Dr. Josip Brumec
Fakultet organizacije i informatike
Varaždin

UDK: 007.5
Izvorni znanstveni rad

# Optimizacija strukture složenih informacijskih sustava

*Greške u početnoj fazi projektiranja bitno poskupljuju razvoj te umanjuju učinkovitost primjene novog informacijskog sustava. Zbog toga se u ovom radu analiziraju problemi i mogućnosti optimiranja osnovne strukture IS-a, predlaže drugi pristup i izlaže jedan novi opći algoritam za praktičnu provedbu te optimizacije.*

**Ključne riječi:** optimizacija, informacijski sustavi, analiza afiniteta među procesima.

## 1 Uvodna razmatranja

Svaki realni sustav (proizvodni, poslovni, društveni itd.) ostvaruje svoje zadaće, skladno postavljenim ciljevima, izvođenjem velikog broja međusobno povezanih i usklađenih procesa. Oblik povezivanja procesa, način, redoslijed i uvjeti njihovog izvođenja obično se nazivaju **poslovnom tehnologijom** . Djelotvornost realnog sustava (RS) bitno zavisi, osim o ostalim čimbenicima, o kvaliteti poslovne tehnologije te o brzini i učinkovitosti izvođenja svakog pojedinačnog procesa.

Informacijski sustav (IS) nekog realnog sustava možemo smatrati informatičkim modelom poslovne tehnologije RS-a. Kao što procesi čine poslovnu tehnologiju realnog sustava, tako i pojedinačni programi za računalo, koji podržavaju pripadajuće procese, čine programsku podršku informacijskog sustava[1] Definiranje poslovne tehnologije RS-a i projektiranje informacijskog sustava za njega su dva pristupa rješavanju istog problema: povećanja djelotvornosti realnog sustava. Iz toga slijedi da se IS može projektirati samo na osnovi dobrog poznavanja svih procesa realnog sustava, bez obzira

[1] Informacijski sustav ne čini samo programska podrška, premda je ona njegov najsloženiji tehnološki dio, već i niz pisanih ili nepisanih pravila o radu i komuniciranju, uputa, izvještaja itd. Razvojem i uvođenjem programske podrške još nije izgrađen IS. To se prečesto zaboravlja, pa programska podrška i sklopovska oprema, nabavljena za "novi IS" ostaje strano tijelo u realnom sustavu, a njegov pravi IS djeluje onako kako je djelovao i prije nemale investicije.

bili oni na razini izvođenja, upravljanja ili odlučivanja. Suvremene metodike projektiranja IS-a imaju stoga svoje ishodište u analizi procesa i njihovih međusobnih veza.

Procese i stanja realnog sustava, ili zbirno rečeno **objekte**, povezuju različiti tokovi: materijala, resursa, ljudi, energije, naloga, informacija itd. Ti tokovi mijenjaju stanja u realnom sustavu (zaliha, ugovora, sredstava, kapaciteta itd.). Ekvivalent tih tokova u IS-u su **klase podataka**, koje možemo definirati kao *skup različitih podataka o istom objektu* . Procesi realnog sustava različiti su po svojoj naravi i sadržaju, ali ih možemo na općoj, semantičkoj razini predstaviti kao objekte s više ulaznih i više izlaznih klasa podataka *(slika 1)* . Ulaznim klasama podataka k smatrat ćemo one koje promatrani proces koristi, a nastale su (ili su generirane) u nekom drugom procesu. Izlazne klase podataka g neka su one, koje se stvaraju (ili generiraju) u promatranom procesu, a koriste se u nekim drugim procesima.


Slika 1: Model procesa

Čimbenike iz realnog sustava, relevantne za strateško planiranje njegovog informacijskog sustava, moguće je prikazati u matrici međuzavisnosti procesa i klasa podataka ili kraće (**P/K** -matrici *(slika 2)* .

Ova tehnika, poznata iz BSP-analize [7], pogodno je ishodište za optimizaciju os-

novne strukture IS-a, pa će biti korištena za daljnja razmatranja.

| KLASE PO-DATAKA / PROCESI | $k_1$ | $k_2$ | $k_j$ | $k_n$ |
|---|---|---|---|---|
| $P_1$ | G | - | K | - |
| $P_2$ | - | G | - | K |
| | | | | |
| $P_i$ | K | K | G | - |
| | | | | |
| $P_m$ | - | K | K | G |

m - broj procesa u sustavu

n - broj klasa podataka u sustavu

$e_{ij}$ { S - $i$ - ti proces stvara $j$ - tu klasu podataka
K - $i$ - ti proces koristi $j$ - tu klasu podataka
- - $i$ - ti proces i $j$ - ta klasa podataka nisu u uzajamnoj vezi

Slika 2: Matrica odnosa **procesi/klase podataka**

Da bi gornja matrica predstavljala formalno ispravan model IS-a, ona mora imati slijedeća svojstva:

1. Matrica mora biti **m x n** reda, što znači da predstavlja sustav sa **m** procesa i **n** klasa podataka.

2. U svakom retku matrice sadržan je potpun skup klasa podataka koje pripadaju jednom procesu. U i-tom retku a j-tom stupcu upisuje se oznaka $\mathcal{K}$ ako i-ti proces koristi j-tu klasu podataka, a oznaka $\mathcal{G}$ ako i-ti proces generira (ili stvara) j-tu klasu podataka.

3. U svakom stupcu sadržan je potpun skup procesa kojima pripada jedna klasa podataka, bilo da je korištena $\mathcal{K}$ ili generirana $\mathcal{G}$ u nekom procesu.

4. U jednom retku smije se nalaziti više elemenata s vrijednošću $\mathcal{K}$ i više elemenata s vrijednošću $\mathcal{G}$, što znači da jedan proces ima više ulaznih klasa podataka te da svojim djelovanjem može stvoriti više novih klasa podataka. Redak ne smije biti prazan jer ne postoji proces koji niti treba niti stvara barem jedinu jednu klasu podataka.

5. U jednom stupcu smije se nalaziti samo jedan element s vrijednošću $\mathcal{G}$ jer se klasa podataka može stvoriti samo u jednom procesu. Odstupanje od tog pravila

ukazivalo bi na organizacijski kaos u poslovnoj tehnologiji realnog sustava. Nasuprot tome, u istom stupcu smije biti više elemenata s vrijednošću $\mathcal{K}$. Ako u j-tom stupcu nema oznake $\mathcal{G}$, što je dozvoljeno, onda to znači da j-ta klasa ulazi u sustav iz njegove okoline.

Matricu s gornjim svojstvima valja smatrati formalno ispravnim modelom IS-a. Kod izrade matrice pažnju treba posvetiti nekim slučajevima, koji ukazuju na greške kod izrade ili pak na možebitne anomalije u poslovnoj tehnologiji realnog sustava:

a) Redak koji ima samo elemente s vrijednošću $\mathcal{G}$ ukazuje na pro- proces koji djeluje neovisno od drugih zbivanja u sustavu.

b) Stupac koji sadrži samo $\mathcal{G}$ -vrijednosti ukazuje da je neka klasa podataka suvišna.

Na osnovi ovakve matrice može se izvesti osnovna struktura novog informacijskog sustava. Heuristički algoritam za taj postupak, koji se često navodi u literaturi (npr. [1]), predlaže slijedeće korake:

1. Valja odrediti granice **RS-a** i izraditi korektnu **P/K** -matricu s potpunim skupom procesa i klasa podataka.[2] Početna matrica obično ima jednoliko raspoređene oznake $\mathcal{G}$ i $\mathcal{K}$ po cijeloj površini.

