

# INTELIGENTNI SUSTAVI

---

Prof. dr. sc. Božidar Kliček  
Sveučilište u Zagrebu  
Fakultet organizacije i informatike, Varaždin

# Nesigurnost

---

Bayesov teorem, Bayesove vjerojatnosne mreže, Dempster  
Shafer, neizraziti sustavi

# Ograničenje klasične logike

---

- **Klasična logika smatra** da su svi iskazi bilo istiniti ili lažni. U stvarnom životu takve situacije najčešće ne postoje.
-

# Nesigurne informacije

---

Definicija pojam nesiguran (engl. *uncertain*):

- koji nije sigurno poznat; upitan, problematičan.
  - neodređen; koji nije konačan ili definiran.
  - dvojben; nema sigurnog znanja; nije siguran.
  - dvosmislen.
  - koji nije stalan ili stabilan; promjenljiv.
  - podložan promjeni ili promjenjiv; nepouzdan ili koji nije vjerodostojan (**Websterov rječnik**).
-

# Neodređenost i dvosmislenost

---

Neodređenost (engl. *vagueness*)

- poteškoća određivanja oštrog i preciznog razlučivanja u svijetu, gdje se stvari ne mogu odijeliti oštrim granicama.

Dvosmislenost (engl. *ambiguity*):

- postojanje relacija jedan-više; situacije u kojima je izbor između jedne ili više alternativa manje određen.
-

# Razlozi nastajanja nesigurnosti

---

- nepouzdani podaci** - zbog načina na koji se prikupljaju, podaci mogu biti nepouzdani ili puni grešaka.
  - nepotpuni podaci** - ponekad se traži djelovanje na temelju djelomičnog podataka ili neki podaci nedostaju.
  - neprecizni podaci**: koji su samo približno poznati.
-

# Uključivanja nesigurnosti u sustave temeljene na znanju

---

1. Kako **brojčano** izraziti naše uvjerenje da je neka informacija točna ili ne?
  2. Kako izraziti jačinu utjecaja **pojedinog dokaza** na stvaranje **zaključka**?
  3. Kako više **neovisnih dokaza** utječe na stvaranje konačnog zaključka?
  4. Kako postići **konačno rješenje** ako se ono temelji na većem broju **koraka zaključivanja** s prisutnom nesigurnošću?
-

# Pregled faktora sigurnosti

---

- Bayesova statistika
  - Funkcije vjerovanja
  - Mreže vjerovanja
  - Fuzzy logika
  - Faktori sigurnosti
-

# Pristup temeljen na vjerojatnosti

---

**Subjektivna vjerojatnost** - prikazuje vjerovanje neke osobe.

**Objektivna vjerojatnost** temeljena na matematičkim odnosima.

Definiranje vjerojatnosti:

- odnos
  - relativna frekvencija
  - vjerovanje.
-

# Bayesov teorem

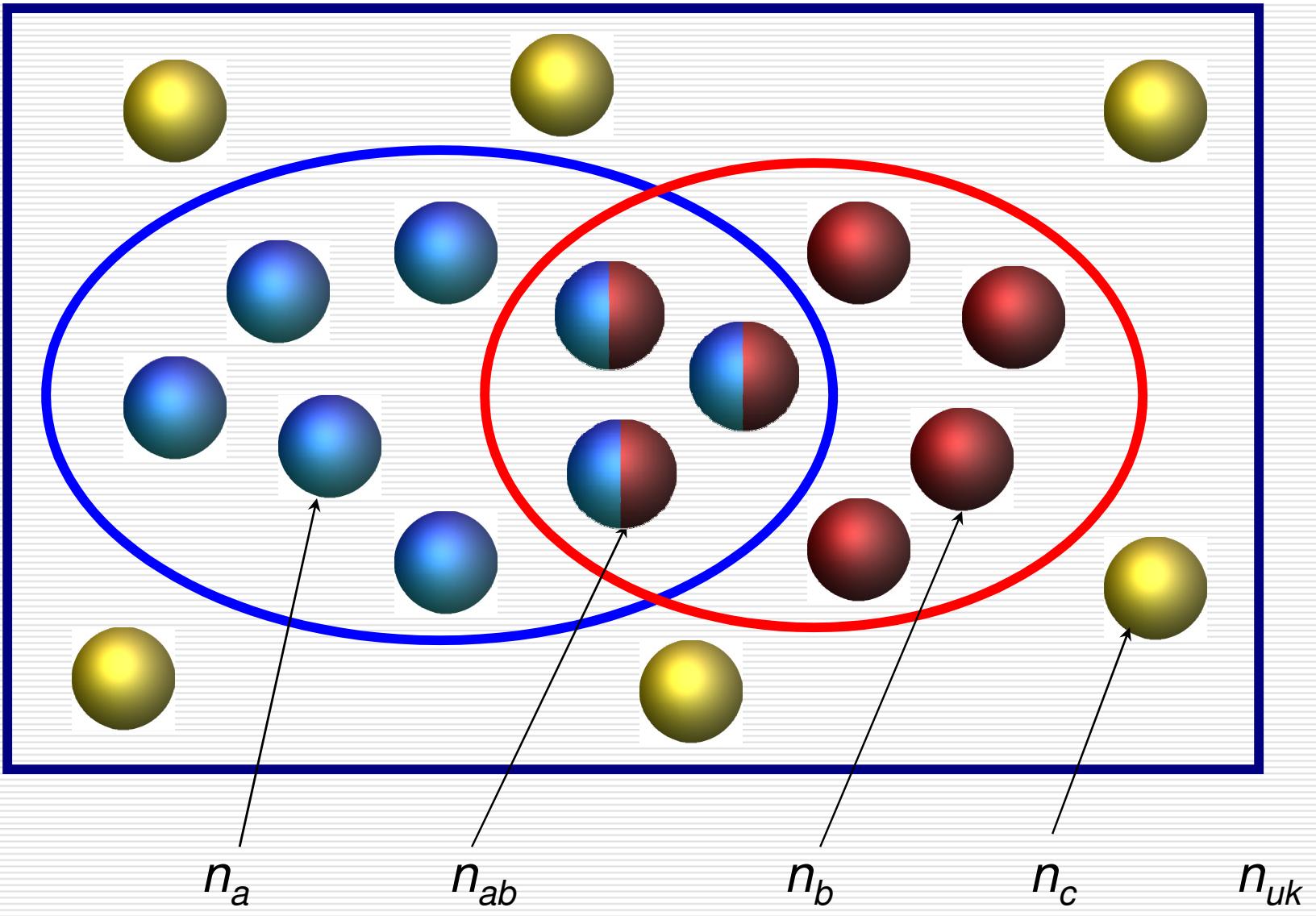
---

## Primjer.

Imamo kutiju s lopticama koje su označene s "a", "b", "ab" i "c".

Kolika je vjerojatnost da iz kutije nasumice izvučemo lopticu koja ima na sebi oznaku "a" (ne obazirući se pri tom na mogućnost postojanja i drugih oznaka, npr. "b").

---



# Bayesova formula

---

- Kako možemo odrediti vjerojatnost nekog događaja A, ako znamo da je nastupio događaj B?
- Oznake:  $A$  – događaj,  $B$  – događaj;  $p(A)$  i  $p(B)$  su vjerojatnosti događaja  $A$  i  $B$ ;  $p(A|B)$  je vjerojatnost događaja  $A$ , ako je nastupio događaj  $B$ .
- Bayesova formula uvjetne vjerojatnosti:  
$$p(A|B) = p(B|A) * p(A) / p(B).$$



Thomas Bayes, engleski matematičar (1702-1761)

## Primjer. Dijagnoza bolesti

---

- Kako možemo odrediti vjerojatnost određene dijagnoze, ako su prisutni određeni simptomi.

- Oznake:

$S$  – simptom

$D$  – dijagnoza

- Bayesova formula poprima oblik:

$$P(D|S) = P(S|D) * P(D) / P(S)$$

## Primjer. Dijagnoza bolesti

---

- Neki statistički podaci pokazuju slijedeće:
  - $P(\text{bronhitis}) = 0.05$
  - $P(\text{kašalj}) = 0.2$
  - $P(\text{kašalj} \mid \text{bronhitis}) = 0.8.$
- Ako neki bolesnik kašlje, kolika je vjerojatnost bronhitisa?
  - $$P(D|S) = P(S|D) * P(D) / P(S) = 0.8 * 0.05 / 0.2 = 0.2$$

# Primjena Bayesovog teorema

---

- Bayesov teorem je bio uzor za niz metoda koje određuju dijagnozu preko ovog tipičnog postupka:
    - Odrediti *a priorne* dijagnoze svih dijagnoza.
    - Na temelju svakog simptoma prilagoditi vjerojatnosti svih dijegnoza na temelju vjerojatnosti simptom-dijagnoza.
    - Odabratи najvjerojatniju dijagnozu.
-

# Općí oblik Bayesove formule

---

$$P(D_i / S_1 \wedge S_2 \wedge \dots \wedge S_m) = P(S_1 \wedge S_2 \wedge \dots \wedge S_m / D_i) \frac{P(D_i)}{P(S_1 \wedge S_2 \wedge \dots \wedge S_m)}$$

$$P(D_j / S_1 \wedge S_2 \wedge \dots \wedge S_m) = P(S_1 \wedge S_2 \wedge \dots \wedge S_m / D_j) \frac{P(D_j)}{P(S_1 \wedge S_2 \wedge \dots \wedge S_m)}$$

---

# Tablica praćenja dijagnoza i simptoma

---

	$S_1$	$S_2$	...	$S_m$	$D_1$	$D_2$	...	$D_n$
$C_1$	+	+			+			
$C_2$		+		+		+		
...								
$C_N$		+		+				+

---

# Uvjeti primjene Bayesove formule

---

Uvjeti primjene Bayesovog teorema:

- Simptomi ne ovise o dijagnozi i međusobno su nezavisni.
  - Potpunost skupa dijagnoza.
  - Međusobno isključivanje dijagnoza.
  - Postojanje ispravne i potpune statistike za izvođenje *a priornih* vjerojatnosti dijagnoza i uvjetnih vjerojatnosti simptom dijagnoza.
  - Konstantnost vjerojatnosti.
-

