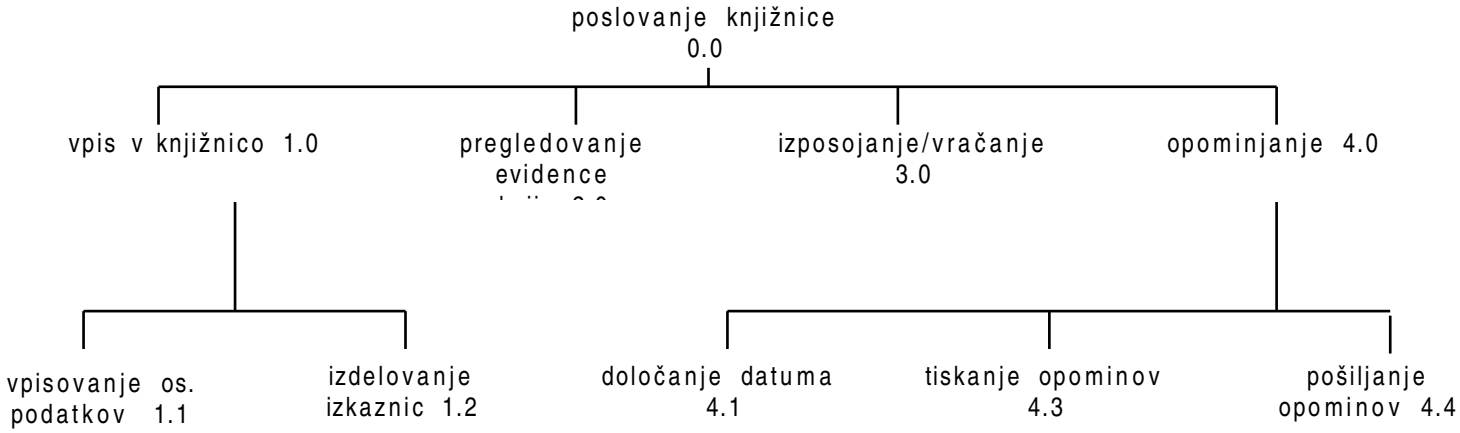


Primer:

Poslovanje knjižnice

1. Občan se vpiše v knjižnico tako, da poda svoje osebne podatke. Prejme izkaznico.
2. Člani pregledujejo sezname knjig. Iščejo po avtorju, naslovu, zvrsti
3. Izbrane knjige si izposojajo in vračajo. Knjižnica vodi evidenco izposoje in vračil.
4. Knjižnica pošilja članom opomine.





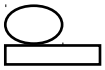
Analiza informacijskih potreb:
Tu opredelimo podatke, ki so potrebni za delovanje sistema.

Primer za knjižnico:
1. (prva informacijska potreba) – Osebni podatki: ime, priimek, pošta, rojstni datum, EMŠO
2. Evidenca članov: osebni podatki, št. izkaznice, datum vpisa, datum plačila članarine
3. Plačilo: št. izkaznice, znesek, namen, datum plačila, način plačila
4. Evidenca knjig: avtor, naslov, zvrst, leto izdaje, založba, št. strani, format, št. knjige, čas izposoje
5. Evidenca izposoje: št. knjige, št. izkaznice, datum izposoje
6. Izkaznica: št. izkaznice, ime, priimek, datum izdaje
7. Želja: avtor, zvrst, naslov
8. Seznam: št. knjige, avtor, naslov, nahajališče knjige
9. Opomin: znesek, rok. plačila, ime, priimek, naslov, razlog za plačilo, naslov knjige, avtor, datum izposoje, rok vrnitve

SEMINARSKA NALOGA (primer)
Namen:
- praktična uporaba teorije
- sposobnost sodelovanja z informatiki
- osnovne gradnje in načrtovanja informacijskih sistemov
- materializacija teorije organizacije, reinženiring

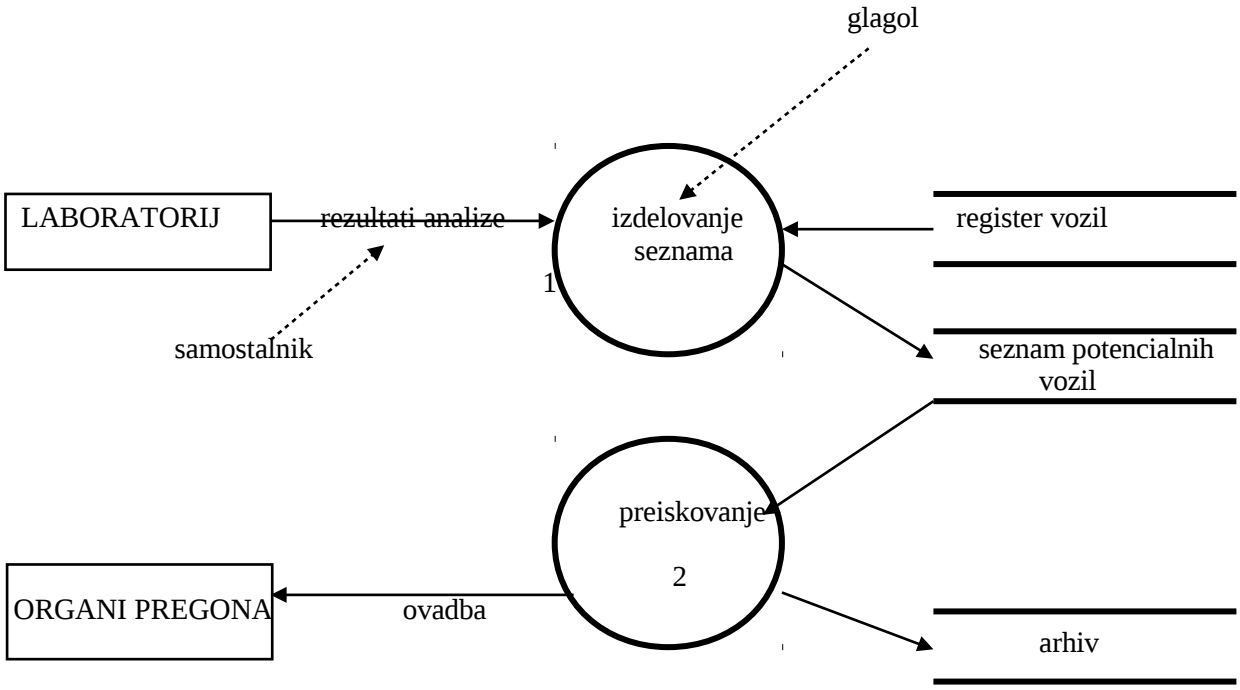
Trije pogledi na organizacijo:
- hierarhični diagram
- procesna razgradnja
    - strukturni diagram
    - diagram pretoka podatkov
- podatkovna razgradnja
    - E/R diagram
    - slovar entitet, povezav
    - slovar atributov

Oblika seminarske naloge:
1. predstavitev sistema
2. Informacijske potrebe (popis dokumentov in kaj vsebujejo)
3. Hierarhični diagram (direktor, pomočnik direktorja …)
4. E/R model
5. slovar entitet, povezav in atributov
6. strukturni diagram
7. diagram toka podatkov