2. Treba poredati procese po redoslijedu, koji proizlazi iz životnog ciklusa osnovnih resursa **RS-a** , a zatim rasporediti stupce s klasama podataka tako da oznake $\mathcal{G}$ budu raspoređene po glavnoj dijagonali matrice. Novi raspored oznaka $\mathcal{K}$ mora odražavati isti smisao veze procesa i klase podataka.

3. Treba formirati $s^2$ submatrica (gdje je s zeljeni broj podsustava) vodoravnim i okomitim dijeljenjem **P/K** -matrice. Oznake $\mathcal{G}$ moraju ostati unutar submatrica koje se nalaze na glavnoj dijagonali.

4. Podskupovi procesa, koji pripadaju submatricama na glavnoj dijagonali, predstavljaju *podsustave* informacijskog sustava. Nakon toga treba nacrtati veze između podsustava na taj način da se klase podataka generirane u bilo kojem procesu jednog podsustava smatraju *izlaznim* veličinama iz tog podsustava. I obrnuto, klase podataka koje se koriste u bilo kojem procesu jednog podsustava treba smatrati njegovim *ulaznim* veličinama.

Primjena ovog algoritma je prikazana na *slikama 3a) b) i c)* za jednostavnu matricu, koja će se kao primjer koristiti i kod kasnijih razmatranja o optimizaciji osnovne strukture informacijskog sustava. Primjer predstavlja model sasvim jednostavnog realnog sustava sa m=9, procesa i n=8 klasa podataka. Sustav iz ovog primjera možemo smatrati uvjetno zatvorenim, što se zaključuje iz dvije činjenice:

[2]Prepoznavanje potpunog skupa procesa u složenom sustavu nije lagan zadatak, pa za to postoje brojne metode. Najčešće se koristi postupak dekompozicije poslovnih funkcija i metoda životnog ciklusa.

a) Sve se klase podataka generiraju unutar sustava. To znači da ne postoji, ili nije prikazana, klasa podataka koju bi razmatrani sustav dobivao iz okoline.

b) Postoje dva naizgled nesvrsishodna procesa ($P_3$ i $P_6$), koji ne generiraju nijednu klasu podataka. Za njih se pretpostavlja da izvode neke aktivnosti za potrebe okoline, pa klase podataka, koje oni vjerojatno generiraju, nisu prikazane u matrici.

Na *slici 3a)* data je početna matrica, a na *slici 3b)* tzv. dijagonalizirana matrica s naznačenom podjelom u tri podsustava. *Slika 3c)* pokazuje moguću varijantu **osnovne arhitekture** informacijskog sustava, a *slika 3d)* **organizacijsku strukturu** razmatranog realnog sustava.

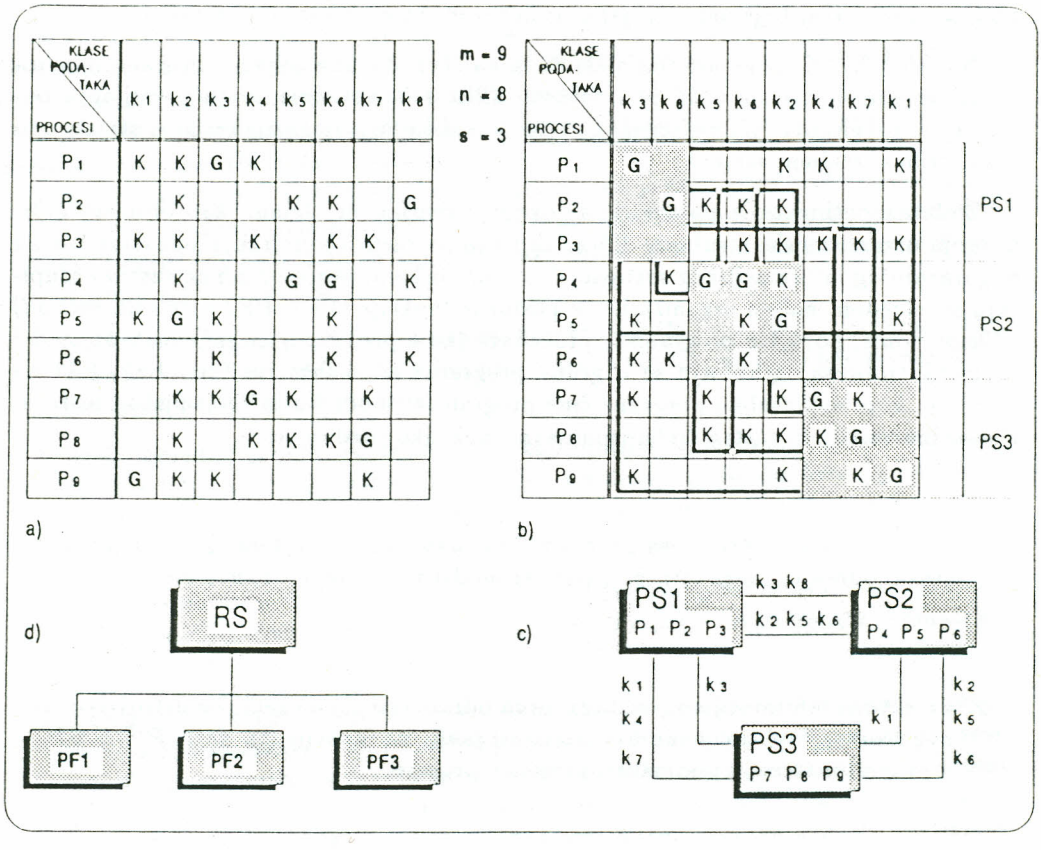
Zadatak optimizacije pojavljuje se ovdje u dvojakom obliku: kao problem optimiranja organizacijske strukture RS-a, ali i kao problem optimiranja strukture njemu odgovarajućeg informacijskog sustava. Pojedini poslovni procesi realnog sustava grupiraju se, slijedom načela organizacije, u poslovne funkcije (PF1, PF2 i PF3 na *slici 3d)* . Slično tome, pojedine programske procedure (koje predstavljaju informatički model poslovnih procesa) grupiraju se u grupe programa (ili aplikacije PS1, PS2, PS3 na *slici 3c)* , koje međusobno povezane čine programsku podršku informacijskog sustava. Dobar informacijski sustav treba ispunjavati nekoliko preduvjeta:

1. Informacijski sustav mora potpuno podržavati poslovnu tehnologiju realnog sustava, odnosno mora biti informatički model te poslovne tehnologije.[3]

2. Struktura informacijskog sustava mora odražavati organizacijsko ustrojstvo realnog sustava: aplikacije moraju pokrivati poslovne funkcije (tj. važi: $\mathcal{PF}_i \Longleftrightarrow \mathcal{PS}_i$, a programi trebaju podržavati poslovne procese.

3. Organizacijske veze u realnom sustavu, odnosno podatkovne veze u informacijskom sustavu, moraju biti definirane i jednoznačne.

[3] Nepoštivanje ovog načela vjerojatno je glavni uzrok relativno slabe učinkovitosti računalom podržanih informacijskih sustava. Ako se nabavlja gotova, tzv. standardna programska podrška, onda i cjelokupnu poslovnu tehnologiju treba podvesti pod te iste standarde.




Slika 3. Razvoj osnovne arhitekture informacijskog sustava

## 2 Formalizacija problema optimizacije

Matrica prikazana na *slici 3a)* prikladan je ishodišni model za provedbu optimizacije ustrojstva informacijskog sustava. U literaturi (npr. [3]) se navodi da je informacijski sustav dobar ako su zadovoljeni slijedeći specifični kriteriji:

- maksimalna unutrašnja kohezija unutar podsustava,
- minimalna vanjska povezanost između podsustava i
- visok stupanj sličnosti njihove unutarnje strukture.