# Prevladavanje ograničenja

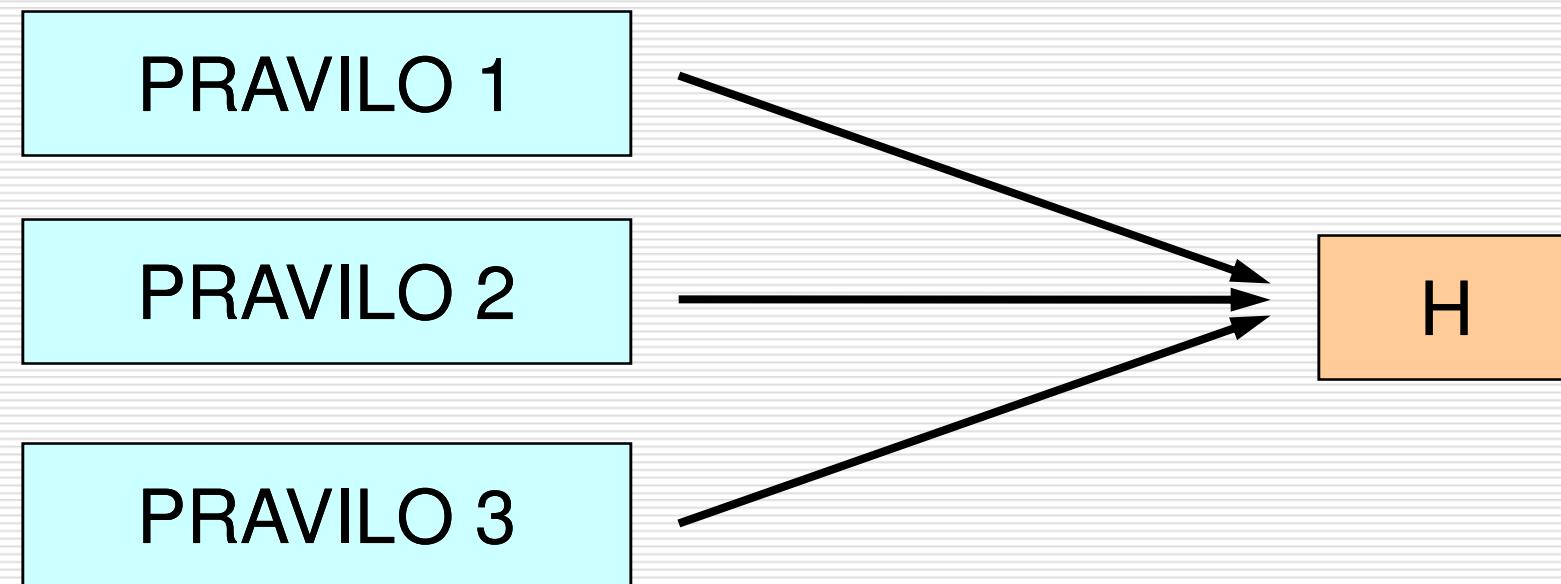
---

Prevladavanje ograničenja Bayesove formule u primjeni:

- Kombinacije simptoma ostvaruju jednostavniji modeli poput MYCINA.
  - Ne može se procijeniti pouzdanost rezultata (to rješava Dempster-Shaferova teorija).
  - Simptomi koji potvrđuju dijagnozu X%, protiv (100 - X%). Prihvatljivije je vrednovati pozitivne i negativne dokaze odvojeno.
-

# Više pravila potvrđuje istu hipotezu

---



# Bayesove mreže

---

Primjer primjene Bayesovog teorema u intelligentnim sustavima

# Prevladavanje ograničenja

---

Prevladavanje ograničenja Bayesove formule u primjeni:

- Kombinacije simptoma ostvaruju jednostavniji modeli poput MYCINA.
  - Ne može se procijeniti pouzdanost rezultata (to rješava Dempster-Shaferova teorija).
  - Simptomi koji potvrđuju dijagnozu X%, protiv (100 - X%). Prihvatljivije je vrednovati pozitivne i negativne dokaze odvojeno.
-

## Osnovna svojstva Bayesovih mreža

---

- Bayesova mreža je grafički model koji prikazuje vjerojatnosne ovisnosti među varijabalama koje se razmatraju.
  - Vrlo su korisne metode učenja parametara i strukture Bayesovih mreža, koje također nude učenje s nepotpunim podacima.
  - Bayesove tehnike mogu se koristiti za nadgledano i nenadgledano učenje, te daju grafički prikaz ovisnosti varijabli u nekom sustavu.
-

# Osnovna svojstva Bayesovih mreža

---

- Kada se koriste zajedno sa statističkim tehnikama, grafički model ima nekoliko prednosti za analizu podataka:
    - 1) Kada model kodira ovisnosti između svih varijabli, on može rješavati situacije kada nedostaju neki unosi podataka.
    - 2) Bayesove mreže se mogu koristiti za učenje uzročnih ovisnosti, te se stoga mogu koristiti da poboljšaju razumijevanje o problemskoj domeni i predviđaju posljdice nekog djelovanja (npr. u poslovnim, društvenim, ekološkim i sličnim sustavima).
    - 3) Zbog toga što model ima i uzročnu i vjerojatnosnu semantiku, model je idealan prikaz kombinacijom prethodnog znanja (koji često dolazi u uzročnom obliku) i podataka.
    - 4) Bayesove statističke metode povezane s Bayesovim mrežama nude učinkovit i principjelan pristup za izbjegavanje pretreniranosti podataka.
-

## Definicija Bayesovih mreža

---

- Bayesova mreža, mreža vjerovanja ili usmjereni aciklički grafički model je vjerojatnosni grafički model koji prikazuje skup slučajnih varijabli i njihovu uvjetnu međusobnu ovisnost preko usmjerenih acikličkih grafova (*directed acyclic graph*, DAG).
  - Bayesova mreža može prikazivati vjerojatnosne odnose između bolesti i simptoma. Za dana simptome, mreža se može koristiti za izrčunavanje vjerojatnosti prisutnosti različitih bolesti.
-

# Definicija Bayesovih mreža

---

- Čvorovi Bayesove mreže prikazuju slučajne varijable: neke mjerljive veličine, sakrivene varijable, nepoznate parametre ili hipoteze. Lukovi prikazuju uvjetne ovisnosti među varijablama.
  - Čvorovi koji nisu povezani prikazuju varijable koje su međusobno uvjetno nezavisne.
  - Svaki čvor je povezan s funkcijom vjerojatnosti koji dobiva kao ulaze skup vrijednosti čvorova roditelja i daje kao rezultat vjerojatnost varijable prikazane čvorom.
  - Postoje učinkoviti algoritmi koji omogućuju učenje i zaključivanje u Bayesovim mrežama. Bayesove mreže mogu modelirati niz varijabli (npr. kod govornog signala ili niz u bjelančevinama) i tada se nazivaju dinamičke Bayesove mreže. Bayesove mreže mogu prikazivati i rješavati probleme odlučivanja i tada se nazivaju dijagrami utjecaja.
-

# Definicija Bayesovih mreža

---

- Neka je  $G = \{V, E\}$  usmjereni aciklički graf (ili *directed acyclic graph*, DAG), i neka je  $X = \{X_v\}_{v \in V}$  skup slučajnih varijabli indeksiranih sa  $V$ .
- **Definicija faktorizacije.**  $X$  je Bayesova mreža u odnosu na  $G$  ako se njena zajednička funkcija gustoće vjerojatnosti (u odnosu na mjeru produkta), može pisati kao produkt pojedinačnih fukcija gustoće, uvjetno prema njihovim roditeljskim varijablama.

$$p(x) = \prod_{v \in V} p(x_v | x_{\text{pa}(v)})$$

- Gdje je  $\text{pa}(v)$  je skup roditelja od  $v$  (tj. tih čvorova čiji čvorovi direktno pokazuju na  $v$  preko jednog luka).
-

# Definicija Bayesovih mreža

---

- Za bilo koji skup slučajnih varijabli, vjerojatnost zajedničke distribucije bilo kojeg čvora se može izračunati preko uvjetne vjerojatnosti korištenjem pravila ulančavanja kako slijedi:

$$P(X_1 = x_1, \dots, X_n = x_n) = \prod_{v=1}^n P(X_v = x_v \mid X_{v+1} = x_{v+1}, \dots, X_n = x_n)$$

- to se također može prikazati kao:

$$P(X_1 = x_1, \dots, X_n = x_n) = \prod_{v=1}^n P(X_v = x_v \mid X_j = x_j)$$

za svakog  $X_j$  koji je roditelj od  $X_v$

- Razlika između ovih izraza je u uvjetnoj nezavisnosti varijabli od njihovih ne-potomaka, s danim vrijednostima njihovih roditeljskih varijabli.
-

## Lokalna Markovljeva svojstva

---

- $X$  je Bayesova mreža u odnosu na  $G$  ako zadovoljava *lokalna Markovljeva svojstva*, gdje je svaka varijabla uvjetno nezavisna od ne-potomaka danih njenim roditeljskim varijablama.

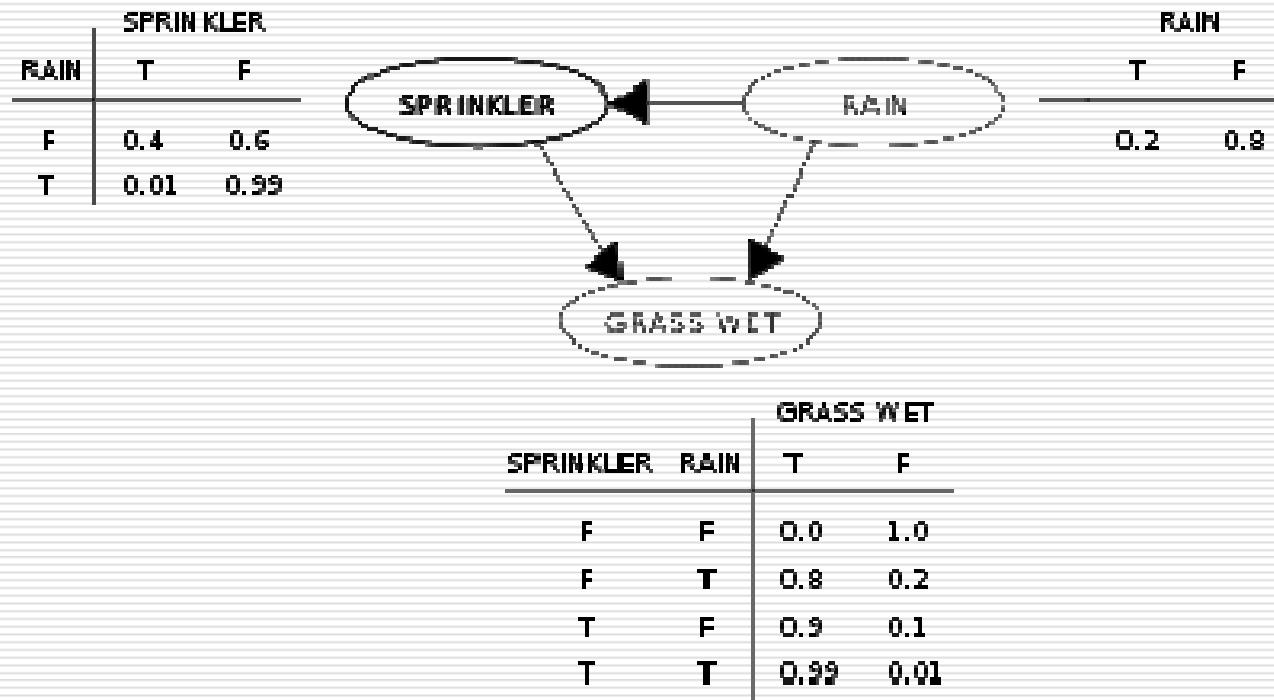
$$X_v \perp\!\!\!\perp X_{V \setminus \text{de}(v)} \mid X_{\text{pa}(v)} \quad \text{for all } v \in V$$