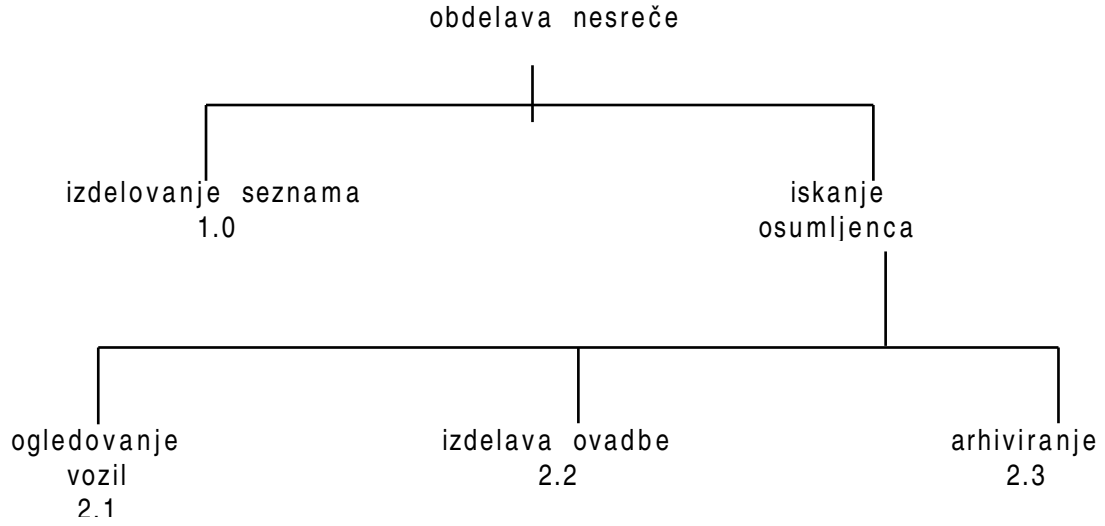
primer:
Predstavitev sistema nesreče s pobegom
1. policijski laboratorij preko analize ostankov nesreče ugotovi barvo, tip in letnik vozila
2. informacijski sistem poda seznam vozil te vrste
3. s preiskavo na terenu odkriti osumljenca

Organigram (diagram)
Narišeš strukturo MNZ
diagram toka podatkov:
- je namenjen modeliranju postopkovnega dela IS
- prikazuje dinamičnost IS in zaporedje izvajanja postopkov
- omogoča predstavitev štirih elementov sistema:
    - postopek
    - zbirka podatkov
    - zunanja entiteta
    - tok podatkov



Pogoste napake v DTP:
1. DTP ni organizacijska struktura
2. procesi morajo biti imenovani z glagolom, tokovi pa s samostalnikom
3. zunanje entitete ne smejo biti medsebojno povezane, enako velja za zbirko podatkov in procese
4. znanja entiteta mora biti povezana s procesom
5. vsak proces mora vsebovati vhodne in izhodne tokove
6. diagram ne sme biti prekinjen
7. zunanje entitete niso izvajalci samega postopka

Strukturni graf:
1. je namenjen modeliranju postopkovnega dela IS
2. ne prikazuje dinamičnosti IS, temveč le hierarhično strukturo postopkov
3. vozlišča predstavljajo postopki obravnavanega IS
4. loki prikazujejo strukturo postopkov IS
5. do grafa pridemo s pomočjo funkcijske dekompozicije
6. dekompozicija pripelje do razmerja sestoji_iz med nadrejenim vozliščem in podrejenimi vozlišči
7. zaključki vej so elementarni postopki



3.4 PROBLEMI PREHODA IZ LOGIČNE NA FIZIČNO RAVEN MODELIRANJA IS

Najpomembnejša šibka točka pravzaprav vseh doslej znanih modelov je njihova omejena uporabnost skozi različne faze razvoja IS. Najostrejša je meja med logično ter fizično ravnijo modeliranja podatkov.
Modeli ki jih uporabljamo na logični ravni (danes je to predvsem model entiteta – povezava), niso direktno uporabni na izvedbeni ravni, saj trenutno še nimamo krmilnega sistema baze podatkov (KSBP), ki bi temeljil na E-R modelu. Velja pa tudi obratno, da modeli na katerih temeljijo današnji najbolj razširjeni komercialni produkti krmilnega sistema baze podatkov (relacijski, hierarhični ...), niso prikladni za modeliranje podatkov na logični ravni, saj ne omogočajo dovolj jasnega vpogleda v semantiko podatkov, ki nam nastopajo v okviru obravnavanega sistema.

Model ki ga naredimo v drugi fazi (slika stran 11.) to je logični model, je potrebno v nadaljevanju pretvoriti v fizični model. E-R model moramo pretvoriti v relacijski model in razviti bazo podatkov.

Pri razvoju konkretne baze podatkov smo zato običajno prej ali slej prisiljeni zapustiti udobje E-R modela in ta model pretvoriti v model, ki ga podpira izbrani krmilni sistem baze podatkov. Pri tem smo soočeni z dvema načeloma:

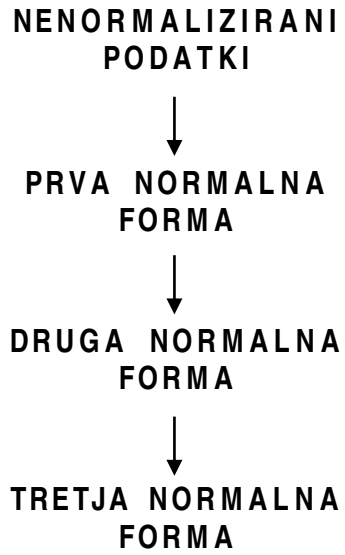
1. model podatkov, ki ga bomo dejansko izvedli s pomočjo izbranega krmilnega sistema baz podatkov, naj bi bil tak, da bo zagotavljal dolgoročno zadovoljevanje vseh informacijskih potreb uporabnikov in da ga bo potrebno čim manj spreminjati ne glede na vrsto uporabljene strojne in programske opreme (**kriterij stabilnosti**).
2. model podatkov, na osnovi katerega zgradimo konkretno bazo podatkov obravnavanega informacijskega sistema, mora zagotavljati čim bolj učinkovito delovanje baze podatkov (dodajanje novih podatkov, spreminjanje, iskanje ...); tu gre za **kriterij učinkovitosti ali performančni kriterij**

Prvo načelo upoštevamo tako, da postopno transformiramo naš logični model podatkov v fizični, to je izvedbeni model, na način, ki nas bo pripeljal do optimalnih ter čim bolj stabilnih podatkovnih struktur za izvedbeni model, ki ga podpira izbrani krmilni sistem baze podatkov. Temu postopku pravimo **normalizacija.**

NORMALIZACIJA
so postopki pretvorbe iz logične na fizično raven. Normalizacija poteka v treh korakih, od katerih nas vsak privede do normalne forme. Ko so podatki v tretji normalni formi, so direktno izvedljivi v relacijskem modelu. Problem pretvorbe iz logične na fizično raven je kardinalnost M:N, ki na logični ravni ni problematična, na fizični ravni pa moramo povezavo M:N spremeniti v povezavo 1:N.
Postopek normalizacije nas torej vodi skozi tako imenovane **normalne forme.**

Nenormalizirani podatki
Ko začnemo normalizirati podatke v okviru obravnavanega informacijskega sistema, nas v začetni fazi zanima predvsem opredelitev vseh pomembnejših tipov entitet in njihovih medsebojnih povezav. Ko govorimo o nenormaliziranih podatkih govorimo o podatkih v logičnem modelu, torej so podatki v logičnem modelu nenormalizirani podatki.
E-R model, ki je na logični ravni povsem sprejemljiv in razumljiv razvijalcem in uporabnikom, pa običajno direktno ni uporaben za konkretno izvedbo, denimo relacijske baze podatkov. Zato ga je potrebno normalizirati.
Normalizacijo podatkov lahko obravnavamo kot večfazni postopek, skozi katerega postopoma razstavljamo ter odpravljamo slabe relacije v enostavnejše, ki ustrezajo kriterijem normalizacije, dokler ne dobimo vseh relacij, ki ustrezajo pogojem tretje normalne forme. Osnovne korake postopka normalizacije prikazuje spodnja slika.