Međutim, atributi "maksimalni", "minimalni" i "visok", u smislu prethodne rečenice, opće su i teško mjerljive odrednice, a njihovo vrednovanje podliježe subjektivnoj procjeni ocjenjivača. Analizom svojstava **P/K** -matrice i njezinih submatrica može se zaključiti da je:

- konzistentniji onaj sustav, koji unutar podsustava ima više oznaka $\mathcal{K}$, odnosno veću gustoću pokrivanja dijagonalnih submatrica.
- slabije povezan onaj sustav, koji ima manje oznaka $\mathcal{K}$ izvan dijagonalnih submatrica i
- veći stupanj sličnosti postoji između onih podsustava, čiji je broj procesa bliži nekoj prosječnoj vrijednosti (m:k)

Problem optimizacije složenog informacijskog sustava, sastavljenog od više podsustava, može se formulirati na slijedeći način:
**valja pronaći takav poredak procesa i način njihovog grupiranja u podsustave da oni imaju što veću unutrašnju konzistentnost i što manju međusobnu povezanost**, a da broj procesa u podsustavima odstupa od prosječne vrijednosti za neki dozvoljeni iznos. Za rješavanje toga treba:

- definirati ciljnu funkciju,
- postaviti teoretske ili praktične granice, unutar kojih se može tražiti rješenje i
- pronaći heuristički ili egzaktni algoritam kojim se, uz prihvatljiv opseg računanja, može doći do rješenja.

Osnovna ideja optimizacije, izložena u daljnjem tekstu ovog članka, sastoji se u slijedećem: utvrditi sve moguće načine sastavljanja sustava od prepoznatih procesa i uvesti određenu metriku pomoću koje se može vrednovati svaki od dozvoljenih načina strukturiranja.

## 2.1 Formalizacija parametara optimizacije

Neka je oznakom m određen red sustava, tj. ukupan broj procesa u **P/K** -matrici, a oznakom s stupanj kompozicije sustava, tj. broj podsustava (ili submatrica) od kojih je sastavljen sustav. Na osnovi toga uvodi se pojam *plan kompozicije s-tog stupnja* $\mathcal{PL}_f$. To je s-komponentni vektor, čije vrijednosti komponenata sadrže broj procesa grupiranih u pojedinačnim podsustavima $\mathcal{PS}_j$. Broj komponenata vektora plana jednak je broju podsustava, a vrijednost i-te komponente određen je brojem procesa grupiranih u i-tom podsustavu. Vrijedi dakle:

$$\mathcal{PL}_s = \{p^{(1)}, p^{(2)}, ..., p^{(i)}, ..., p^{(s)}\} \tag{1}$$

$$\text{tako da je: } m = \sum_{i=1}^{s} p^{(i)} \tag{2}$$

gdje su: $p^{(i)}$ broj procesa u i-tom podsustavu,
s - stupanj kompozicije i
m - red sustava.

Broj stupnjeva kompozicije kreće se teoretski u rasponu $1 \leq s \leq m$. Međutim, vrijednosti s=1 i s=2 nemaju praktičnog smisla (jer bi to značilo da se sustav sastoji samo od jednog ili dva podsustava). S druge pak strane, iz teorije o organizaciji poznato je pravilo " $7 \pm 2$ " (za broj podsustava u nadređenom sustavu, ili opseg rukovođenja). Iako se to pravilo ne odnosi sasvim na problem o kojem se ovdje raspravlja, smatrat ćemo prihvatljivim uzeti kao gornju granicu vrijednosti s=11. Na takav se način može definirati domena stupnja kompozicije:

$$3 \leq s \leq 11 \tag{3}$$

Element plana kompozicije je cjelobrojna konstanta iz intervala:

$$p_{min} \leq p^{(i)} \leq p_{max} \tag{4}$$

Pojedinačne vrijednosti $p_i$ nisu sasvim nezavisne s obzirom na izraz (2), a granične vrijednosti elemenata vektora plana kompozicije $\mathcal{PL}_s$ određene su odnosom reda sustava m i stupnja kompozicije s:

$$p_{min} \leq \frac{m}{s}; \quad p_{max} = m - (s-1) \cdot p_{min} \tag{5}$$

Navedimo kao primjer da je na *slici 3b)* provedena kompozicija trećeg stupnja te vektor plana kompozicije glasi: $\mathcal{PL}_s = 3, 3, 3$.

Broj mogućih planova kompozicije s-tog stupnja, za sustav m-tog reda, jednak je matematskom broju kombinacija s ponavljanjem s-tog razreda, reda, koje se mogu sastaviti od onoliko različitih elemenata, koliko cijelih brojeva ima u intervalu $[p_{min}, p_{max}]$, ali uz uvjet da se broje samo one kombinacije za koje zbroj vrijednosti elemenata vektora plana na kompozicije zadovoljava izraz (2). Možemo dakle zapisati:

$$n_s = C'\binom{s}{b} = \frac{(b+s-1)!}{(b-1)!\cdot s!} \quad \text{uz uvjet: } \sum_{i=1}^{s} p^{(i)} \tag{6}$$

gdje je b pomoćna varijabla za koju vrijedi: $b = (p_{max} - p_{min} + 1)$. S obzirom na izraz [5] za $b_{max}$, konačno vrijedi:

$$b = m + 1 - s \cdot p_{min} \tag{7}$$

Broj planova kompozicije zavisi dakle samo o stupnju kompozicije i najmanjem dozvoljenom broju procesa u podsustavu. Ako je npr.: m=19; s=3 i $p_{min} = 5$, onda je $p_{max} = 19 - 2 \cdot 5 = 9$ i $b = 5$. U tom slučaju kombinira se 5 elemenata iz skupa { 5,6,7,8,9 } u razrede od po tri elementa. Takvih kombinacija ima prema izrazu (6) ukupno 35, ali samo 4 od njih zadovoljavaju uvjet (2). Postoje dakle 4 vektora plana kompozicije:

$$\mathcal{PL}_3^{(1)} = \{5,5,9\} \qquad \mathcal{PL}_3^{2} = \{5,6,8\}$$
$$\mathcal{PL}_3^{(3)} = \{5,7,7\} \qquad \mathcal{PL}_3^{4} = \{6,6,7\}$$

Na osnovi ovih razmatranja relativno se jednostavno može izraditi algoritam i napraviti program za računalo pomoću kojeg će se generirati svi dozvoljeni planovi kompozicije, uz zadane ulazne vrijednosti parametara za **m, s i $p_{min}$**.[4]

Ukupni broj svih dozvoljenih planova kompozicije za sustav **m**-tog reda $N_m$ dobiva se tako da se zbroje planovi za svaki stupanj kompozicije. Vrijedi dakle:

$$N_m = \sum_{s=3}^{s=11} n_s \tag{8}$$

Na tablici I je prikazan broj planova kompozicije, zavisno od stupnja kompozicije s i najmanjeg dozvoljenog broja procesa u podsustavu, za red sustava m=20. Rezultati su dobiveni generiranjem planova pomoću računala, ali je red sustava takav da se neke izračunate vrijednosti (npr. za $p_{min} = 4$ i $s = 3$) još mogu provjeriti običnim računanjem. Očito je i iz ove tablice da broj planova kompozicije za sustav reda m ovisi samo od parametara s i $p_{min}$, kao što je to bilo pokazano izrazom (7).

[4] Kod izrade programskog algoritma za generiranje planova kompozicije jednostavnije je krenuti od varijacija s ponavljanjem s-tog reda nad b elemenata, uz selekciju onih koje sadrže različite elemente i zadovoljavaju uvjet (2).