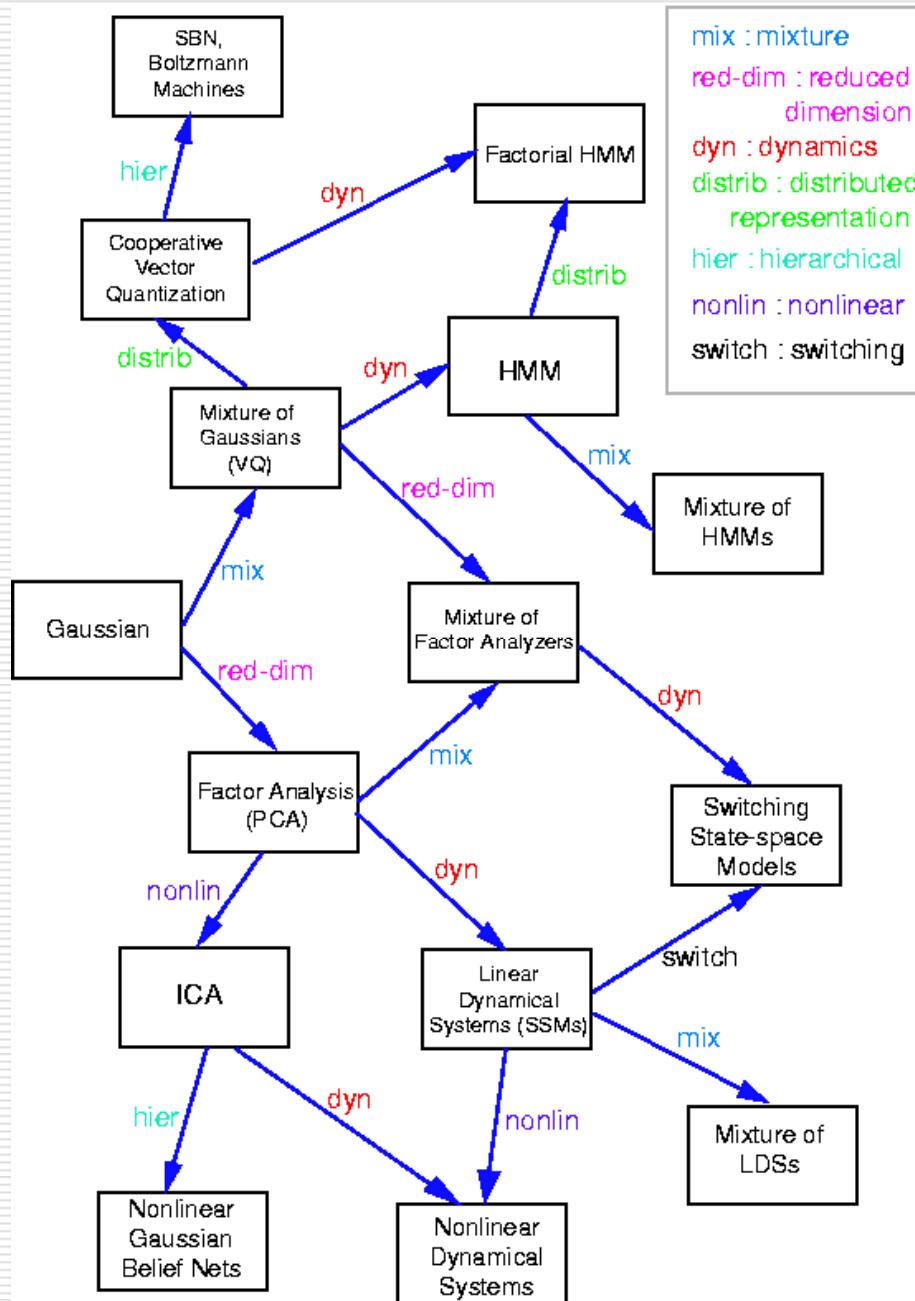
- gdje je  $\text{de}(v)$  je skup prethodnika od  $v$ .

$P(X_v = x_v \mid X_i = x_i \text{ for each } X_i \text{ which is not a descendent of } X_v) = P(X_v = x_v \mid X_j = x_j \text{ for each } X_j \text{ which is a parent of } X_v)$

---

# Primjer





## □ Ovisnost linearnih modela

mix : mixture  
 red-dim : reduced dimension  
 dyn : dynamics  
 distrib : distributed representation  
 hier : hierarchical  
 nonlin : nonlinear  
 switch : switching

# Dempster-Shaferova teorija vjerovanja

---

Primjer primjene Dempster-Shaferove teorije

# Načini rada s nesigurnošću

---

- Bayesova formula za uvjetnu vjerojatnost
  - Faktori sigurnosti (MYCIN i Guru)
  - Dempster-Shaferova teorija vjerovanja
  - Zadehovi neizraziti skupovi
-

# Osnovno o teoriji vjerovanja

---

## Razlozi uvođenja teorije vjerovanja

- Bayesova teorija ne zna dovoljno dobro opisati neznanje.
  - Shafer: u sličnim se slučajevima ne pravi nikakva razlika između nedefiniranosti (nedovoljnog znanja) i jednakog stupnja uvjerenosti.
  - Dempster-Shaferova teorija modelira proces odbacivanja hipoteza zbog prikupljanja ocjena koeficijenata uvjerenosti.
-

# Osnovno o teoriji vjerovanja

---

Uvode se dvije funkcije:

- mjera vjerovanja (*belief*):  $Bel(A)$
  - mjera uvjerljivosti (*plausibility*):  $Pl(A)$
  
  - Njihove karakteristike su dane aksiomima od 1 do 5 (vidi predavanje).
-

## Osnovne karakteristike funkcija $Bel$ i $Pl$

---

- $Bel : \mathcal{P}(X) \rightarrow [0, 1]$
- $Bel(A) + Bel(A^c) \leq 1$
- $Pl(A) = 1 - Bel(A^c)$  za sve  $A \in \mathcal{P}(X)$
- $Bel(A) = 1 - Pl(A^c)$
- $Pl(A) + Pl(A^c) \geq 1$

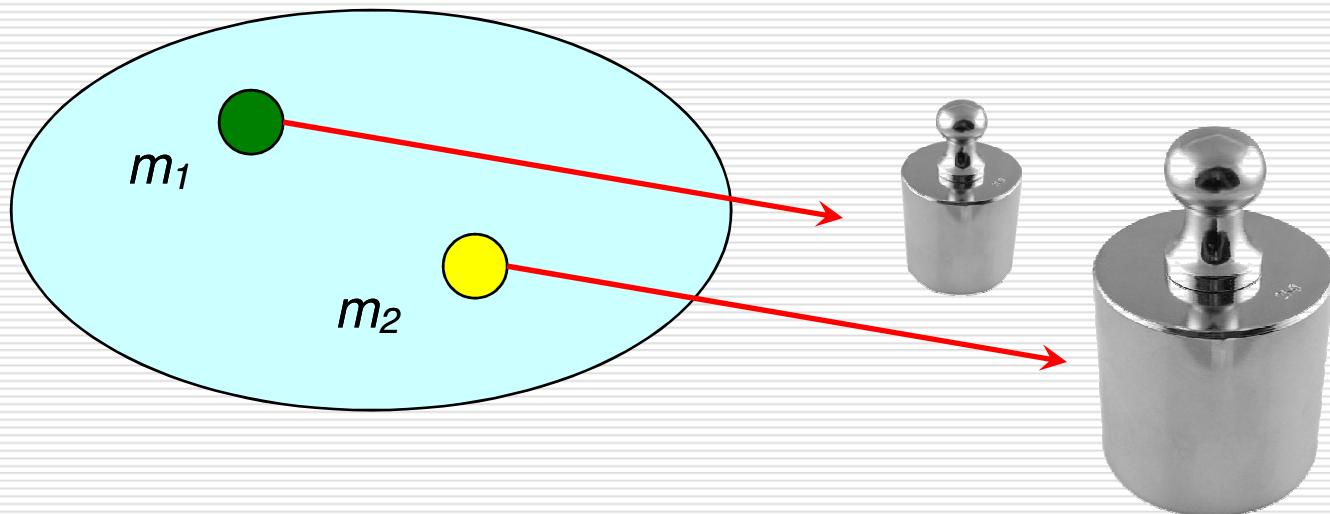
$\mathcal{P}(X)$  je partititivni skup od  $X$ , skup svih podskupova.  $Bel(A)$  je funkcija vjerovanja u  $A$  (*belief*).  $A^c$  je komplement od  $A$ .  $Pl(A)$  je uvjerenost u  $A$  (*plausibility*).

---

## Stupanj vjerovanja u dokaz

---

- U D-S teoriji stupanj vjerovanja u dokaz analogan je masi  $m$  fizičkog objekta:
- Svaki skup u okolini koji ima masu veću od 0 je *fokalni element*.

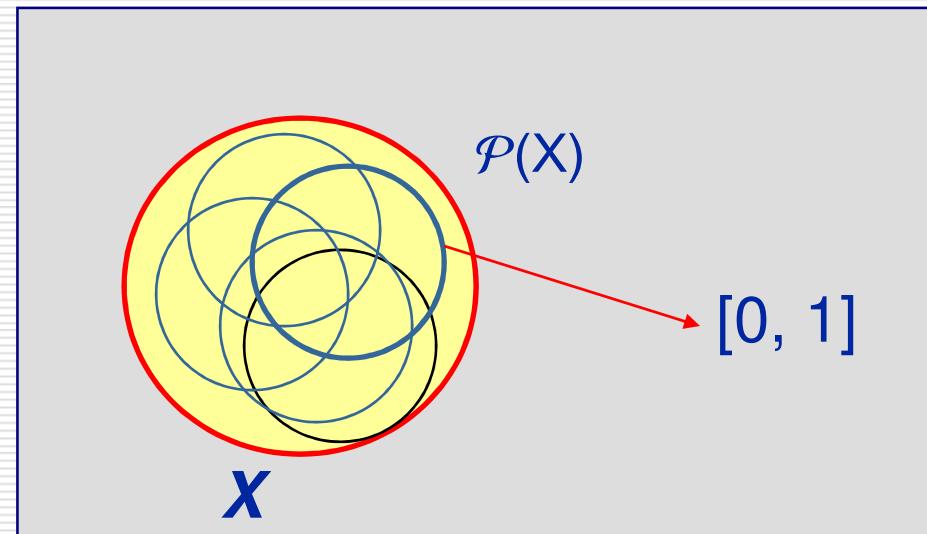


# Temeljna dodjela vjerojatnosti

---

- Svaka se mjera vjerovanja i uvjerljivosti definira pomoću funkcija *temeljne dodjele vjerojatnosti* ili *temeljne dodjele m*:
- $m : \mathcal{P}(X) \rightarrow [0, 1]$
- $m(\emptyset) = 0$
- $\sum m(A) = 1$

$B \cap A \neq \emptyset$



## Ovisnost $Bel(A)$ i $PI(A)$

---

- Ovisnost  $Bel(A)$  i  $PI(A)$  o temeljnim dodjelama  $m$
  - $Bel(A) = \sum_{B \subseteq A} m(B)$
  - $PI(A) = \sum_{B \cap A \neq \emptyset} m(B)$
  - $PI(A) \geq Bel(A)$
-

# Dempster-Shaferovo pravilo kombinacije

---

- Dempster-Shaferovo pravilo kombinacije
  - $m_{1,2}(A) = \frac{\sum_{B \cap C = A} m_1(B) \times m_2(C)}{1 - K}$
  - $K = \sum_{B \cap C = \emptyset} m_1(B) \times m_2(C)$
  - $m_1$  i  $m_2$  su temeljne dodjele od nezavisnih izvora, a  $m_{12}$  je temeljna dodjela nastala njihovom kombinacijom
-