Shema postopka normalizacije



1. izločitev vseh ponavljajočih se skupin atributov v samostojne relacije. (ko to opravimo pridemo v prvo normalno formo)

2. V relacijah s sestavljenimi ključi je potrebno zagotoviti, da so vsi ostali atributi odvisni od celotnega ključa. (ko to opravimo pridemo v drugo normalno formo)

3. Odstraniti vse prehodne odvisnosti med atributi ter po potrebi oblikovati nove relacije. (ko to opravimo pridemo v tretjo formo)

Opis slike:
Izhajamo iz E-R modela, ki je nastal v fazi logičnega modeliranja podatkov in ki je običajno nenormaliziran. Vsak ti entitete nam v tej fazi predstavlja eno nenormalizirano relacijo.
V prvem koraku, ki nas pripelje do modela v prvi normalni formi (1NF), moramo odstraniti iz relacij tako imenovane ponavljajoče se skupine atributov ter z njimi tvoriti nove relacije.
V drugem koraku se osredotočimo na tiste relacije, katerih ključi so sestavljeni iz več atributov, in iz njih izločimo vse tiste atribute, ki niso odvisni od celotnega sestavljenega ključa relacije in z njimi ponovno oblikujemo nove relacije in s tem pripeljemo model v drugo normalno formo. (2NF)
V tretjem koraku pa odstranimo še tako imenovane prehodne ali tranzitivne odvisnosti med atributi in tudi tu po potrebi oblikujemo nove relacije in s tem pripeljemo model v tretjo normalno formo (3NF).

Funkcionalne odvisnosti
Za lažjo predstavitev druge in tretje normalne forme moramo na spregovoriti še o tako imenovanih funkcionalnih odvisnostih.
Funkcionalne odvisnosti se nanašajo na odvisnosti med vrednostmi posameznih atributov v relaciji, kar lahko opredelimo takole:
Vrednost atributa B v relaciji R je funkcionalno odvisna od vrednosti atributa A, če v vsakem trenutku pripada vsaki vrednosti atributa A ena sama vrednost atributa B. Rečemo lahko tudi, da A identificira B, kar pomeni, da če poznamo vrednost atributa A lahko vedno najdemo tudi pripadajočo vrednost atributa B.
V slikah 3.37 – 3.39 so funkcionalne odvisnosti med atributi prikazane s puščicami. V relaciji NAROČILO je atribut Datum funkcionalno odvisen od atributa Šifra_naročila. Datum naročila bomo običajno iskali prek Šifre_naročila. Z vsako vrednostjo Šifre_naročila je povezana ena sama vrednost Datum_NAROČILA. Obratno pa seveda ne drži, saj preko vrednosti atributa Datum ni mogoče zanesljivo določiti vrednosti atributa Šifra_naročila, saj imamo lahko celo množico naročil, ki smo jih prejeli na isti datum. To pomeni da atribut Šifra_naročila ni funkcionalno odvisen od atributa Datum.
V normaliziranih relacijah zahtevamo, da so vsi atributi relacije funkcionalno odvisni od primarnega ključa.
Stvar se zaplete, kadar je primarni ključ sestavljen iz več atributov, to je primeru spetih ključev, zato sta v strokovni literaturi vpeljani tako imenovana **popolna funkcionalna odvisnost in delna funkcionalna odvisnost.** Popolna funkcionalna odvisnost pomeni, da so vrednosti atributa B v relaciji R popolno funkcionalno odvisne od sestavljenega ključa, ki se sestoji iz množice atributov A, niso pa popolno odvisne od podmnožice množice A.

Prva normalna forma:

Prva normalna forma prepoveduje večvrednostne atribute, sestavljene atribute ali njihove kombinacije. 1NF torej zahteva, da smejo domene atributov vsebovati samo atomarne vrednosti in da so vsi atributi v relaciji enovrednostni, to pomeni, da lahko zavzemajo eno samo vrednost.
Ponavljajočo se skupino atributov izločimo in dobimo novo relacijo, ki ima sestavljen ključ da je ohranjena povezava.

Primer:
Relacija NAROČILO (slika 3.37a) vsebuje skupino večvrednostnih atributov, kot so: Šifra_artikla, Naziv_artikla, Količina, Cena_artikla in Vrednost_artikla, saj lahko kupec z enim naročilom naroči veliko število različnih artiklov. Relacija NAROČILO ni v prvi normalni formi. Da bo ta relacija izponjevala kriterije prve normalne forme, je potrebno vse večvrednostne atribute izločiti in jih oblikovati v samostojne relacije, kar prikazuje slika 3.37b, kjer so izločeni atributi oblikovani v novo relacijo NAROČILO-ARTIKEL.

Primer je relacija NAROČILO-ARTIKEL na sliki 3.37b.
Atribut Vrednost_artikla je polno funkcionalno odvisen od sestavljenega ključa Šifra_naročila + Šifra_artikla, saj lahko iz vrednosti sestavljenega ključa vedno določimo tudi vrednost atributa Vrednost_artikla, delni ključ pa za to ne zadošča.
Za atribut Cena_artikla pa velja le delna funkcionalna odvisnost, saj za določitev njegovih vrednost zadošča že delni ključ, to je Šifra_artikla.

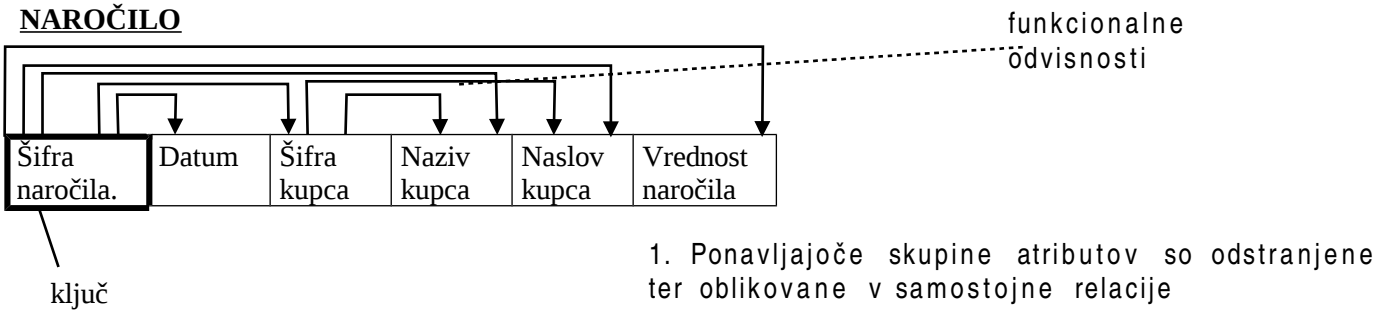
slika 3.37
a. Nenormalizirana relacija

NAROČILO

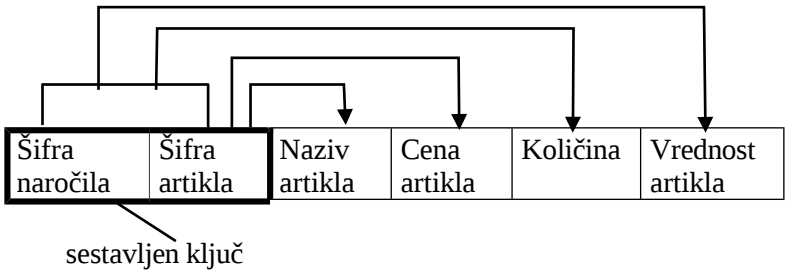
Šifra Naroč.	Datum	Šifra kupca	Naziv kupca	Naslov kupca	Šifra artikla	Naziv artikla	Količina	Cena artikla	Vrednost artikla	Vrednost naročila