Tablica I

| $P_{min}$ | Stupanj kompozicije (s) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 1 | 33 | 64 | 84 | 90 | 82 | 70 | 54 | 42 | 30 |
| 2 | 24 | 34 | 30 | 20 | 11 | 5 | 2 | 1 | |
| 3 | 16 | 15 | 7 | 2 | | | | | |
| 4 | 10 | 5 | 1 | | | | | | |
| 5 | 5 | 1 | | | | | | | |
| 6 | 2 | | | | | | | | |

Na tablici II dat je pregled broja mogućih planova kompozicije u zavisnosti od stupnja kompozicije s i reda sustava m , za slučaj da je dozvoljen podsustav sa samo jednim procesom (tj. $p_{min} = 1$ ).

Tablica II

| $m$ | Stupanj kompozicije (s) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 20 | 33 | 64 | 84 | 90 | 82 | 70 | 54 | 42 | 30 |
| 30 | 75 | 206 | 377 | 532 | 618 | 638 | 598 | 530 | 445 |
| 40 | 133 | 478 | 1115 | 1945 | 2738 | 3319 | 3589 | 3590 | 3370 |
| 50 | 208 | 920 | 2611 | 5427 | 8946 | 12450 | 15244 | 16928 | 17475 |
| 60 | 300 | 1575 | 5260 | 12692 | 23961 | 37638 | 51294 | 62740 | 70515 |

Broj mogućih planova kompozicije raste brzo i nelinearno s rastom reda sustava i stupnja kompozicije. Ako treba npr. sustav s 50 procesa (m=50) sastaviti tako da ima 10 podsustava (s=10), dozvoljavajući pri tome da postoje i podsustavi sa samo jednim procesom ($p_{min} = 1$), onda se to može učiniti na 16.928 različitih načina.

Izrazom (8) utvrđen je broj dozvoljenih vektora planova kompozicije, odnosno mogućih načina formiranja s -podsustava od m -procesa. S time, međutim, još nije određen sadržaj svakog podsustava, tj. nisu poznati procesi koji ga čine. Svaki pojedinačni plan kompozicije predstavlja samo predložak za objedinjavanje procesa u podsustave. Tako je npr. sustav 9-og reda sa *slike 3a)* moguće oblikovati u 3 podsustava, dozvoljavajući pri tome da minimalni broj procesa u jednom podsustavu bude $p_{min} = 2$. Vektori planova kompozicije za taj slučaj su {2,2,5}, {2,3,4} i {3,3,3}. Svaki takav vektor plana kompozicije može se popuniti s jednom permutacijom od m raspoloživih procesa. Da saznamo koliko ima ukupno tih permutacija, valja razmišljati na slijedeći način: za prvi podsustav $\mathcal{PS}_i$ moguće je odabrati prvih $p^{(i)}$ procesa na

$$\binom{m}{p^{(1)}} = \frac{m!}{p^{(1)}! \cdot (m - p^{(1)})!}$$

načina (kombinacije bez ponavljanja od $p^{(1)}$ elemenata po slogu, od ukupno m elemenata). Za idući podsustav treba odabrati $p^{(2)}$ elemenata od preostalih $m - p^{(1)}$, što se može učiniti na

$$\binom{m-p^{(1)}}{p^{(2)}} = \frac{(m-p^{(1)})!}{p^{(2)}!\cdot(m-p^{(1)}-p^{(2)})!}$$

načina. Slijedeći dalje ovu logiku razmišljanja, dolazi do izraza *(9)* :

$$\mathcal{D}_s = \binom{m}{p^{(1)}} \cdot \binom{m-p^{(1)}}{p^{(2)}} \cdot \binom{m-p^{(1)}-p^{(2)}}{p^{(2)}} \ldots \binom{m-p^{(1)}-p^{(2)}-\ldots p^{(s-1)}}{p^{(s)}} \tag{9}$$

Sređivanjem gornjeg izraza dobiva se:

$$\mathcal{D}_s = \frac{m!}{p^{(1)}!\cdot p^{(2)}!\ldots p^{(s)}!} \tag{10}$$

Iz dosad razmatranog slijedi da ukupan broj sadržaja, kojima se može popuniti sustav m -tog reda, sastavljen od s -podsustava, iznosi:

$$\mathcal{V} = \sum N_s \cdot \mathcal{D}_s \tag{11}$$

Jedno stanje $\mathcal{S}$ strukture sustava može se dakle općenito opisati skupom od s disjunktnih podskupova, čiji su elementi izabrani iz m -procesa:

$$\mathcal{S} = \mathcal{PS}_1 \cup \mathcal{PS}_2 \cup \ldots \cup \mathcal{PS}_s = \{\sum_{i=1}^{m}\} \tag{12}$$

Matrica na *slici 3b)* je popunjena sadržajem koji se može opisati kao:

$$\mathcal{S} = (\mathcal{PS}_1\{p_1, p_2, p_3\}, \mathcal{PS}_2\{p_4, p_5, p_6\}, \mathcal{PS}_3\{p_7, p_8, p_9\})$$

Na opisani način su u cijelosti formalizirani parametri sustava čiju strukturu treba optimirati.

## 2.2 Formalizacija ciljne funkcije

Sustav djeluje onda, ako su procesi informacijski povezani klasama podataka. U **P/K** - matrici su te veze prikazane oznakama $\mathcal{K}$. Zato neka je definirana informacijska funkcija sustava $\mathcal{K}$ na slijedeći način:

$$\mathcal{RS} = \sum_{i=1}^{m} \sum_{j=1}^{n} \mathcal{R}_{i,j} \tag{13}$$

gdje je $\mathcal{R}_{i,j}$ informacijska funkcija elementa **P/K** -matrice, definirana za svaki element prema izrazu (14):

$$\mathcal{R}_{i,j} = \begin{cases} 0 \text{ za svaki } e_{i,j} = \text{ prazno} \\ z_{i,j} \text{ za svaki } e_{i,j} = \mathcal{K} \end{cases} \tag{14}$$

Oznake $z_{i,j}$ u prethodnom izrazu nazovimo **značaj informacijske veze**, koje u općem slučaju ovise o vrsti odnosa između procesa i klasa podataka. Vrsta tog odnosa može se zasebno istražiti, a u ovom radu će se uzeti da je to diskretna funkcija definirana na slijedeći način:

$$z_{i,j} = \begin{cases} 0,4 \text{ znači da } p_i \text{ treba } k_j \text{ kao } \textbf{dopunsku} \text{ informaciju} \\ 0,7 \text{ znači da } p_i \text{ treba } k_j \text{ kao } \textbf{važnu} \text{ informaciju} \\ 0,4 \text{ znači da } p_i \text{ treba } k_j \text{ kao } \textbf{neophodnu} \text{ informaciju} \end{cases}$$

Ovakva interpretacija značaja informacijske veze omogućava da se u odnos procesa i klasa podataka uključi potrebna metrika.