# Primjer

---

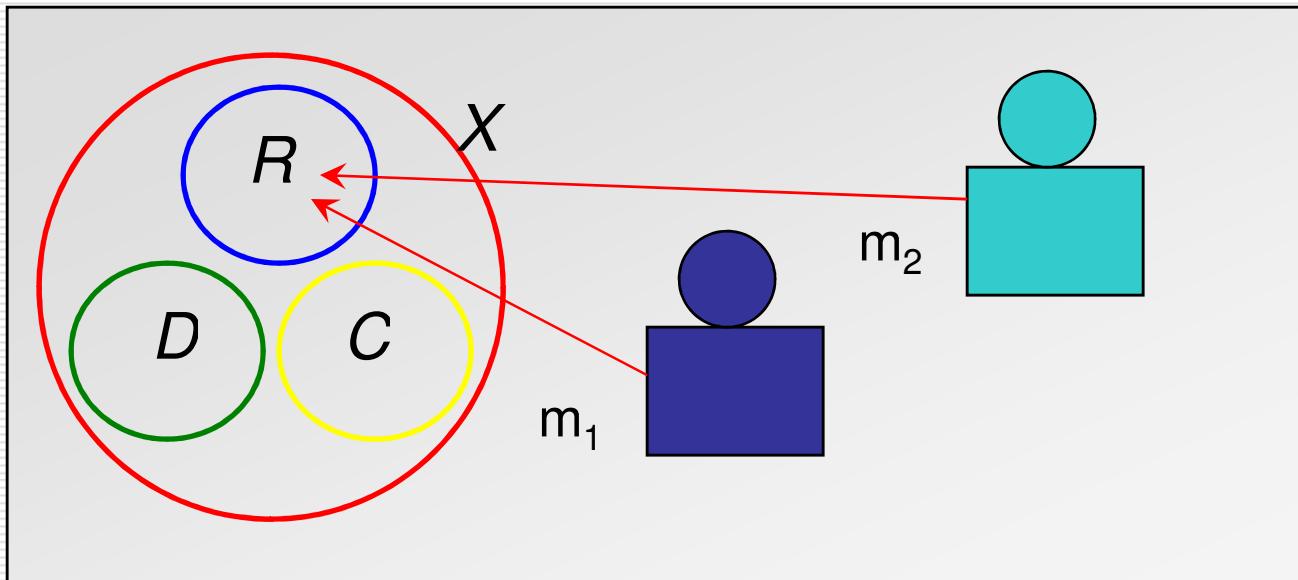


Otkrivena je neka slika za koju se smatra da je Rafaelova. Imamo tri mogućnosti:

1. Autor slike je Rafael. (R)
2. Autor je neki Rafaelov učenik. (D)
3. Slika je krivotvorina. (C)

# Primjer

---



- $R$ ,  $D$  i  $C$  su podskupovi našeg univerzalnog skupa  $X$ .
  - Dva eksperta proučavaju slike i daju svoje temeljne dodjele  $m_1$  i  $m_2$ , o pojedinim skupovima.
-

## Primjer

---

<b>Fokalni element</b>	$m_1$	$Bel_1$	$m_2$	$Bel_2$	$m_{1,2}$	$Bel_{1,2}$
$R$	0,05	0,05	0,15	0,15	0,21	0,21
$D$	0	0	0	0	0,01	0,01
$C$	0,05	0,05	0,05	0,05	0,09	0,09
$R \cup D$	0,15	0,2	0,05	0,2	0,12	0,34
$R \cup C$	0,1	0,2	0,2	0,4	0,2	0,5
$C \cup D$	0,05	0,1	0,05	0,1	0,06	0,16
$R \cup C \cup D$	0,6	1	0,5	1	0,31	1

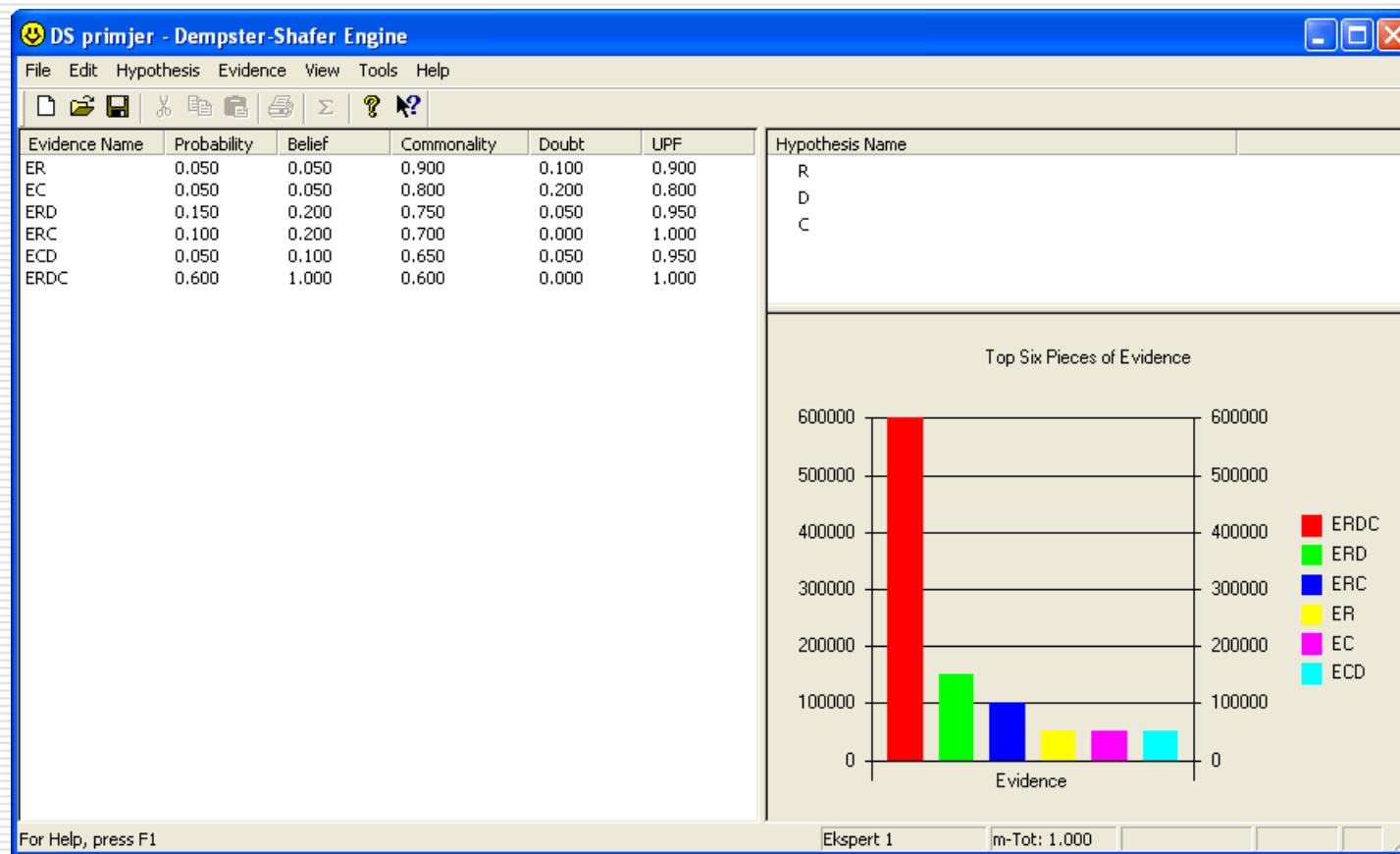
---

# Alat Dempster-Shafer Engine

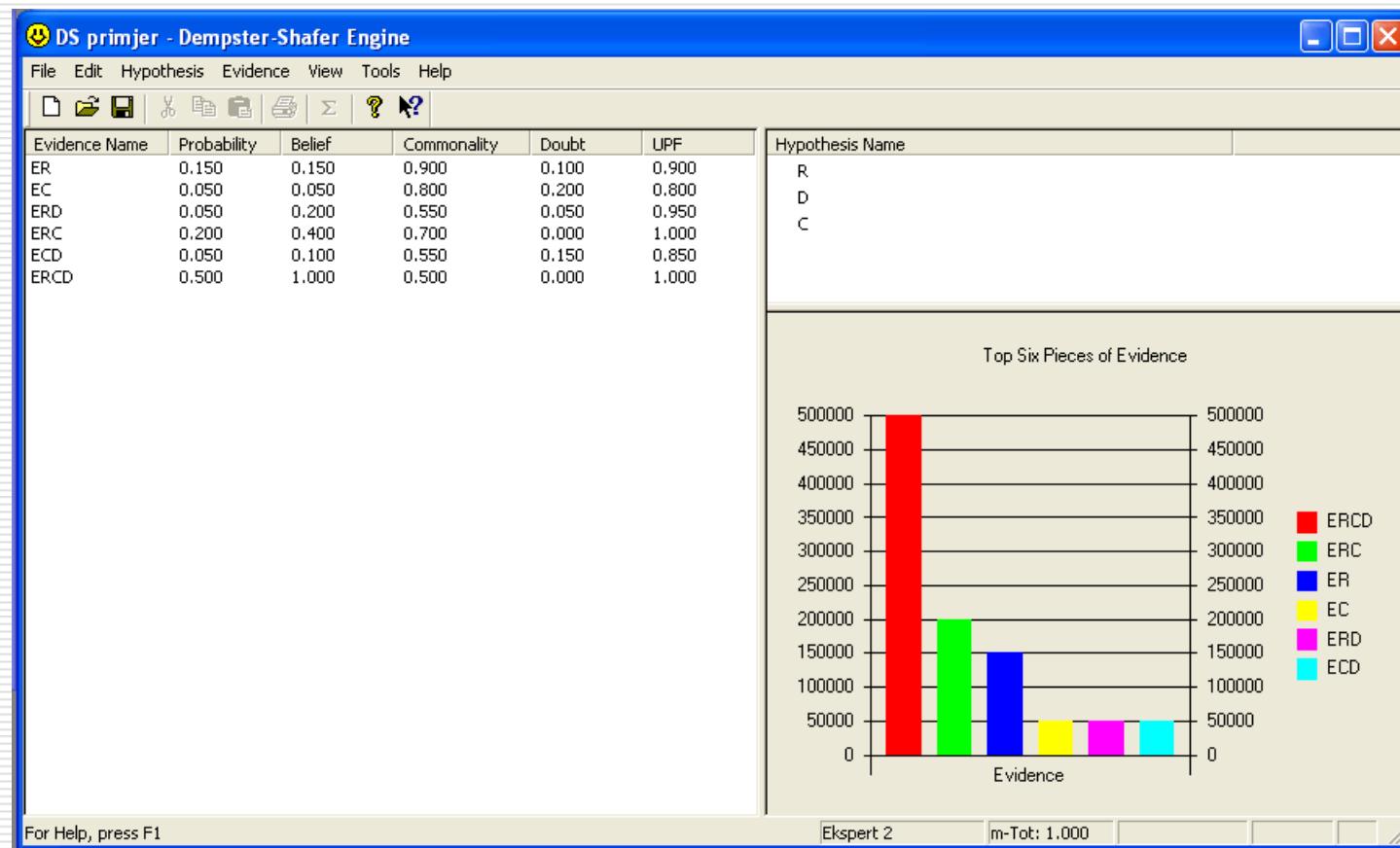
---

- Program koji omogućuje rad s Dempster-Shaferovom teorijom
  - Na slijedećim slikama je ilustrirana primjena na prošlom primjeru s dokazivanjem autentičnosti Rafaelove slike
  - Primijetite da su tri hipoteze: R, D i C
  - Dokazi (*evidences*): ER, ED, EC, itd.
-