b. pretvorba v prvo normalno formo

NAROČILO



NAROČILO-ARTIKEL

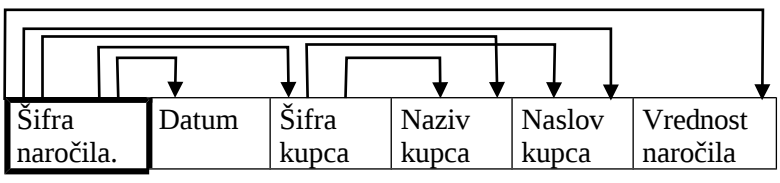


Druga normalna forma:
zahteva, da so vsi atributi relacije R (ki niso ključi) popolnoma funkcionalno odvisni od celotnega primarnega ključa relacije, ne glede na to , ali je le ta sestavljen iz enega ali več atributov.
Pretvorbo opravimo tako, da so vsi atributi odvisni le od celotnega primarnega ključa.

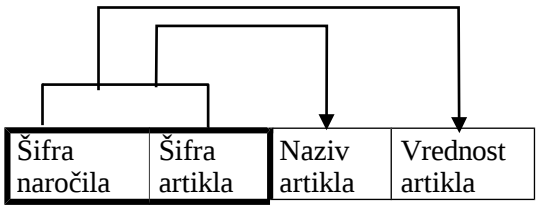
Primer:
Glede na to zahtevo relacija NAROČILO-ARTIKEL na sliki 3.37b ni v drugi normalni formi in jo je potrebno razstaviti na dve relaciji NAROČILO-ARTIKEL in ARTIKEL, kakor je to prikazano na sliki 3.38. Relacije na sliki 3.38 so torej v drugi normalni formi, saj so atributi v vseh relacijah funkcionalno odvisni od celotnega ključa relacije.

slika 3.38
pretvorba v drugo normalno formo

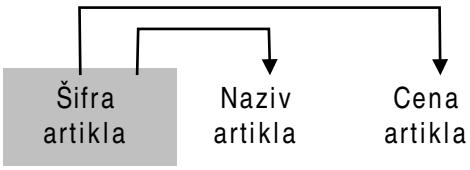
NAROČILO



NAROČILO-ARTIKEL



ARTIKEL



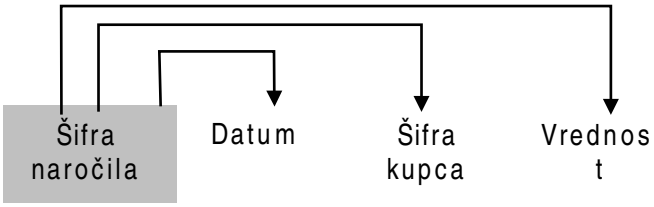
2. Vsi atributi, ki niso bili odvisni od celotnega ključa, kot je bil primer pri relaciji NAROČILO- ARTIKEL so bili preneseni v samostojno relacijo artikel

Tretja normalna forma:
temelji na konceptu prehodnih ali tranzitivnih odvisnosti. Relacije, ki so v 3NF imajo lahko še vedno določene anomalije. Lahko vsebujejo atribute, ki niso ključi ali njihovi deli, pa vendar sami zase identificirajo druge atribute, kar imenujemo **prehodna ali tranzitivna odvisnost.** Tranzitivne odvisnosti lahko povzročajo probleme pri uporabi baze podatkov, zato jih je potrebno odstraniti iz relacij.
Pri pretvorbi v tretjo normalno formo na koncu med atributi ne sme biti medsebojne odvisnosti. Atribut mora biti odvisen le od primarnega ključa in ne od drugega atributa.

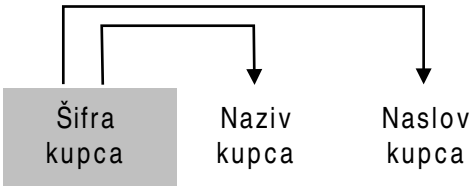
Primer:
Pri relaciji NAROČILO iz slike 3.39 si podrobneje oglejmo funkcionalne odvisnosti. Atributa Naziv_kupca in Naslov_kupca sta funkcionalno odvisna od Šifre_kupca in tranzitivno odvisna od primarnega ključa Šifra_naročila. Tej tranzitivni odvisnosti se izognemo tako, da skupino atributov izločimo in oblikujemo novo relacijo KUPEC.
Rezultat celotnega procesa normalizacije je torej model, prikazan na sliki 3.40, kjer so vse relacije v tretji normalni formi. Tak model je mogoče direktno izvesti s poljubnim krmilnim sistemom baz podatkov, ki temelji na relacijskem modelu.

slika 3.39
PRETVORBA V TRETJO NORMALNO FORMO

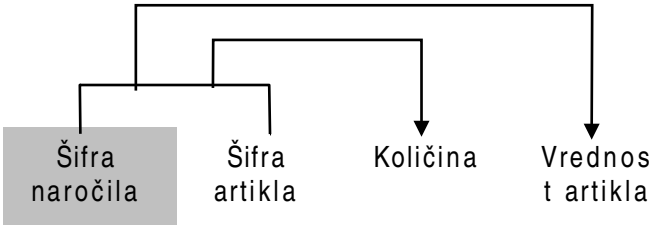
NAROČILO



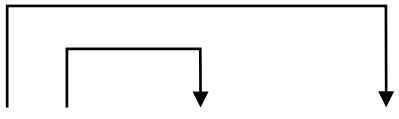
KUPEC



NAROČILO - ARTIKEL



ARTIKEL



3. Vsi atributi, ki so bili odvisni od drugih atributov (ki niso ključi), kot je bil to primer pri relaciji NAROČILO, so odstranjeni ter oblikovani v samostojne relacije

Šifra artikla	Naziv artikla	Cena artikla

TEHNOLOŠKA, ORGANIZACIJSKA IN KADROVSKA IZHODIŠČA (5. Poglavje)

5.1 UPORABA INFORMACIJSKIH ORODIJ

Informacijska orodja so specializirane rešitve, bolj ali manj standardizirane, ki nam olajšajo delo, izvajanje aktivnosti v okviru celotnega razvojnega procesa od analiz, načrtovanja in gradnje IS. Povečajo produktivnost vseh udeleženih v razvojnih timih, omogočajo aktivnosti v posameznih procesih. Informacijska orodja v večini primerov slonijo na programskih jezikih četrte generacije, povezanih z relacijskimi bazami podatkov in sistemskimi knjižnicami.

Uporabnost informacijskih orodij
Največjo odliko krmilnih sistemov baz podatkov, zasnovanih na relacijskih bazah podatkov, predstavlja njihova enostavnost uporabe. Le ta izhaja iz predstavitve podatkov uporabniku na nazoren in lahko sprejemljiv način v obliki tabel.