Ako je sustav nestrukturiran, kao na *slici 3a)*, onda svaki proces može u načelu biti u informacijskoj vezi sa svim ostalim procesima. Najveći broj veza tipa $\mathcal{K}$ može biti $v_{max} = m \cdot (n-1)$, dok je u stvarnosti on manji, tj. $v \leq v_{max}$. U strukturiranom sustavu, kao na *slici 3b)*, postoje dvije vrste informacijskih veza: **unutrašnje** $\mathbf{v_u}$, između procesa istog podsustava i **vanjske** $\mathbf{v_v = v - v_u}$, između različitih podsustava. Složenim sustavom, s velikim brojem procesa, teško se upravlja s jedne razine, pa se stoga koristi postupak organizacijskog strukturiranja. U ovom radu razmatra se slučaj **dvorazinskog strukturiranja** , kod kojeg se s gornje razine upravlja vezama između podsustava, dok se kontrola veza unutar podsustava prepušta donjoj razini. Manji broj nadziranih veza znači jednostavnije upravljanje sustavom. Ako se ovi zaključci primijene na **P/K** -matricu, slijedi da je bolja takva struktura sustava, kod koje je više veza smješteno unutar, a manje između podsustava.Zato ćemo prema izrazima *(12), (13) i (14)* , a sukladno oznakama na *slici 4* , *definirati informacijsku funkciju podsustava* $\mathcal{RP}$ na slijedeći način:

$$\mathcal{RP}_s = \sum_{i \in PS_s} \sum_{j=l_s}^{l_s+r_s} R_{i,j} \tag{15}$$

Ovdje je $\mathcal{PS}$ podskup svih procesa, koji prema trenutnoj strukturi pripadaju s-tom podsustavu. Varijable $l_s$ i $r_s$ uvedene u izraz *(15)* imaju slijedeće značenje:

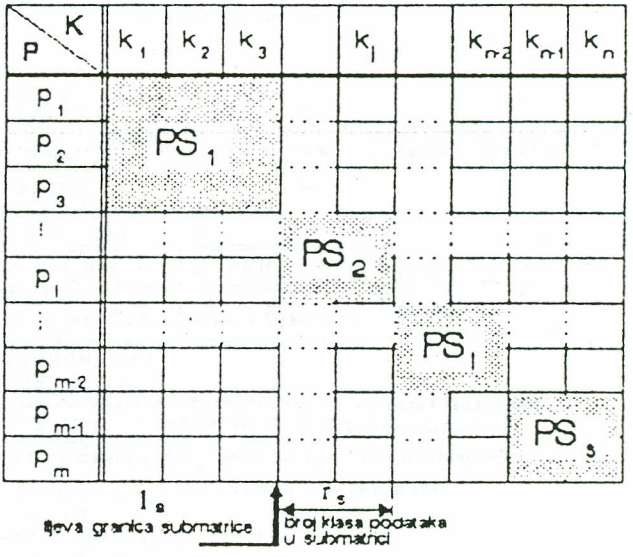
$l_s$ - redni broj stupca lijeve granice s-te submatrice (pokazuje na prvu klasu podataka koja joj pripada),

$r_s$ - broj klasa podataka koje pripadaju s -toj submatrici.

Za mjeru kvalitete strukture sustava odabrat ćemo funkciju $\mathcal{R}$, koja neka predstavlja razliku *informacijske funkcije sustava* i zbroja *informacijskih funkcija* svih njegovih pojedinačnih podsustava:

$$\mathcal{R} = \mathcal{RS} - \sum_{i=1}^{s} \mathcal{RP} \tag{16}$$

Na ovaj način iskazana je u prikladnom matematičkom obliku objektivna i kvantitativna mjera, pomoću koje se može ocijeniti kvaliteta svakog stanja strukture planiranog informacijskog sustava.




Slika 4: P/K-matrica i njezine submatrice

Na osnovi dosadašnjih razmatranja može se zaključiti da se ciljem optimizacije osnovne arhitekture informacijskog sustava pokazuje **nalaženje takve strukture koja će dati minimalnu vrijednost funkcije** $\mathcal{R}$:

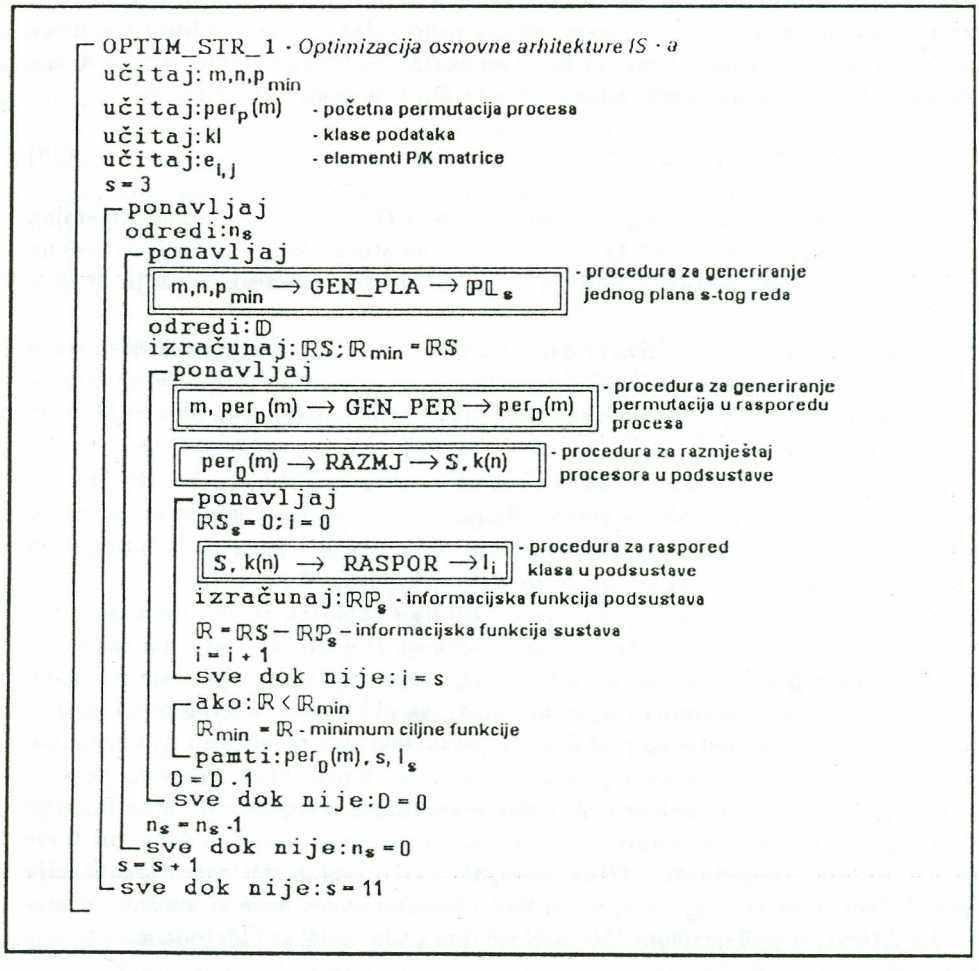
$$\mathcal{R} \Rightarrow \mathcal{R}_{min} \tag{17}$$

Provedena formalizacija prikladna je za izradu algoritama koji će se koristiti u računalnim programima za izvođenje optimizacije.

# 3 Određivanje optimalne strukture informacijskog sustava

Provedena formalizacija parametara i ciljne funkcije omogućava postavljanje temeljnog algoritma za optimizaciju osnovne arhitekture informacijskog sustava. Naredni

algoritam načelno je prikazan u obliku **akcijskog dijagrama**, uz neke dopune u prikazu složenih procedura. One su prikazane dvostrukim pravokutnicima, unutar kojih su shematski navedene ulazne varijable potrebne za rad procedure, ime procedure i njezine izlazne varijable.




Slika 5: Načelni algoritam za optimizaciju strukture IS-a

U prikazanom algoritmu korištene su oznake iz prethodnih razmatranja i uvedene dvije nove:

$$per_D(m) = p_{a,1}, p_{b,2}, \ldots, p_{\alpha,i} \cdots p_{z,m} \tag{18}$$

je uređen skup procesa, definiran u početnoj **P/K** -matrici, čiji se prvi slovčani indeks ($\alpha = a, b, ..., z$) odnosi na stalna imena procesa, a drugi brojčani indeks ($i = 1, 2, ..., m$) na redak koji zaposjeda određeni proces. Način zaposjedanja redaka od procesa može se smatrati jednom permutacijom od **m** elemenata. Sukladno ovome, svaka druga permutacija procesa, kojih prema izrazu *10* ima $\mathcal{D}_s$, označena je sa $per_D(m)$.

$$k(n) = k_{a,1}, k_{b,2}, ..., k_{\beta,j}, ..., k_{z,n} \tag{19}$$

je uređen skup klasa podataka, čiji se slovčani indeks ($\beta = a, b, ....z$) odnosi na stalna imena klasa, a drugi brojčani indeks ($j = 1, 2, ...n$) na stupac koji zaposjeda određena klasa podataka. Indeks stupca mijenja se nakon svakog prestrukturiranja **P/K**)-matrice.