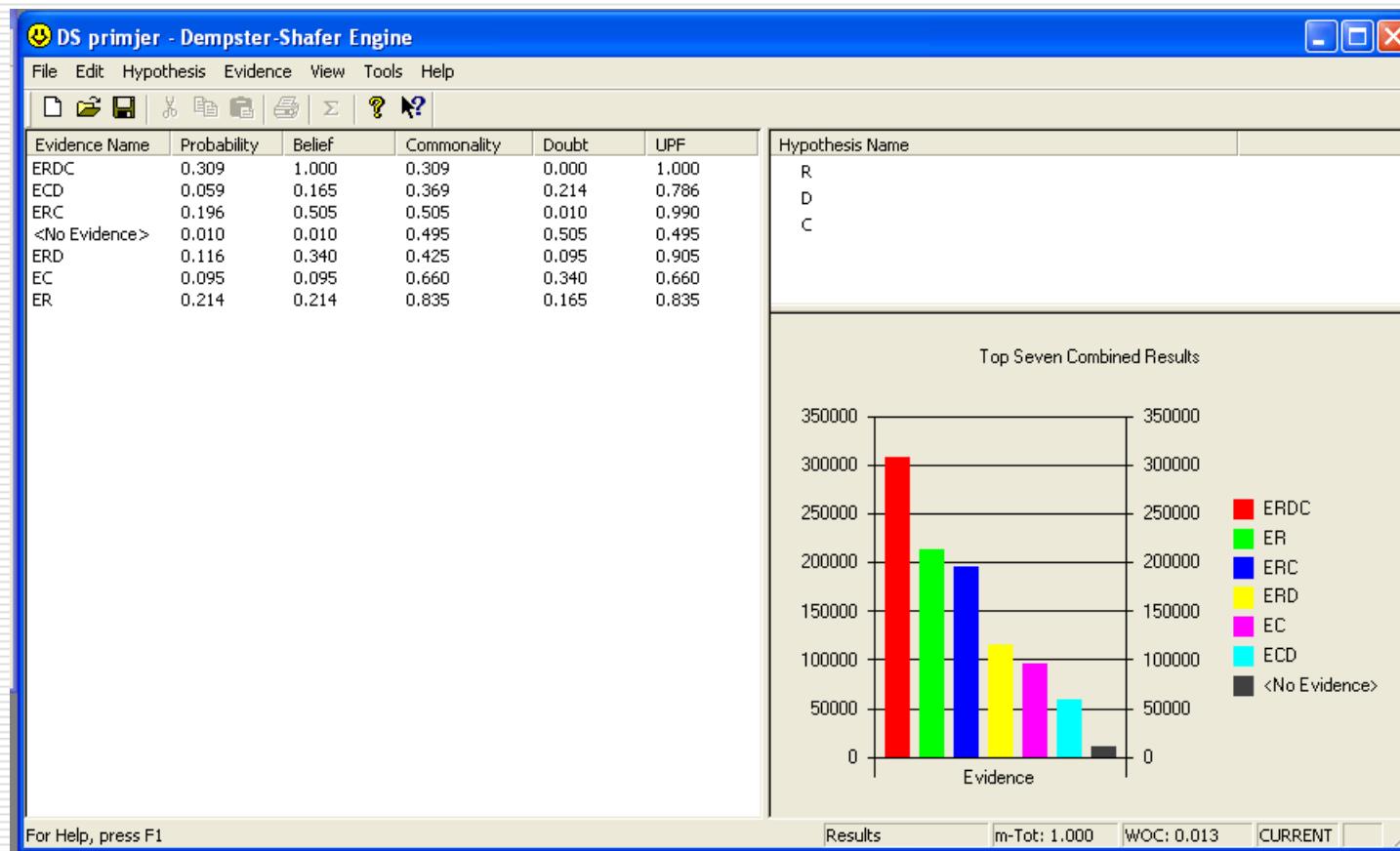
# Primjer. Alat Dempster Shafer Engine



# Primjer. Alat Dempster Shafer Engine



# Primjer. Alat Dempster Shafer Engine



# Neizraziti skupovi i neizrazito zaključivanje (repetitorij)

---

Dvije vrste skupova:

- izraziti skupovi (*crisp set*) i izrazita logika (*crisp logic*)
  - neizraziti skupovi (*fuzzy set*) i neizrazita logika (*fuzzy logic*)
  - Izražavanje općeg znanja korištenjem lingvističkih varijabli, neizrazita pravila i za njih definirane funkcije članstva, primjena neizrazitog zaključivanja.
  - Lotfi Zadeh 1965. - neizraziti skupovi i neizrazita logika.
-

# Neizraziti skupovi i neizrazito zaključivanje

---

Dvije vrste skupova:

- Izraziti skupovi (*crisp set*) i izrazita logika (*crisp logic*)
  - Neizraziti skupovi (*fuzzy set*) i neizrazita logika (*fuzzy logic*)
  - Izražavanje općeg znanja korištenjem lingvističkih varijabli, neizrazita pravila i za njih definirane funkcije članstva, primjena neizrazitog zaključivanja.
  - Lotfi Zadeh 1965. - neizraziti skupovi i neizrazita logika
-

## Izraziti skupovi

---

Izraziti (obični) skupovi:

Neka je  $U$  neki univerzalni konačni skup. Neka je  $A$  podskup tog univerzalnog skupa  $A \subseteq U$ .

Podskup  $A$  možemo odrediti preko karakteristične funkcije  $\mu(A)$ :

- $\mu_A(x) = 1$ , ako je  $x \in A$
  - $\mu_A(x) = 0$ , ako je  $x \notin A$
-

# Definicija neizrazitih skupova

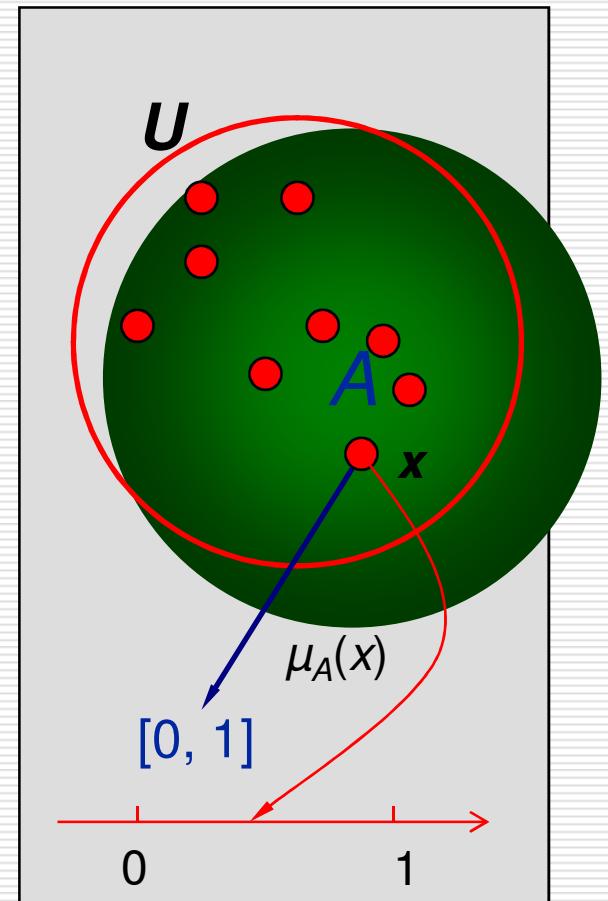
Definicija:

Neizraziti skup  $A$  univerzalnog skupa  $U$  definiran je **funkcijom članstva**

$$\mu_A: U \rightarrow [0, 1]$$

koji pridružuje svakom elementu od  $U$  broj  $\mu_A(x)$  u intervalu  $[0, 1]$ .

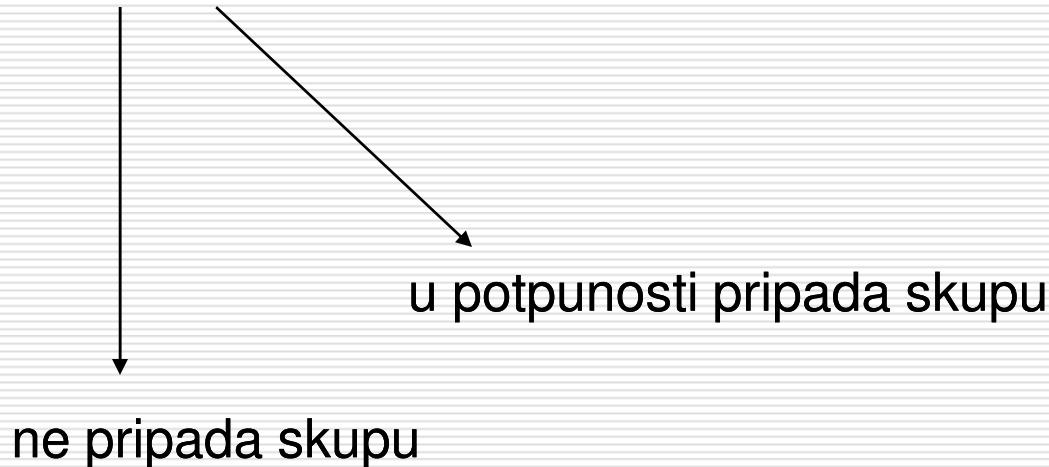
$\mu_A(x)$  je **stupanj članstva** elementa  $x$  u neizrazitom skupu  $A$ .