Značilnosti sodobnih informacijskih orodij:
1. Združljivost (kompatibilnost), ki izhaja predvsem iz usmeritve proizvajalca orodja k uveljavljanju standardizacije.
2. Prenosljivost, kjer je osnovni cilj uporaba informacijskega orodja, ki bo s stališča uporabnika enak ne glede na uporabljeni računalnik ali operacijski sistem.
3. povezljivost s ciljem, da predstavljajo podatki, ki se nahajajo na različnih medsebojno povezanih računalnikih, enotno porazdeljeno bazo podatkov
4. produktivnost, tako v smeri prototipnega razvoja programskih rešitev kot rudi v smeri učinkovitega pridobivanja podatkov za odločanje.

jeziki 3. generacije (razviti v letih 1960 – 1980) so še danes v uporabi (COBOL, FORTRAN ...). Po letu 1980 je pa bila razvita četrta generacija jezikov.

jeziki 4. generacije:
pojav v začetku 80-ih let; Jeziki 4. generacije omogočajo bistveno krajše čase in povečano produktivnost. Ti jeziki so največkrat nepostopkovni, kar pomeni, da z njimi povemo računalniku KAJ naj izvede namesto KAKO naj to izvede. Največkrat zajemajo jezike za vzdrževanje baze podatkov, generatorje zaslonskih slik in poročil...

Prednosti 4. Generacije: - skrajšajo razvojni čas, vendar so rešitve manj učinkovite in potrebujejo več sistemskih delov kot pa rešitve, ki so izdelane z orodji 3. generacije. Jeziki 3. generacije nam omogočajo boljše možnosti pri optimiziranju rešitev, ki zahtevajo veliko sistemskih virov.
Pri rešitvah, kjer ocenjujemo, da bodo razvojni stroški večji od obratovalnih se bomo odločili za 4. generacijo, ki nam znižujejo razvojne stroške. Pri rešitvah, kjer so razvojni stroški manjši od obratovalnih se odločimo za 3. generacijo.

Razvojne smeri 4GL so:
1. nepostopkovni jeziki
2. report generatorji
3. generatorji zaslona
4. poizvedovalni jeziki

Relacijska baza podatkov (bp):
je strukturirana zbirka podatkov, ki so shranjeni na računalniškem mediju in temeljijo na preddefiniranih tipih in povezavah, ki so namenjeni nekemu poslovnemu procesu. To je koncept organiziranja podatkov v okviru sodobnih IS, ki nam zagotavlja splošno uporabnost podatkov, ki omogoča da iste podatke uporablja poljubno število uporabnikov in ki zagotavlja minimalno podvajanje podatkov, visoko stopnjo razvoja in zaščite podatkov.

Imamo dva tipa baz podatkov:
1. namizne baze (za PC, do 5 uporabnikov, to so ACCESS, PARADOX, DBASE, FOX PRO …)
2. produkcijske baze (Oracle, DB2 od IBM, SQL-server, Informix …)

Baza podatkov omogoča varovanje podatkov na naslednje načine:
1. transakcija
   - 1.1 potrditev sprememb
   - 1.2 razveljavitev sprememb
   - 1.3 zaklepanje zapisa
2. avtorizacija
3. obnavljanje sprememb
4. BACKUP/RESTORE podatkov (vsaka sprememba se zabeleži v posebno datoteko)
5. senčenje/podvajanje podatkov (imamo dva diska in v primeru okvare prvega delo prevzame drugi)

Za izgradnjo in za razvoj ter uporabo BP potrebujemo specializirana orodja - **krmilni sistem baz podakov**. Gre za kompleksen sistem programov, ki nam omogočajo vse funkcije v zvezi z razvojem, vzpostavitvijo in uporabo baz podatkov. Vsak KSBP temelji na izbranem podatkovnem modelu . večina jih temelji na relacijskem podatkovnem modelu. KSBP priskrbi za pretvorbo iz fizičnega v logični pogled.
katalogi podatkov so, samostojno ali kot del CASE orodij, informacijska orodja, v katerih se nahajajo vsi opisi in opredelitve podatkov, torej podatki o podatkih, zajetih v bazi podatkov in ostalih podatkih organizacije.
slovar podatkov: Če imamo velike IS, postane vzdrževanje takega slovarja obsežna naloga. Zato so razvili posebna orodja, ki omogočajo vzpostavitev in vzdrževanje slovarjev, ki bistveno zmanjšuje obseg dela, olajša sestavo slovarja in nadzor nad integriteto takega stanja.

5.2 SQL
SQL je s stališča uporabe večnivojski, angleškemu jeziku podoben programski jezik, ratificiran kot standardni jezik za delo s krmilnimi sistemi relacijskih baz podatkov. Vsebuje dva nivoja in sicer nižji nivo (za uporabnike brez izkušenj), višji nivo (za razvijalce IS ter programerje).
SQL je standardizirani jezik za delo s krmilnimi sistemi za baze podatkov..
Je večfunkcijski jezik med programskimi jeziki in edini, ki se ga ni potrebno učiti v smislu pomnenja množice ukazov, saj vsebuje le ukaze za:
1. pridobivanje podatkov iz baze podatkov (Query) – ukaz SELECT
2. vstavljanje, ažuriranje in brisanje podatkov iz baze podatkov – ukazi INSERT, UPDATE in DELETE
3. dodajanje novih tabel v bazo in brisanje tabel iz baze podatkov – ukaza CREATE in DROP
4. nadzor nad uporabo podatkov, ki preprečuje nenadzorovan in nepooblaščen pristop do podatkov – ukaza GRANT in REVOKE

SQL je visokonivojsko nepostopkovni programski jezik, s katerim opredelimo, kaj naj se izvede in kaj ne, kako naj se pripravijo ustrezni algoritmi, da bodo izvedljivi s pomočjo računalnika.
primer:

```
SELECT  reg, ime, priimek, naslov
FROM    vozilo, obcan
```

WHERE vozilo.emso = obcan.emso
AND tip = Z101
AND barva = ze
AND letnik = 1988

5.3.CASE ORODJA: (computer added software engineering-Računalniško podprta gradnja IS)
Nastala so z namenom, da bi celoviteje podprli in do neke mere avtomatizirali razvojni proces IS. CASE, ki je bil še pred desetletjem skorajda nepoznan, postaja najpomembnejša nova tehnologija na področju razvoja uporabniških programov oziroma razvoja informatike. To vlogo si je pridobil predvsem zato, ker omogoča veliko hitrejši razvoj uporabniških programov ob boljši kakovosti opravljenega dela. CASE tehnologija nadomešča papir in svinčnik z računalnikom in postopoma transformira razvoj računalniških programov v avtomatiziran proces. Razvoj te tehnologije temelji na ideji, da se razvije množica čim bolj integriranih orodij, katerih uporaba bistveno zmanjša obseg človekovega dela, poveča zanesljivost razvojnega procesa, kakovost izdelanih programov ... Vse to pa bistveno zmanjšuje stroške za razvoj in vzdrževanje računalniških programov. Vsako CASE orodje temelji na izbrani metodologiji.