Osnovna ideja algoritma zasniva se dakle na načelu sistematskog pretraživanja **svih** dozvoljenih planova kompozicije i **svih** permutacija u rasporedu procesa po podsustavima, te izračunavanja ciljne funkcije za svaku takvu kombinaciju, radi nalaženja njene najmanje vrijednosti.

Prikazani načelni algoritam je logički ispravan, ali je praktično iskoristiv za optimizaciju samo relativno malih sustava. Naime, druga petlja ponavlja se $\mathbf{n_S}$ puta, a treća $\mathcal{D}_s$ puta. Tako npr. za sustav: m=9, $3 \leq s \leq 4$ i $p_{min} = 2$ (primjer sa *slike 3a)* dozvoljeni planovi kompozicije su $\mathcal{PL}_3' = \{2, 2, 5\}$, $\mathcal{PL}_3'' = \{2, 3, 4\}$, $\mathcal{PL}_3''' = \{3, 3, 3\}$ i $\mathcal{PL}_4 = \{2, 2, 2, 3\}$, što prema izrazu *(10)* daje slijedeću shemu iteracija: $\mathcal{V} = 756 + 1260 + 1680 + 2520$. Za veće sustave vrijednost $\mathcal{D}$ enormno raste, pa tako npr. za m=20 i vektor plana kompozicije $\mathcal{PL}_5 = \{3, 3, 4, 4, 6\}$ treća petlja daje približno $163 \cdot 10^9$ iteracija. Za rješavanje ovog problema, tj. za oblikovanje takvog programskog algoritma koji bi bio iskoristiv za praktični projektantski rad, moguća su dva pristupa:

a) Treba pronaći kraći postupak pretraživanja mogućih permutacija procesa po podsustavima, po kojem bi se, polazeći od jedne permutacije i vrijednosti ciljne funkcije za nju, prešlo na iduću, koja sigurno daje manju vrijednost za $calR$, a da se pri tome ne pretražuju **sve** mogućnosti. Ovaj postupak traži programsku operacionalizaciju više raznih teorija za traženje optimuma nad permutacijama, koje su opisane u matematičkoj literaturi pod nazivom "Metoda vektora pada" u [4] te "Metoda adaptivnog pretraživanja" u [5] i [6].

b) Valja pojednostavniti algoritam, uvažavajući specifičnosti problema koji se rješava. To se odnosi na prirodu zavisnosti među procesima i na vezu između procesa i klasa podataka. Analiza **M/K** -matrice pokazuje da su neki procesi međusobno više povezani od drugih, a pripadnost klase podataka istom procesu ne mijenja se tijekom restrukturiranja matrice. To navodi na ideju da se za optimizaciju sustava koristi svojstvo "afiniteta među objektima", što je načelno opisano u [1]. Zato daljnja razmatranja u ovom radu imaju polazište u tzv. analizi afiniteta.

Klase podataka povezuju razne procese. Ako dva procesa $p_i$ i $p_j$ ne koriste nijednu zajedničku klasu podataka, smatramo da između njih ne postoji nikakav afinitet, odnosno da je njihov afinitet nula. Obrnuto, ako dva procesa imaju na ulazu sve iste klase podataka, smatrat ćemo da njihov međusobni afinitet ima vrijednost jedan. To su granični slučajevi. U općem slučaju, promatramo li dva procesa, oni na svom ulazu imaju više klasa podataka, od kojih neke mogu biti i zajedničke. Označimo li sa $n(p_i)$ broj ulaznih i izlaznih klasa podataka procesa $p_i$, a sa $n(p_i, p_j)$ broj zajedničkih klasa podataka koje imaju procesi $p_i i p_j$, onda možemo definirati *koeficijent afiniteta* $\mathcal{A}$ između ta dva procesa:

$$\mathcal{A}(p_i, p_j) = \frac{n(p_i,p_j)}{n(p_i)} \cdot 100\,[\%] \qquad (20)$$

Ovako izračunate koeficijente afiniteta možemo također upisati u odgovarajuću matricu međuzavisnosti procesa (kraće **P/P** -matrica), kako je to pokazano na *slici 6* za primjer sa *slike 3a* .

| **PROCESI** | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $P_7$ | $P_8$ | $P_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | = | 25 | 75 | 25 | 75 | 25 | 75 | 50 | 75 |
| $P_2$ | 25 | = | 50 | 100 | 50 | 50 | 50 | 75 | 25 |
| $P_3$ | 60 | 40 | = | 40 | 60 | 20 | 60 | 80 | 60 |
| $P_4$ | 25 | 100 | 50 | = | 50 | 50 | 50 | 75 | 25 |
| $P_5$ | 75 | 50 | 75 | 50 | = | 50 | 50 | 50 | 75 |
| $P_6$ | 33 | 67 | 33 | 67 | 67 | = | 33 | 33 | 33 |
| $P_7$ | 60 | 40 | 60 | 40 | 40 | 20 | = | 80 | 60 |
| $P_8$ | 40 | 60 | 80 | 60 | 40 | 20 | 80 | = | 40 |
| $P_9$ | 75 | 25 | 75 | 25 | 75 | 25 | 75 | 50 | = |

Slika 6: Matrica koeficijenata afiniteta

Koeficijenti afiniteta na *slici 6* računati su tako da su razmatrane sve veze (vrsta $\mathcal{K}$ i vrsta $\mathcal{G}$ ). U račun afiniteta također su uzete vrijednosti $z_{i,j} = 1$ prema izrazu *(14)* , ali se veze procesa i klasa podataka mogu označiti i drugim težinskim faktorima.

Ovako formirana **P/P** -matrica koristi se kao kriterij za grupiranje procesa. Naime, parovi procesa s visokim koeficijentom afiniteta čine jezgru podskupa procesa, koji ulaze u zasebne podsustave.[5] Izaberemo li u gornjem primjeru donju granicu koeficijenta afiniteta (kao kriterij za grupiranje) iznos od 80 % (tj. $\mathcal{A}_{i,j} = 80$), očito je da procesi $p_2$ i $p_4$ čine jezgru jednog, a procesi $p_3, p_7$ i $p_8$ drugog podsustava. Za svaki drugi proces, čiji je afinitet prema ostalim procesima ispod odabrane razine, treba odrediti može li se pridružiti nekoj postojećoj jezgri, ili pak je razina njegovog afiniteta

[5] U literaturi na engleskom jeziku ovakav se podskup naziva "cluster".

prema drugim procesima tako niska da može pripadati bilo kojem podsustavu. Za tu se procjenu koristi algoritam općenito opisan u [1] prema kojem se svi mogući parovi procesa najprije urede po padajućem slijedu koeficijenata afiniteta, a zatim se odrede jezgre podsustava. Želimo li procijeniti treba li neki novi proces $p_r$, čiji je najviši koeficijent afiniteta idući u padajućem nizu, priključiti postojećem jezgru podsustava s kojeg čine procesi $p_m$ i $p_n$, treba izračunati njegov koeficijent afiniteta $\mathcal{A}$ prema procesima, koji su u taj podsustav već ranije uključeni:

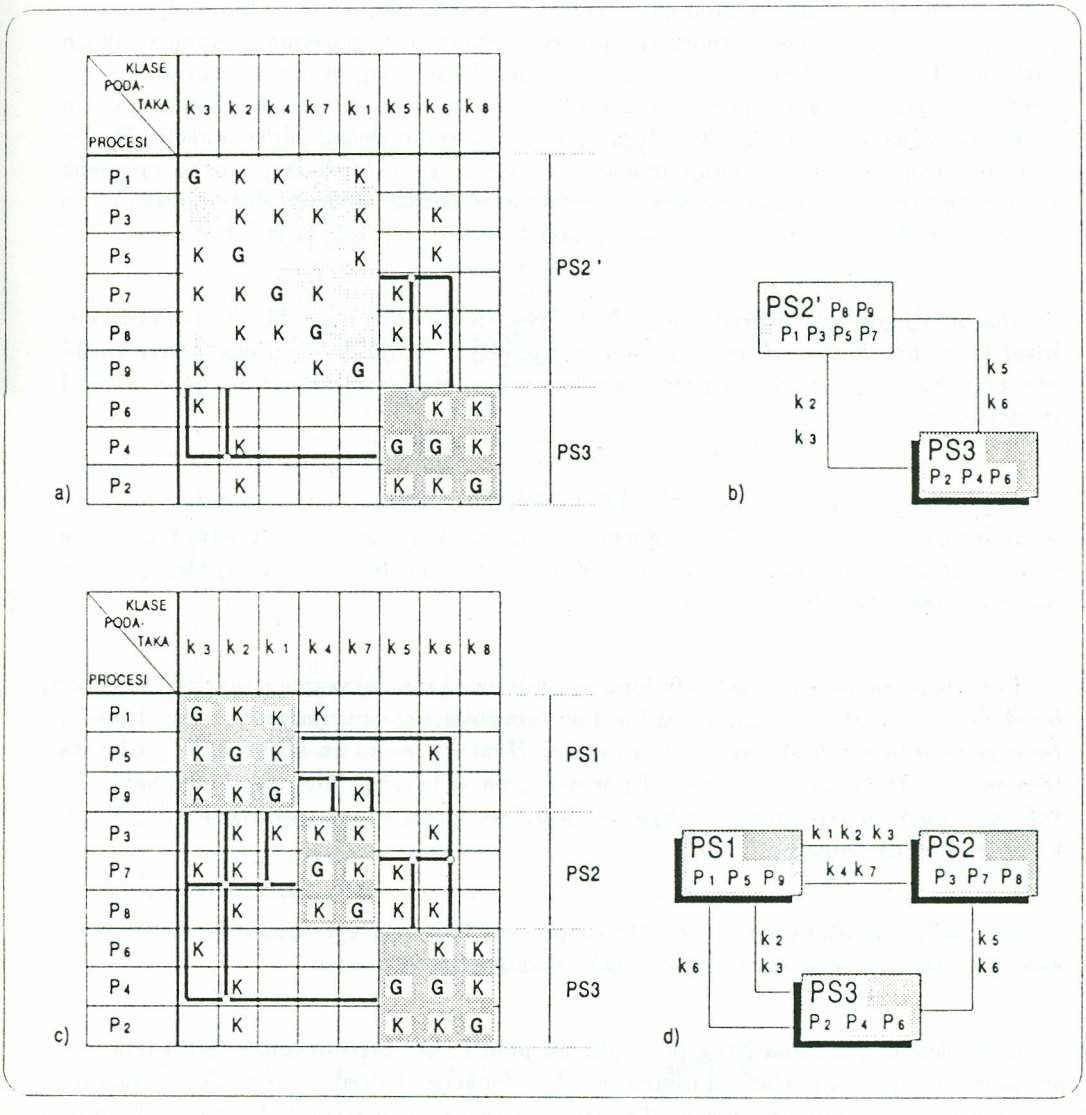
$$\mathcal{A}_s(p_r, s) = \frac{\mathcal{A}(p_r,p_m)\cdot n(p_m)+\mathcal{A}(p_r,p_n)\cdot n(p_n)}{n(p_m)+n(p_n)} \qquad (21)$$

Ako je ovako izračunati koeficijent viši od idućeg po redu, treba proces $p_r$ uključiti u podsustav s , dok se u protivnom smještava u posebni podsustav, zajednički za sve procese slabog afiniteta. Za razmatrani primjer sa *slike 3a)* , uz korištenje matrice afiniteta na *slici 6* , provedena je na taj način dekompozicija sustava na dva podsustava, te je dobivena slijedeća struktura: $\mathcal{S} = \{\mathcal{PS}_3\{p_2, p_4, p_6\}, \mathcal{PS}'_2\{p_1, p_3, p_5, p_7, p_8, p_9\}\}$.

Odgovarajući postupak i dobiveni rezultat prikazani su *slikama 7a i b* . Na opisani način skup od m procesa bit će općenito dekomponiran na s podskupova. Hipoteza **H1** glasi:

**H1: Ciljna funkcija $\mathcal{R}$ za sustav m-tog reda, sastavljen od s-podsustava oblikovanih po kriteriju maksimalnog afiniteta među procesima, ima minimalnu vrijednost za zadane $s$ i $p_{min}$.** .

Prepuštajući izvođenje egzaktnog dokaza matematičkoj znanosti, provjerimo ispravnost gornje hipoteze na razmatranom praktičnom primjeru: Na osnovi razmatranja u točki 2.2, uz $z_{i,j} = 1$, vrlo jednostavno se može izračunati da vrijednost ciljne funkcije za sustav strukturiran kao na *slici 3* iznosi: $\mathcal{R}_1 = 24$. Za taj isti sustav, strukturiran kao na *slikama 7a i b* , ciljna funkcija poprima vrijednost $\mathcal{R}_2 = 8$, te je očito $\mathcal{R}_2 < \mathcal{R}_1$.




Slika 7: Strukturiranje sustava po kriteriju afiniteta među procesima

U općem slučaju neki podskupovi, određeni po kriteriju afiniteta, sadrže procese čiji su koeficijenti afiniteta premali za stvaranje novog jezgra ili priključivanje nekom postojećem jezgru, odnosno imaju više procesa nego što je to poželjno iz organizacijskih razloga. Takve podskupove treba dalje dijeliti do željenog stupnja kompozicije s . Kod tog dijeljenja će se, prema izrazu *(16)* , smanjiti zbroj informacijskih funkcija podsustava (jer više veza $K$ ostaje izvan njih), odnosno povećati ciljna funkcija $\mathcal{R}$. Da bi to povećanje bilo što je manje moguće, u ovom se radu predlaže *ponovna primjena analize afiniteta za submatricu koja predstavlja podsustav dobiven dekompozicijom u prethodnom koraku* . Temeljem toga postavlja se hipoteza **H2**, koja glasi:

**H2: Daljnja dekompozicija podsustava, dobivenih u prethodnom koraku, imat će za posljedicu najmanje povećanje ciljne funkcije cijelog sustava onda ako se također provede uz primjenu kriterija analize afiniteta na razmatrani podsustav.**