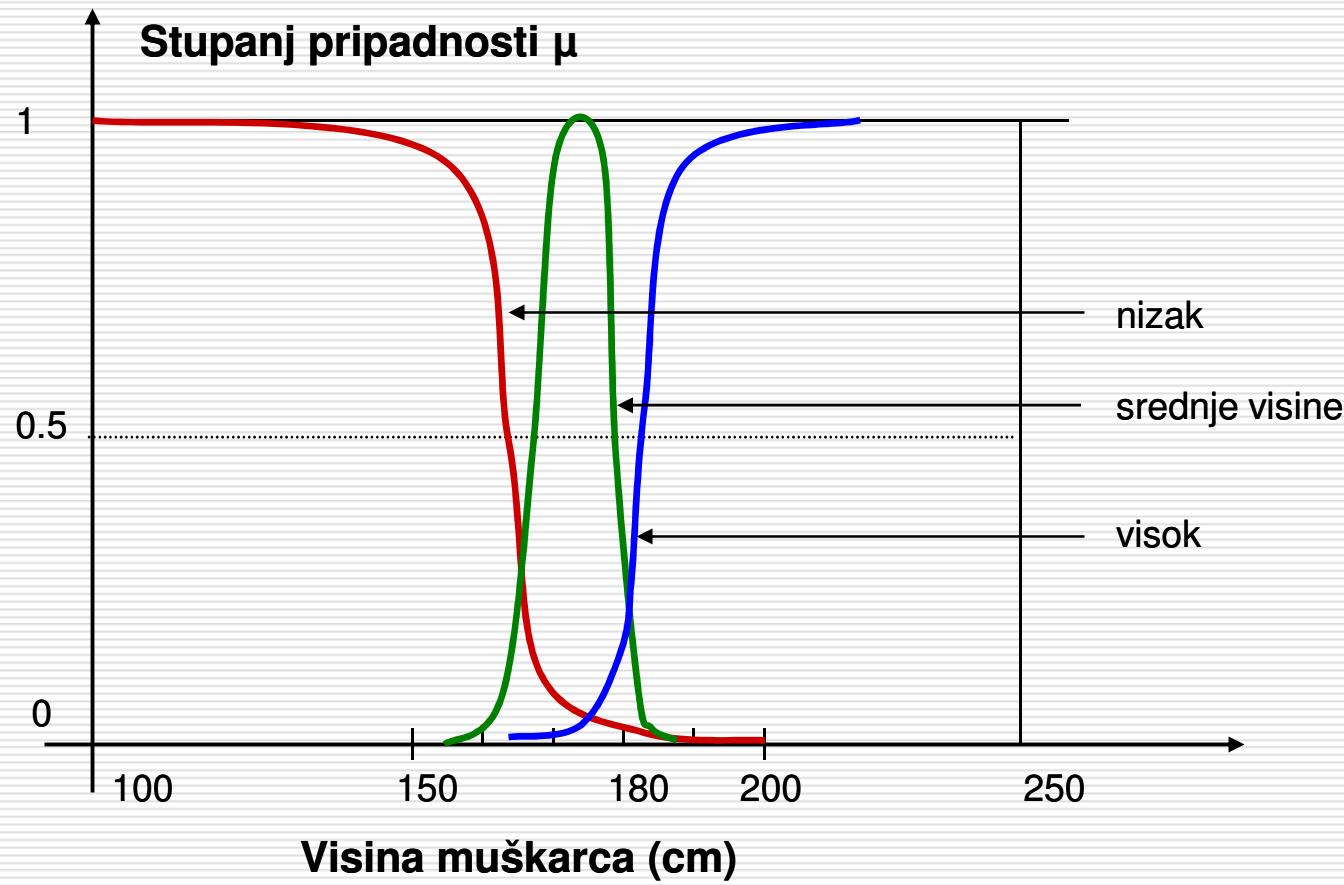
# Značenje stupnja pripadnosti

---

- [0, 1]



# Primjer neizrazitog skupa



## Zapisivanje neizrazitog skupa

---

□  $A = \mu(u_1)/u_1 + \mu(u_2)/u_2 + \dots + \mu(u_n)/u_n = \sum \mu_i/u_i$

Oblik pisanja je prikladan za diskretni univerzalni skup.

/ je znak odjeljivanja

+ je umjesto znaka unije

---

## Zapisivanje neizrazitog skupa

---

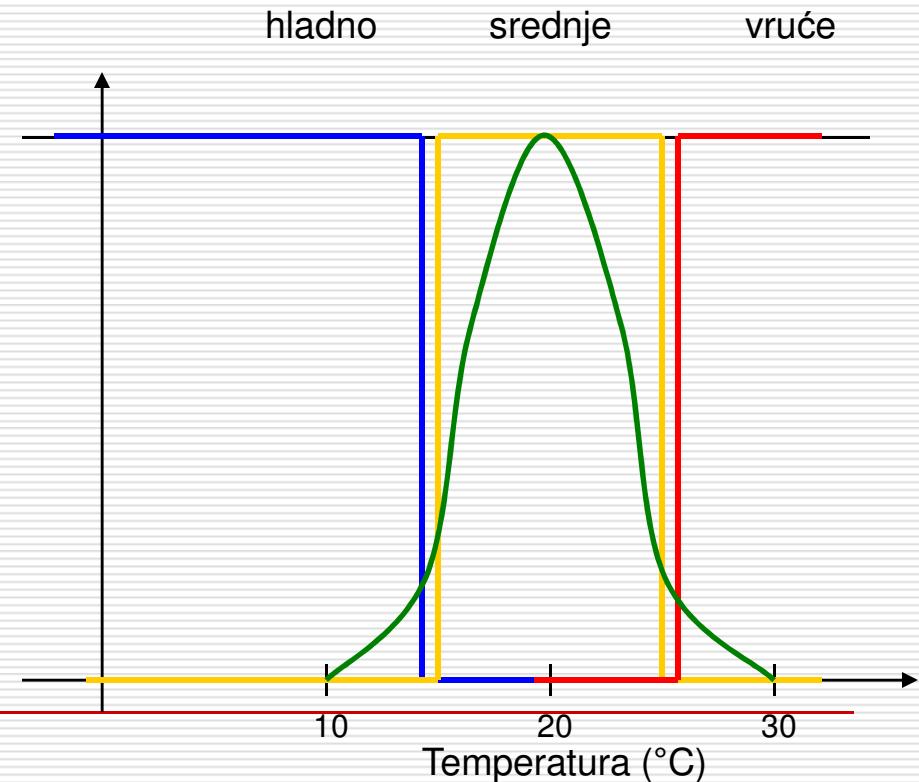
- Primjer kod GURU-a. Neizrazita (fuzzy) varijabla  $A$  predstavlja neizraziti skup.

SAVJET = {"audi" cf 90, "opel" cf 60, "mazda" cf 55}

---

# Razlika u grafičkom prikazu

- Venneov dijagram nije prikladan za prikaz neizrazitih skupova
- Za prikaz su prikladniji grafovi, naročito kad je domena kontinuirani skup npr. realni brojevi.

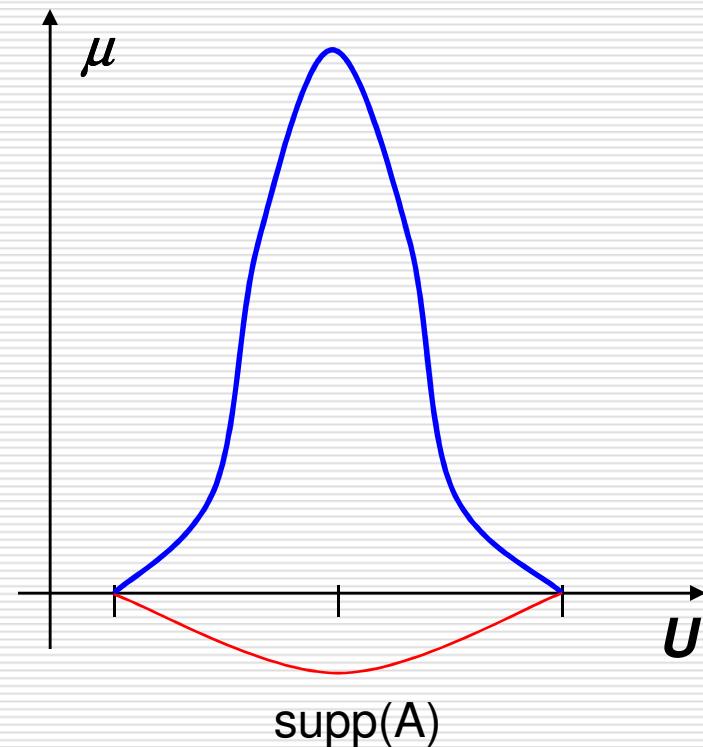


# Potpore (*support*) skupa

---

- *Support* - potpora, podskup od  $U$ , za kojeg svaki element ima stupanj članstva u  $A$  veći od 0

$$\text{supp}(A) = \{u \mid u \in U, m_a(u) > 0\}$$



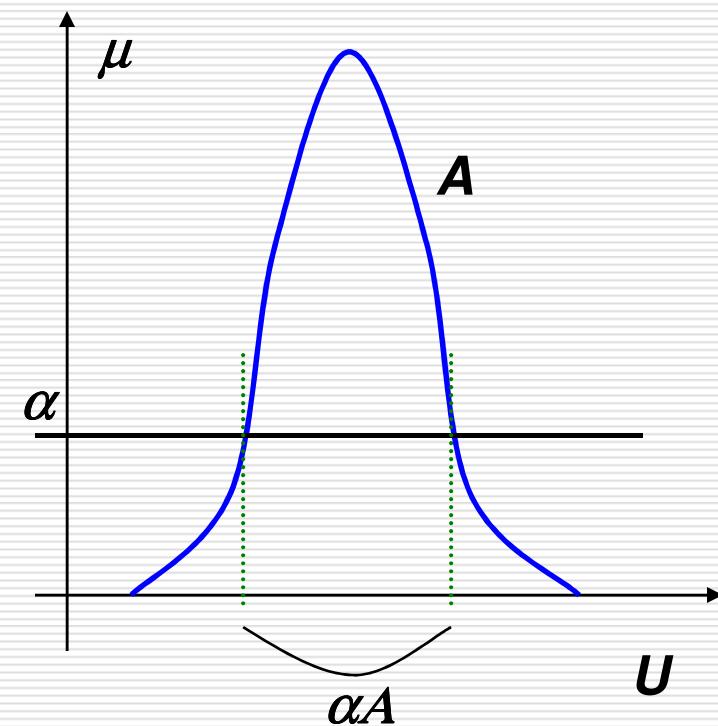
## Definicije

---

- Kardinalni broj izrazitog skupa je broj elemenata u njemu
  - Kardinalnost** neizrazitog skupa je definirana kao:  
$$M(A) = \sum \mu_A(u), u \in U$$
  - Partititivni skup je skup svih podskupova nekog skupa.
  - Skup  $A$  se naziva **normalni neizraziti skup** ako njegova funkcija članstva ima stupanj 1 za najmanje jednu vrijednost iz univerzalnog skupa  $U$ .
-

# Definicije

- Jaki (slabi) neizraziti skup se sastoji od vrijednosti koje pripadaju neizrazitom skupu  $A$  sa stupnjem članstva većim (jaki rez), ili većim ili jednakim (slabi rez), od vrijednosti  $\alpha \in [0, 1]$



# Opreacije sa neizrazitim skupovima

---

- Unija
  - Presjek
  - Jednakost
  - Komplement skupa
  - Koncentracija
  - Dilatacija
  - Podskup
  - Algebarski produkt
  - Ograničena suma
  - Ograničena razlika
  - Ograničeni produkt
  - Normalizacija
  - Algebarska suma
-