V 15-ih letih se je razvila množica orodij, ki jih razdelimo v karakteristične skupine:
A) orodja za vzdrževanje sistemske knjižnice: orodja so nekoliko širša kot orodja, ki so namenjena podatkovnemu slovarju v sistemski knjižnici, imamo podatke v postopkih, o vseh elementih načrtovanega sistema. Sistemska knjižnica vsebuje opredelitve vseh bistvenih objektov sistema, od opredelitve postopkov v obliki diagramov pretokov podatkov, strukturnih diagramov do entitetni-relacijskih modelov, shem baze podatkov ...;
B) orodja za prenovo (reinženiring): Prenova je zahtevna naloga, da bi jo poenostavili, sistemizirali so začeli razvijati posebna orodja. Naraščajoča kompleksnost na področju informatike zahteva orodja, ki omogočajo računalniško podporo pri analizi učinkov, ki jih utegnejo imeti predvidene spremembe v organizaciji na obnašanje celotne organizacije in ki omogočajo izvedbo sprememb na ravni opredelitve potrebnih rešitev namesto na ravni programske kode. Vse to storimo s povratno preslikavo značilnosti sistema z ravni programske kode na raven logične opredelitve, kar imenujemo reinženiring. Tako lahko izvedemo spremembe na višji ravni in nato s posebnimi orodji ponovno avtomatično zgeneriramo spremenjeno programsko rešitev.;
C) orodja za podporo celotnemu razvojnemu ciklu IS: to so najcelovitejša orodja zasnovana tako, da podpirajo celoten IS. Na trgu trenutno še ni proizvodov ki bi pokrivali celotni razvojni cikel. Prevladujejo torej proizvodi, ki pokrivajo ožji ali širši odsek razvojnega procesa. Ta orodja delimo na zbirke orodij in na sisteme.
D) orodja, ki nam omogočajo nadzor nad označevanjem projekta razvoja IS (proizvodi za podporo izvedbe projekta): Projektna naloga mora biti čimbolj orientirana, določiti se mora tim, vodja tima, roki za izvedbo posameznih aktivnosti. Gre za kompleksne procese in je dobro če ga vodimo s pomočjo ustreznih računalniško podprtih orodij za vodenje inf. Projektov (pre-CASE).
E) orodja za izboljšanje kakovosti opravljenega dela: največja vrednost orodij se kaže v povečanju kakovosti opravljenega dela, v celovitosti zasnove in izgradnje celotne rešitve in v njegovi dokumentiranosti. Kakovost opravljenega dela je potrebno spremljati skozi vse razvojne faze.

Prednosti CASE ORODIJ v posameznih fazah (pogosto izpitno vprašanje!!!!):
1. FAZA NAČRTOVANJA:
   - 1.1 boljše razumevanje obravnavanega področja ter ciljev celotne organizacije in njenih sestavnih delov;
   - 1.2 boljše razumevanje dejavnikov in postopkov, ki vplivajo na doseganje ciljev;
   - 1.3 boljši pregled nad delovanjem in upravljanjem posameznih oddelkov ;
   - 1.4 boljši pregled nad informacijami, ki so pomembne za uspeh organizacije;
   - 1.5 uspešnejše načrtovanje pravočasnosti in zaporedja postopkov;
2. FAZA SNOVANJA:
   - 2.1 uspešnejše sodelovanje med snovalci oz. informatiki in uporabniki informatike;
   - 2.2 celovitejša zasnova novih rešitev;
   - 2.3 lažje prilagajanje zasnove glede na želje in potrebe uporabnikov;
   - 2.4 sprotno dokumentiranje opravljenega dela;
   - 2.5 možnost izdelave prototipov sistema v zgodnjih razvojnih fazah;
3. FAZA RAZVIJANJA IN GRADNJE:
   - 3.1 Veliko krajši razvojni časi,
   - 3.2 avtomatično dokumentiranje opravljenega dela;
   - 3.3 enostavno izvajanje sprememb

Poleg vseh teh prednosti pa je potrebno omeniti še eno, ki pa se pokaže šele kasneje v fazi delovanja, to je veliko lažje vzdrževanje rešitev, ki so bile razvite s pomočjo CASE orodij, saj so le te dobro dokumentirane skozi vse razvojne faze.
Slabosti:
1. pomanjkanje znanja razvijalcev in uporabnikov informatike,
2. izbira najustreznejšega CASE orodja zahteva uskladitev več faktorjev,
3. upoštevati je treba vrsto računalniške opreme, na kateri nameravamo uporabljati CASE orodja, specifičnosti informacijske infrastrukture...

5.4 PROBLEMATIKA VODENJA INFORMATIKE V ORGANIZACIJI
Nezadržen razvoj informacijske tehnologije močno pogojuje tudi trende vodenja informatike v organizaciji. Ti trendi so (tehnološki vidiki):
1. cenejši in zmogljivejši računalniki
2. porazdeljeni sistemi, ki povezujejo veliko število procesorjev
3. telekomunikacijska omrežja za prenos podatkov (LAN, WAN ...)
4. avtomatizacija poslovnih procesov z integracijo teksta, slike, glasu in grafike
5. računalniško povezana proizvodnja, ki povezuje CAD, CAM, robotiko, grupno tehnologijo, principe in standarde
6. profesionalno in uporabniško usmerjena informacijska orodja, ekspertni sistemi ...
7. internet/intranet

S stališča vodenja so kritični predvsem naslednji trendi:
1. spajanje različnih tehnologij, kjer gre predvsem za razkorak konceptov. Informatika tako postaja vse bolj infrastrukturna dejavnost, ki je ni več mogoče voditi na ravni rač. centra.
2. razvoj uporabniškega programiranja, ki počasi odpravlja klasično vlogo informatika kot vmesnega moža med tehnologijo in uporabniki.
3. spremenjena vloga informatike v organizaciji. Informatika se vse bolj uveljavlja kot strateška sila in orožje, nujno potrebno za enakopraven boj organizacije s konkurenco na tržišču.

Omrežno računalništvo:
1. celotno procesiranje opravi strežnik, delovna postaja samo prikazuje izhodno sliko
2. cena je manjša ker plačaš samo eno licenco
3. obremenjenost mreže (notranja vodila so bistveno hitrejša)
4. delovna postaja je lahko enostavna (80386)

5.5. KADRI
Informatikom so potrebna znanja naslednjih področij:
1. vodenje računalniških centrov
2. vodenje informacijskih centrov
3. operacijski sistemi
4. informacijska orodja

5. snovanje baz podatkov
6. skrbništvo baz podatkov
7. skrbništvo podatkov
8. varnost in zaščita podatkov in obdelav
9. računalniške komunikacije
10. revizija informacijskega sistema

Uporabniki pa morajo obvladovati:
1. osnove računalništva in informatike
2. osnove informatike za vodilne delavce
3. osnove delovanja osebnega računalnika
4. uporaba baz podatkov in informacijskih orodij
5. uporaba orodij na osebnem računalniku

5.6 Računalniški center – služba za informatiko

Z razvojem osebnih računalnikov in lokalnih mrež so se naloge iz računskega centra prenesle na osebne računalnike in lokalno mrežo. Namesto operativne vloge dobi računalniški center nove naloge:
1. koordinira razvoj informacijske strukture
2. izobraževanje uporabnikov

Služba za informatiko

Je nosilec razvoja informatike.
Naloga službe za informatiko je:
1. vodenje in koordinacija dela na projektih,
2. usklajevanje in zagotavljanje predpogojev za gradnjo posameznih podsistemov,
3. načrtovanje razvoja informacijskega sistema v skladu z razvojem in potrebami poslovnega sistema in dosežki informacijske tehnologije,
4. izobraževanje uporabnikov in nudenje pomoči uporabnikom pri razvoju svojih rešitev ter
5. zagotavljanje varnosti , zanesljivosti in kakovosti delovanja informacijskega sistema v organizaciji.