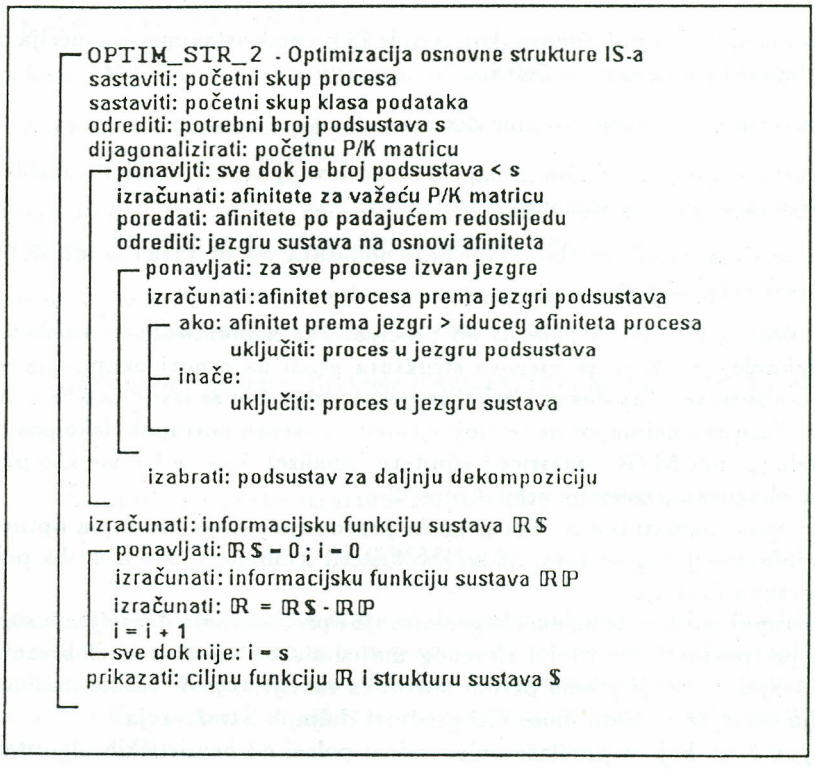
Provjerimo ispravnost hipoteze **H2** empirijski, na razmatranom primjeru: ako se podsustav $\mathcal{P}S_2'$ na *slici 7a* ponovno dekomponira po opisanom postupku, dobije se rješenje prikazano *slikama 7c i d* . Vrijednost ciljne funkcije za taj slučaj iznosi $\mathcal{R}_3 = 15$, dakle vrijedi: $\mathcal{R}_3 < \mathcal{R}_1$.

Premda ispravnost postavljenih hipoteza nije egzaktno dokazana, empirijski zaključci, izvedeni na razmatranom primjeru kao i na opsežnim istraživanja provedenim tijekom razvoja strukture velikih sustava (autor ovog članka je radio na više velikih projekata IS-a sa $m > 70$, $p_{min} > 5$ i $s > 8$ ) pokazuju, da se izložene hipoteze mogu smatrati važećim. Najveća vrijednost postupka, koji polazi od valjanosti postavljenih hipoteza, nalazi se u dvije činjenice:

a) Analiza afiniteta vrlo brzo otkriva procese koji su jezgre budućih podsustava, kao i procese koji se mogu priključiti tako zasnovanim jezgrama.

b) Postupna primjena istog postupka na podsustave brzo dekomponira sustav do željenog stupnja i daje relativni minimum ciljne funkcije. Određivanje strukture sustava s apsolutnim minimumom nije doduše riješeno egzaktno matematički, ali oblikovanje IS-a u većini praktičnih slučajeva zavisi i od takvih čimbenika koje nije moguće obuhvatiti teoretskim modelom.
Izloženi postupak optimiranja strukture IS-a u općim crtama pokazan je na *slici 8* u

obliku akcijskog dijagrama.




Slika 8: Optimizacija strukture sustava na osnovi analize afiniteta

Postupak optimizacije strukture sustava na *slici 8* predstavlja opću shemu, opisanu akcijskim dijagramom i strukturiranim hrvatskim jezikom. Na osnovi toga se može razviti cjelovita programska podrška za primjenu ovog postupka korištenjem računala.

# 4 Zaključak

Postavljanje optimalnog ustrojstva jedan je od ključnih problema, koje treba rješavati na samom početku razvoja novog informacijskog sustava. Propusti, učinjeni u toj fazi, nepovoljno se odražavaju na iduće faze životnog ciklusa IS-a, a iskazuju se u nizu teškoća, kao npr.:

- otežan je rad razvojnih timova, koji izvode IS po podsustavima, jer sučelja njihove suradnje ostaju nejasno definirana;
- povećavaju se troškovi i vrijeme razvoja;
- podsustavi novog IS-a nisu prilagođeni poslovnoj tehnologiji i organizacijskom ustrojstvu objektnog sustava;
- loše i neodgovarajuće su oblikovane baze podataka, što otežava i usporava funkcije IS-a kod korištenja itd.

Razvoj svakog IS-a mora polaziti od procesa, kao nepromjenljivih dijelova svake poslovne tehnologije. Zato se njegova struktura gradi na osnovi okupljanja srodnih procesa u podsustave. Taj složen i osjetljiv posao uglavnom se izvodi na bazi intuicije ili iskustva. Za njega naime još ne postoji cjelovit i povezan postupak, iako postoje polazne metode (poput **M/K** -matrice i afinitetne analize), koje se koriste kao pomoćno sredstvo za oblikovanje osnovne arhitekture IS-a.

U ovom radu razmatrana su dva moguća pravca rješavanja problema optimizacije ustrojstva informacijskog sustava, ali se kao kriterij kvalitete rješenja u oba postupka koristi ista ciljna funkcija.

Prvi postupak zasniva se na načelu generiranja i pretraživanja dozvoljenih struktura sustava, a pretpostavlja upotrebu složenog matematskog aparata za dobivanje optimalnog rješenja. Rješenje prema prvom postupku razvijeno je do razine matematskog modela, čija detaljna razrada može biti predmet daljnjih istraživanja.

Drugi postupak koji se predlaže ovim radom polazi od heurističkih algoritama, za koje je empirijski pokazana njihova valjanost. Odlika ovog rješenja je relativno laka provedivost kod praktične upotrebe u svakodnevnom projektantskom radu. Nedostatak egzaktnog dokaza za optimalnost rješenja nadomješten je mogućnošću brzog dobivanja jednog rješenja koje je blizu optimalnom, te jednostavnom provjerom vrijednosti ciljne funkcije za skup rješenja koja su u bližoj okolici početnog.

Provedena istraživanja pokazuju da se intuitivno i iskustveno određivanje ustrojstva složenog IS-a može uspješno zamijeniti strukturiranim pristupom problemu, tražiti i naći optimalno rješenje s obzirom na zadanu ciljnu funkciju uz dopustiv utrošak vremena i tako u velikoj mjeri izbjeći lošu polaznu arhitekturu u projektu novog informacijskog sustava.

# Literatura

[1] Martin, J.: Information Engineering, Book II: Planning and Analysis, Prentice-Hall, Inc.; Englewood Cliffs, 1989.

[2] Finkelstein, C.: Introduction to Information Engineering, Addison-Wesley Publishing Company; New York 1989.

[3] Radovan, M.: Projektiranje informacijskih sistema, Informator; Zagreb 1991.

[4] Sergienko, I.V.: O primenenii vektora spada dlja rešenija zadač, Upravlenie sistemi i mašini, 1975./3, str. 86-94.

[5] Stojan Ju.G.: Ob odnom sposobe poiska nailuščega rešenija dlja ednogo klasa mnogoekstrimalnih zadač, Upravlenie sistemi, 1969./5, str. 37-44.

[6] Lončar, J.: Algoritmi za određivanje optimalnog rješenja problema smještaja figura datog oblika u zadanu figuru, Doktorska disertacija, PMF Sarajevo 1982.

[7] IBM Corporation: Business Systems Planning, IBM Marketing Publications, 1984.

Primljeno: 1993-09-10

Brumec J. Optimization of the structure for complex information systems

## SUMMARY

Mistakes made in the starting stage of development increase financial investments and decrease the benefits after implementation of the new information system. Therefore, in this work has been analysed problems and possibilities in design of the optimal architecture of IS, proposed an other approach, and, given one new algorithm for this optimisation.