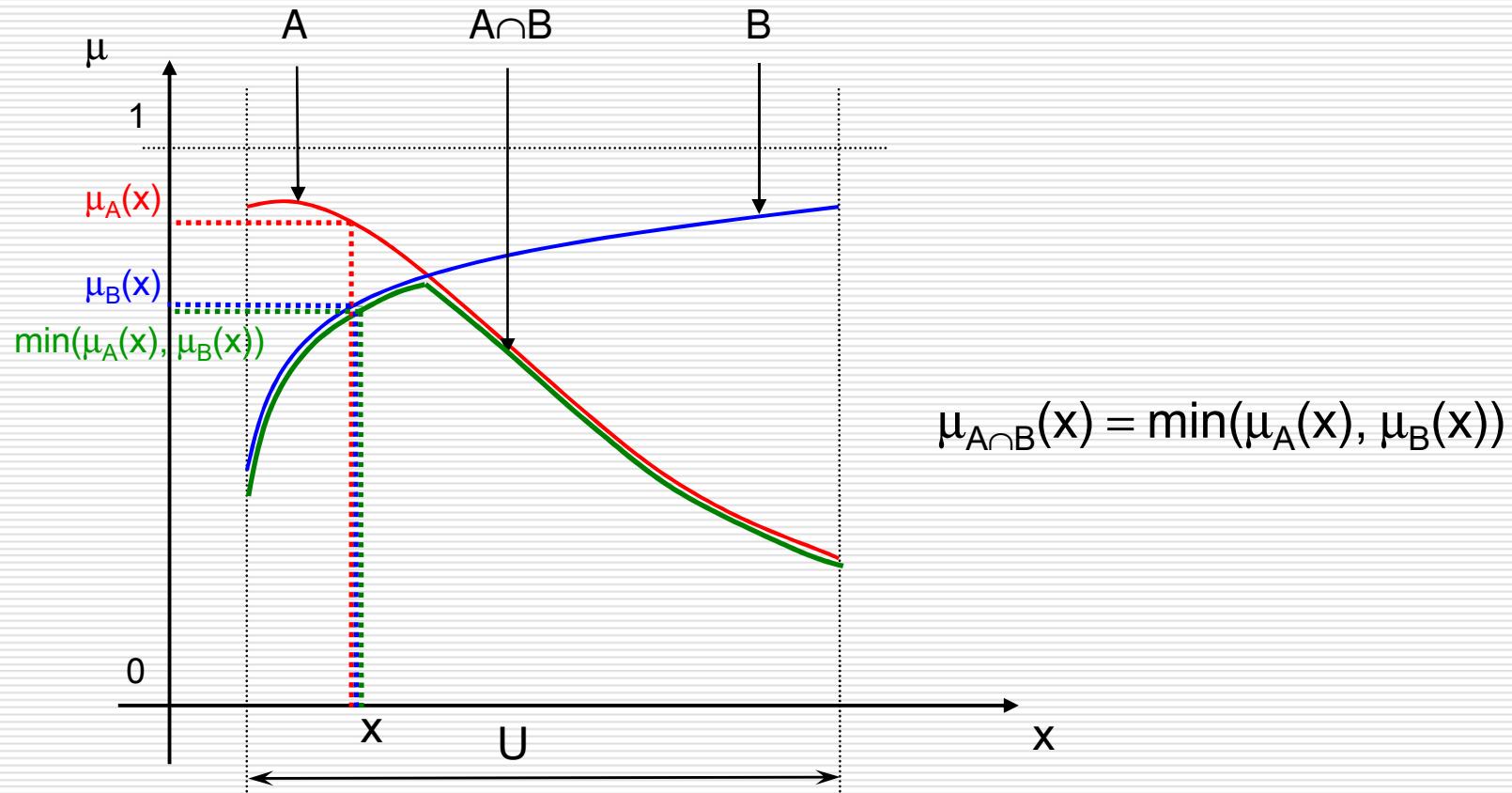
# Opreacije sa neizrazitim skupovima

---

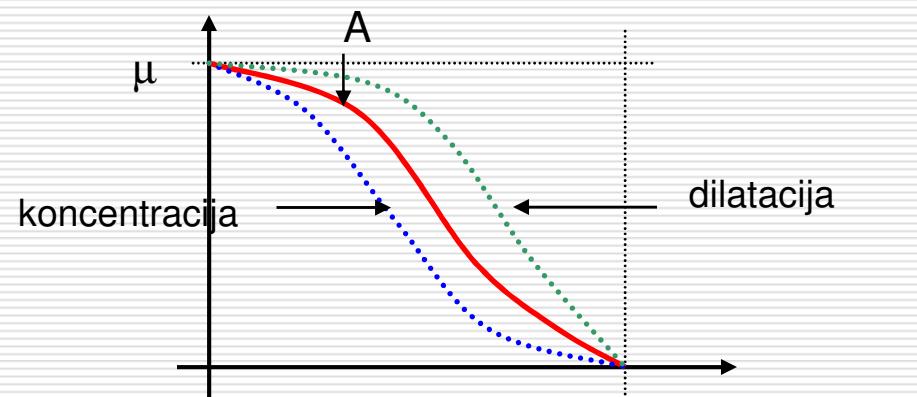
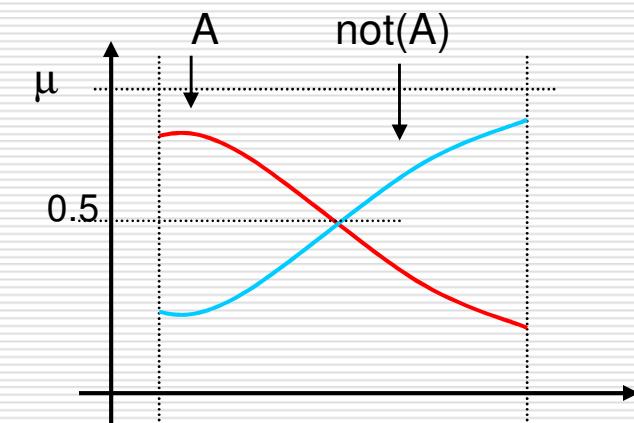
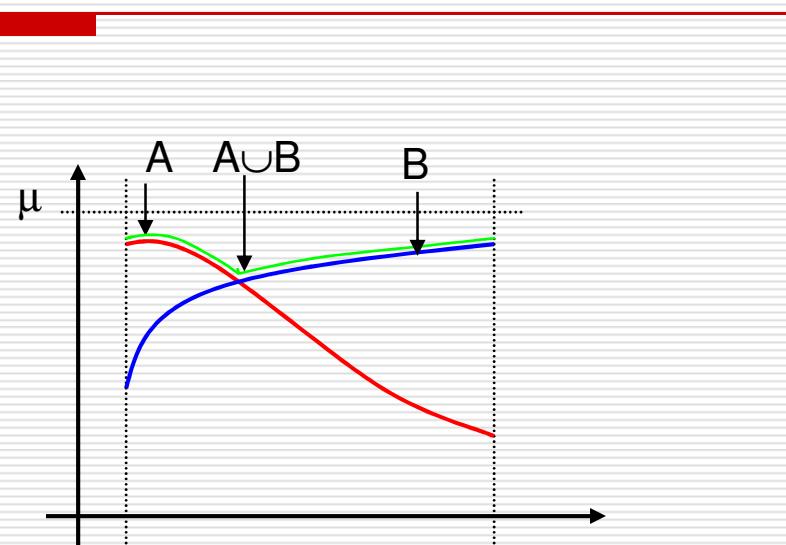
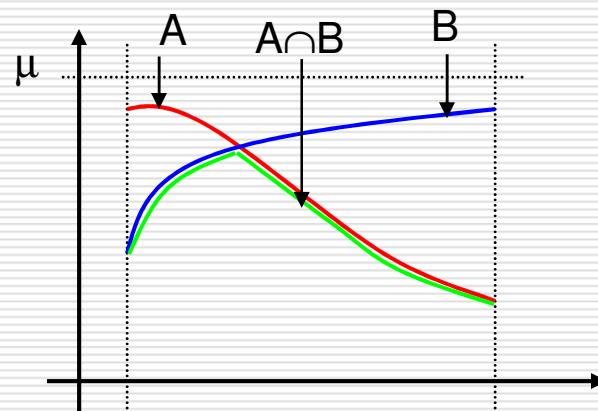
Za neizrazite podskupove  $A \subseteq U$  i  $B \subseteq U$  vrijedi:

- $A$  je podskup od  $B$  onda kada za  $\forall x \in U$  vrijedi da je  $\mu_A(x) \leq \mu_B(x)$
  - $A$  je jednak  $B$  onda kada za  $\forall x \in U$  vrijedi da je  $\mu_A(x) = \mu_B(x)$
  - $B$  je komplement od  $A$  onda kada  $\forall x \in U$  vrijedi da je  $\mu_B(x) = 1 - \mu_A(x)$ . Ako komplement označimo simbolom  $\sim$ , pisat ćeemo  $B = \sim A$ .
  - Presjek  $A \cap B$  dobiva se tako da se za  $\forall x \in U$  odredi vrijednost funkcije pripadnosti presjeku na slijedeći način:
    - $\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$
    - Unija  $A \cup B$  dobiva se tako da se za  $\forall x \in U$  odredi vrijednost funkcije pripadnosti uniji na slijedeći način:
      - $\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$
-

# Presjek neizrazitih skupova



# Ilustracije nekih operacija s neizrazitim skupovima



# Svojstva operacija s neizrazitim skupovima

---

- Asocijativnost
  - Komutativnost
  - Distributivnost
-

# Neizraziti skupovi i zaključivanje

---

- Neizraziti podskupovi i lingvističke varijable
  - Neizrazito zaključivanje
  - Ekspertni sustavi s neizrazitim zaključivanjem
  - Dizajn neizrazitih sustava
  - Primjena neizrazite logike (skupova, sustava)
-

# Konceptualizacija s neizrazitim terminima

---

- Kako prikazati problem pomoću neizrazitih termina
  - Konceptualizacija u neizrazitim terminima
  - Lingvističke varijable ili termini:  
visok, viši, jako visok, vrlo jako, spor, loš
-

# Lingvistička varijabla

---

- Lingvistička varijabla označuje varijablu koja poprima neizrazite vrijednosti i ima lingvističko značenje.
  - Primjer: "brzina" automobila je lingvistička varijabla i poprima vrijednosti: "mala", "srednja", "visoka" Lingvističke varijable mogu biti kvalitativne i kvantitativne.
-

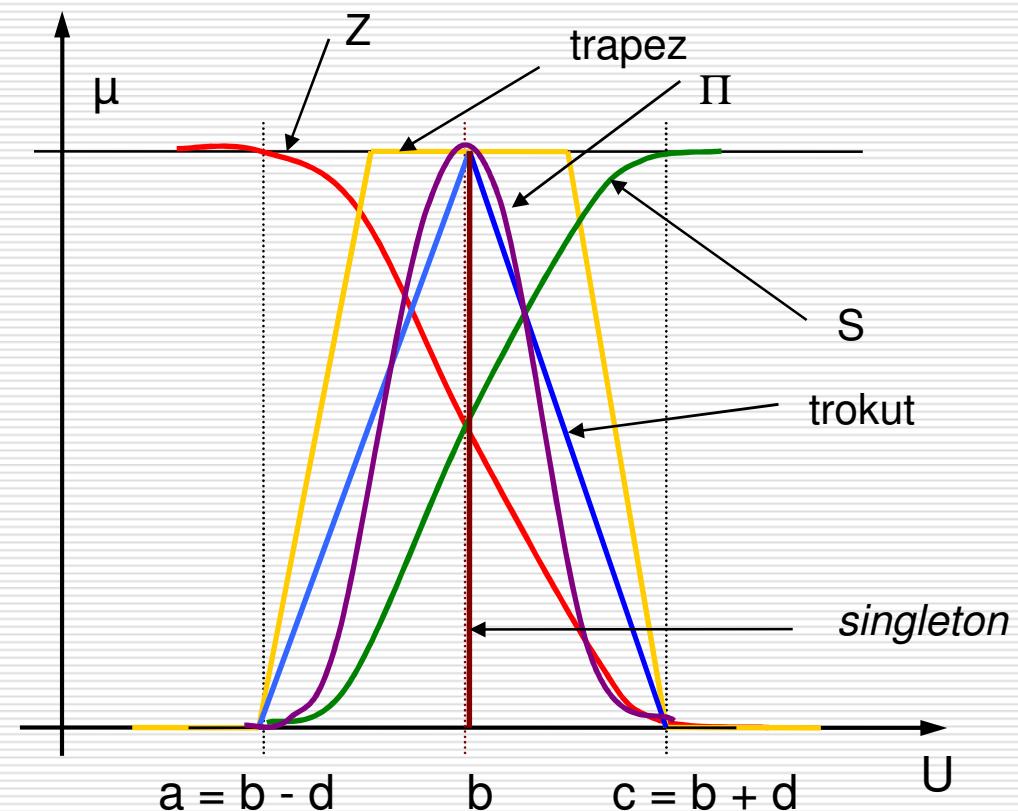
## Neizrazita kvantizacija

---

- Neizrazita kvantizacija je proces prikaza lingvističke varijable u skupu lingvističkih vrijednosti
  - Kvantizacija se može često prikazati preko standardnih funkcija.
-