Organizacijski vidiki:
1. spremenjena vloga informatizacije poslovni funkcij
   - 1.1 od avtomatizacije k informatizaciji
   - 1.2 od AOP centra k štabni službi za informatiko
   - 1.3 strateško načrtovanje in razvoj
   - 1.4 izobraževanje
   - 1.5 podpora uporabnkov
   - 1.6 varovanje in zaščita podatkov
   - 1.7 standardi

INFORMATIKA IN STANDARDI (6. POGLAVJE)

Standardi so najprej začeli nastajati na področju strojne in komunikacijske opreme, na področju programske opreme pa le za posamezna redka področja, kot so programski jeziki, nabori znakov …

Zahteve katere je potrebno upoštevati pri informatiki (glede standardov):
1. poenotenje in prilagoditev vseh upravno-administrativnih postopkov in procedur
2. poenotenje in prilagoditev spremljanja, izkazovanja in revizije ključnih kazalcev poslovnega uspeha in bilance
3. prilagoditev programske opreme in informacijskih storitev evropskemu sistemu kvalitete

Za naše področje so najpomembnejši standardi, ki jih sprejema Mednarodna organizacija za standardizacijo (ISO) s sedežem v Ženevi. Namen te organizacije je izdajanje mednarodnih ISO standardov in ISO priporočil, pa tudi koordiniranje postopkov standardizacije na vseh področjih, razen v elektrotehniki, ki jo pokriva organizacija IEC.

STANDARD ISO 9000-3.

S pomočjo standardov se skuša doseči poenotenje tehničnih rešitev in proizvodov ter njihovo primerljivost tudi v ogledu kvalitete. Kvaliteta proizvoda je glavni atribut večine industrijskih izdelkov, ki so uspešni na trgu, zagotavljati in ugotavljati pa jo je mogoče le ob razdelanem sistemu standardov, ki vpeljujejo red in metriko za merjenje in ugotavljanje kvalitete. Standardi vnašajo ustrezno pravno varnost in zaščito partnerjem, ki so v poslovnem odnosu.

Kot rezultat splošnih potreb po vpeljavi univerzalnega sistema kakovosti industrijskih proizvodov je nastala serija mednarodnih standardov ISO 9001-2,3,4. Ti standardi so nastali sredi 80-ih let. Kmalu pa je postalo očitno, da področje programskih proizvodov, njihovega načrtovanja, razvoja in vzdrževanja, potrebuje svoj standard za vzpostavitev sistema vzdrževanja in ugotavljanja kvalitete. Tako je nastal standard ISO 9000-3.

Ta standard se ne ukvarja samo s kvaliteto končnega proizvoda, temveč opredeljuje vse karakteristične razvojne faze, skozi katere gre poljuben programski proizvod. Opredeljuje pa tudi pogodbeno razmerje med kupcem in proizvajalcem programske opreme.

ISO 9000-3 torej skuša uveljaviti sistem kvalitete skozi celoten razvojni cikel nekega programskega proizvoda. Ugotavljanje kvalitete končnega proizvoda je težavno, zato gre v primeru standarda ISO 9000-3 za to, da se zagotavlja kvaliteta v vseh razvojnih fazah. Če je celoten razvoj kvaliteten, je kvaliteten tudi končni proizvod – informacijska rešitev.

Ključni elementi sistema zagotavljanja kvalitete programskih proizvodov

Sistem kvalitete programskih proizvodov standard ISO 9000-3 gradi preko naslednjih elementov:

1. **Normativni okviri**: standard ISO 9000-3 ne moremo obravnavati samostojno, temveč le v povezavi z drugimi standardi, s katerimi tvori koherenten sistem. To so naslednji standardi: ISO 2382-1:1984 (Data processing – Vocabulary), ISO 8402:1986 (Quality – Vocabulary), ISO9001:1987 in ISO 10011-1:1990;
2. **Osnovne definicije**: Z njimi se skuša podrobneje opredeliti elemente programskega produkta, kakor tudi elemente razvojnega procesa;
3. **Opredelitev sistema kvalitete**: Sistem, ki naj bi zagotavljal kvaliteto programskega proizvoda, se začne graditi pri vodstvu proizvajalca ali dobavitelja programskega proizvoda in se sestoji iz naslednjih pomembnejših točk:
   - 3.1. vodstvo dobavitelja ali proizvajalca programskega proizvoda mora jasno definirati in dokumentirati svojo politiko, cilje in privrženost k zagotavljanju kvalitete; odgovorno je za izvajanje te politike na vseh ravneh organizacije ali podjetja
   - 3.2. na ravni organizacije morajo biti definirane zadolžitve in odgovornost vseh uslužbencev, ki vodijo, izvajajo ali kontrolirajo aktivnosti na področju zagotavljanja kvalitete
   - 3.3. proizvajalec programske opreme je dolžan identificirati elemente in aktivnosti interne kontrole kvalitete
   - 3.4. verifikacijske aktivnosti morajo vključevati inšpekcijo, testiranje in nadzor vseh faz razvojnega procesa in proizvodnega procesa
   - 3.5. vodstvo podjetja proizvajalca programskega produkta je dolžno v ustreznih intervalih opravljati revizijo izvajanja sistema kvalitete na vseh ravneh

   V tem sistemu ima tudi kupec oz. naročnik programskega proizvoda svoje obveznosti, ki se nanašajo predvsem na ustrezno kontaktiranje s proizvajalcem

in posredovanje potrebnih informacij glede:

3.6. podrobne opredelitve naročnikovih zahtev in specifikacije programskega proizvoda
3.7. sprotno odgovarjanje na vsebinska vprašanja, na katera naleti proizvajalec v času razvoja produkta
3.8. potrditev proizvajalčevih predlogov in rešitev

Sistem kvalitete se po ISO 9000-3 gradi na naslednjih elementih:

a. proizvajalec/dobavitelj programske opreme mora vzpostaviti in vzdrževati dokumentiran sistem kvalitete, ki je integriran proces skozi celotni življenjski cikel, kar zagotavlja da so ukrepi za zagotavljanje kvalitete vgrajeni v vse razvojne faze.
b. vse aktivnosti zagotavljanja kvalitete morajo biti vnaprej planirane za vsak razvojni projekt posebej;
c. zagotovljen mora biti nadzor nad izvajanjem teh aktivnosti in analiza vzrokov odstopanja ;

4. **Sistem kvalitete skozi življenjski cikel proizvoda**: Bistven pogoj za doseganje kvalitete programskega proizvoda je, da se le-ta razvija v skladu z izbranim modelom razvojnega cikla in da so ukrepi za zagotavljanje kvalitete vgrajeni v vse razvojne faze. Ker se razvoj programskega produkta največkrat začne s sklenitvijo pogodbenega razmerja, je v standardu posvečena posebna pozornost ustreznemu formuliranju pogodbenega odnosa, ki zagotavlja zadovoljivo pravno jamstvo obema pogodbenima strankama. Standard predvideva oz. zahteva izdelavo razvojnega plana projekta, vključno z opredelitvijo virov, projektnega tima, odgovornosti, subpogodbenikov … Razvojni plan mora opredeljevati vse razvojne faze, vključno z elementi njihove realizacije in kontrole.;
5. **Pomožne aktivnosti**: izdelava plana testiranja programskega proizvoda ter sami izvedbi testiranja in verifikacije. V tem standardu so podrobno opredeljeni vzdrževanje programskega proizvoda, če je le-to predmet pogodbe, plani vzdrževanja …

POSEBNOSTI STANDARDA DIN 66285 V PRIMERJAVI S STANDARDOM ISO 9000-3:

DIN 66285 - nemški standard, ki ureja pogoje določanja in testiranja kvalitete programske opreme Ta standard je bil razvit pred standardom ISO 9000-3. Oba standarda se medsebojno kombinirata.

Razlika med ISO 9000-3 in DIN 66285

ISO 9000-3 ugotavlja razvoj kvalitete skozi celoten razvojni cikel, DIN 66285 pa ugotavlja kakšna je kvaliteta končnega programskega proizvoda. Nemški standard izhaja iz končnega proizvoda, za katerega opredeljuje elemente kvalitete in načine njihovega ugotavljanja. Standard ISO 9000-3 je za uvedbo zelo zahteven, medtem ko je standard DIN 66285 lažje uvesti.

Standard DIN 66285 navaja naslednje elemente kvalitete programskega proizvoda:

1. k vsakemu programskemu proizvodu spada še njegov funkcionalni opis in dokumentacija;
2. funkcionalni opis proizvoda mora vsebovati vse njegove funkcije, in te mora proizvod tudi izpolnjevati,
3. proizvod mora biti opremljen z dokumentacijo, ki mora vsebovati vse o uporabi, inštalacij in vzdrževanju proizvoda;
4. podana mora biti specifikacija vseh elementov, to je programski proizvod sestavljajo.

Vse funkcionalne zahteve, ki se nanašajo na programski proizvod, morajo biti deljene na želene in obvezne. Programski proizvod je v skladu s standardom, če proizvod izpolnjuje vse obvezne funkcionalne zahteve. Poleg tega pa standard DIN 66285 opredeljuje tudi načine in pogoje ugotavljanja kvalitete programskega proizvoda.

INFORMATIZACIJA UE

GLOBALNA IZHODIŠČA:

1. spremenjena vloga države, usmeritev v servisno in perspektivno funkcijo,
2. proces decentralizacije in delovanja upravnih organov;
3. profesionalizacija drž. organov in dvig njihove strokovnosti v odvisnosti od spremenjenih funkcij države;
4. sprememba v notranji organizacijski strukturi države;
5. prehod težišča kontrolne funkcije v upravi na področju ocenjevanja rezultatov upravnega dela;
6. prehajanje servisne funkcije uprave iz državne domene v izvajanje usposobljenim subjektom izven državne uprave.

GLOBALNI CILJI:

1. racionalizacija poslovanja drž. uprave in večja učinkovitost delovanja;
2. zaščita obstoječih investicij v informacijsko tehnologijo z zagotavljanjem čim popolnejšega sistema integracije z uvajanjem in vzdrževanjem enotnih metodologij in standardov, upoštevajoč trende evropskega in mednarodnega koncepta odprtih sistemov;
3. zagotavljanje varovanja in zaščite podatkov, snovanje in oblikovanje okolja informacijske izmenjave med državo in lokalno upravo;
4. zagotavljanje skupne komunikacijske in skupne računalniške zmogljivosti skozi instalacijo, delovanje, upravljanje in vzdrževanje državne inf. in tehn. infrastrukture;
5. integracija telekomunikacijskih storitev a ravni UE.

KLJUČNI PROBLEMI DELOVANJA UE:

1. predolgi časi za reševanje zadev;
2. Nepovezane rešitve;
3. Necelovite evidence (ne evidentira se dokumentov, ki pridejo po FAXU, EP);
4. Šibka podpora strokovnemu delu (rač. podprte so timske aktivnosti, strokovni podatki, ki so potrebni pri reševanju, pa niso na voljo tej podpori);
5. Slab pregled nad delom strokovnih delavcev;
6. nerešene horizontalne/vertikalne povezave.

IZHODIŠČA IN CILJI (PRI NOVI REŠITVI):

1. prenova poslovanja UE: Cilj: preoblikovati, optimizirati, racionalizirati postopke tako, da bo mogoče optimalno izkoristiti možnosti, ki nam jih daje inf. teh.;
2. izboljšanje kakovosti storitev;
3. rešitev za naslednje tisočletje: rešitev, ki bo zagotavljala kakovost državne uprave čim dlje časa;
4. upravljanje postopkov: nov koncept, ki omogoča razvoj rač. rešitev s katerimi lahko vodimo, krmilimo in pripravljamo celotne postopke in nadzorujemo vse aktivnosti znotraj takega postopka;
5. Uvajanje elektronskih akrov: vsebuje vse dokumente v elektronski obliki- skrajševanje časa, problemi - zloraba podatkov, verodostojnost dokumentov;
6. Integracija rešitev: sedanje rešitve so nepovezane, povzročajo veliko stroškov, dodatnega dela;
7. Integracija telekomunikacij: telekomunikacije je treba integrirat v celoten sistema;
8. Kontinuiteta v poslovanju: uprava je sistem, ki mora delovat na daljši rok.

PRIKAZ STRUKTURE CELOTNEGA PROJEKTA: (PROJEKT INFORMATIZACIJE UE)

1. FAZA:
1.1 PRENOVA POSLOVANJA: cilj prenove je temeljito analizirat obstoječe poslovanje, ugotoviti neracionalnosti, kjer je možno to poslovanje spremeniti, posodobiti ob uporabi inf. teh. in postaviti nov model poslovanja;

1.2 podprojekt - RAZVOJ IN PRILAGODITEV PROGRAMSKO TEHNIČNIH REŠITEV: ugotoviti moramo kaj je pri obstoječih rešitvah dobrega, kar pa ni dobro pa je treba razviti;
1.3 PREHOD IN RAZVOJ PODATKOVNIH POVEZAV OBSTOJEČIH NA NOVE REŠITVE;
1.4 UVEDBA NOVEGA NAČINA POSLOVANJA: tu ni mišljena konkretna uvedba, uvajanje bo trajalo 3-4 leta , da bi ta čas skrajšali je treba k temu pristopit čimbolj sistematično.

2. FAZA: PRENOVA POSLOVANJA: UVAJANJE V POSAMEZNE UE (izvedbeni del)

PROBLEMI PRENOVE POSLOVANJA SLOVENSKE DRŽAVNE UPRAVE:

1. nihče v SLO nima podrobnega pregleda nad tem, kateri in koliko postopkov se izvaja v UE;
2. evidenca in register postopkov, ki so v pristojnosti UE ne obstaja;
3. podrobnejših navodil o izvajanju posameznih postopkov ni;
4. kljub formalno isti pravni podlagi se v različnih UE isti postopki izvajajo različno;
5. tudi do 90% pri izvajanju postopkov gre na račun transportnih poti ter različnih čakalnih časov in rokov;
6. ogromno časa se porabi za pridobivanje različnih soglasij in vsebinskih informacij, pomembnih za reševanje konkretnega primera, zaradi nepovezanih upravnih organizacij in organov na lokalni ravni ter nepovezanosti baz podatkov.

PRENOVA POSLOVANJA:

1. **faza** (na ravni postopkov):

- analiza postopkov,
- analiza inf. tokov,
- informacijska integracija,
- razvoj organizacijskega okolja,
- optimizacija, tipizacija, standardizacija.

Cilj podprojekta: poenostavit, optimizirat, standardizirat postopke v upravi. Drugi del prenove poslovanja zahteva spremembo organiziranosti UE in spremembo zakonodaje.Vse postopke je treba podrobno analizirat. Nato analizirat inf. tokove. Na osnovi tega nastane podatkovni model. Analizirati moramo še organizacijske akte.

2. faza (na ravni procesov): - analiza procesov na ravni UE,

- analiza inf. tokov med upravnimi organi in drugimi institucijami,
- prenova normativnih aktov,
- razvoj organizacijskega okolja,
- organizacijska prenova.