# Standardni tipovi funkcija pripadnosti

- Jednostruka vrijednost (singleton)
- Trokut
- Trapez
- S funkcija
- Z funkcija
- $\Pi$  funkcija (zvonolika)



# Neizrazite relacije i neizrazite implikacije

---

- Primjeri:
    - Ocjene na drugoj i trećoj godini su slične
    - Što više jedeš masnu hranu, veća je opasnost od infarkta
  - Neizrazite relacije temelj su za neizrazite implikacije, koja su temelj za zaključivanje kod neizrazitih skupova.
-

## Svojstva neizrazitih relacija

---

- Kompozicija relacija
  - Svojstva neizrazitih relacija
-

# Neizraziti skupovi i neizrazito zaključivanje

---

Dvije vrste skupova:

- izraziti skupovi (*crisp set*) i izrazita logika (*crisp logic*)
  - neizraziti skupovi (*fuzzy set*) i neizrazita logika (*fuzzy logic*)
-

## Neizrazite relacije - primjer

---

- Prikazujemo relaciju

“okorjeli pušač” → “visoki rizik od raka”

- Zanima nas relacija

(“umjereni pušač”, (“okorjeli pušač” → “visoki rizik od raka”))

---

## Primjer

---

"okorjeli pušač"

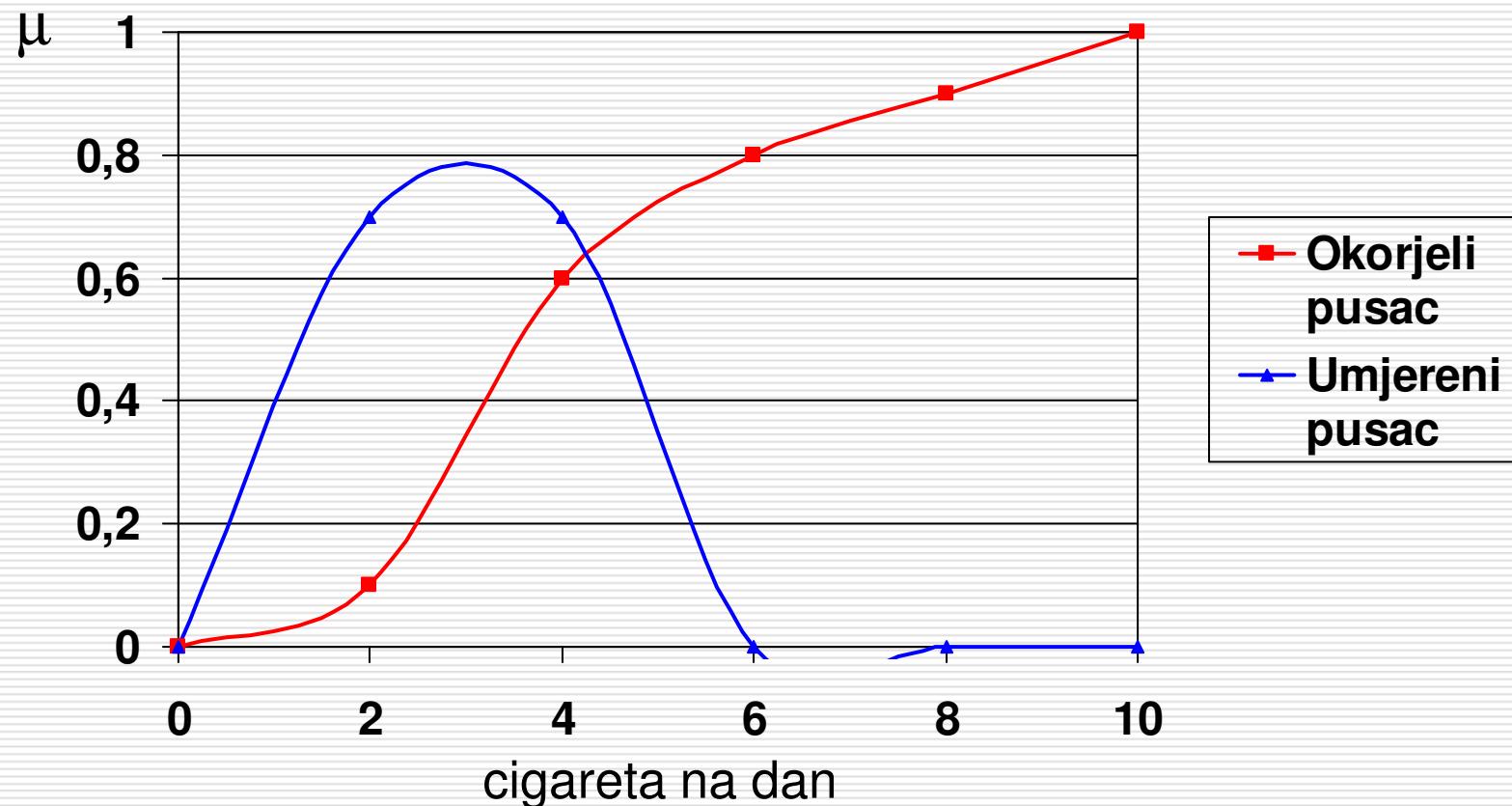
Broj cigareta na dan	0	2	4	6	8	10
Stupanj pripadnosti $\mu$	0	0.1	0.6	0.8	0.9	1.0

"umjereni pušač"

Broj cigareta na dan	0	2	4	6	8	10
Stupanj pripadnosti $\mu$	0	0.7	0.7	0	0	0

---

## Primjer. Grafički prikaz relacija za okorjelog i umjerenog pušača



## Primjer relacije visok rizik od raka

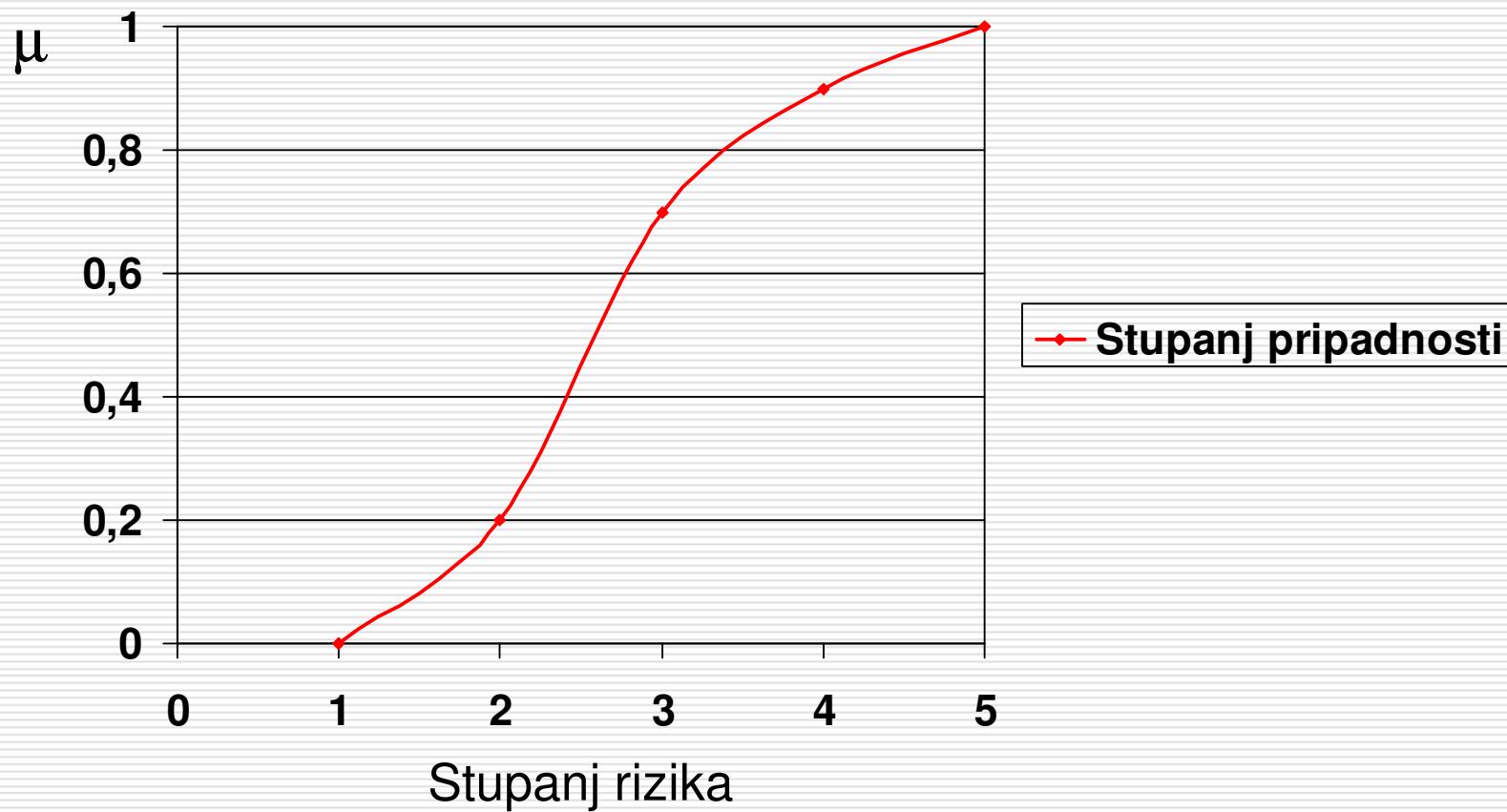
---

- Visok rizik od raka

Stupanj rizika	1	2	3	4	5
Stupanj pripadnosti $\mu$	0	0.2	0.7	0.9	1.0

---

## Primjer. Grafički prikaz



## Primjer. Relacija implikacije

---

- Prikazujemo implikaciju: Ako je okorjeli pušač, tada visok rizik od raka.
- okorjeli pušač → visok rizik od raka

		Rizik				
		1	2	3	4	5
c i g a r e t a	0	0.0	0	0	0	0
	2	0.0	0.1	0.1	0.1	0.1
	4	0.0	0.2	0.6	0.6	0.6
	6	0.0	0.2	0.7	0.8	0.8
	10	0.0	0.2	0.7	0.9	1.0

→  $\mu$

Broj cigareta na dan	0	2	4	6	8	10
Stupanj pripadnosti	0	0.1	0.6	0.8	0.9	1.0

Stupanj rizika	1	2	3	4	5
Stupanj pripadnosti	0	0.2	0.7	0.9	1.0

c i g a r e t a	Rizik				
	1	2	3	4	5
	0.0	0	0	0	0
	0.0	0.1	0.1	0.1	0.1
	0.0	0.2	0.6	0.6	0.6
	0.0	0.2	0.7	0.8	0.8
	0.0	0.2	0.7	0.9	1.0

$\min(0.6, 0.7)$

umjereni pušač

	1
0	0.0
2	0.7
4	0.7
6	0.0
10	0.0

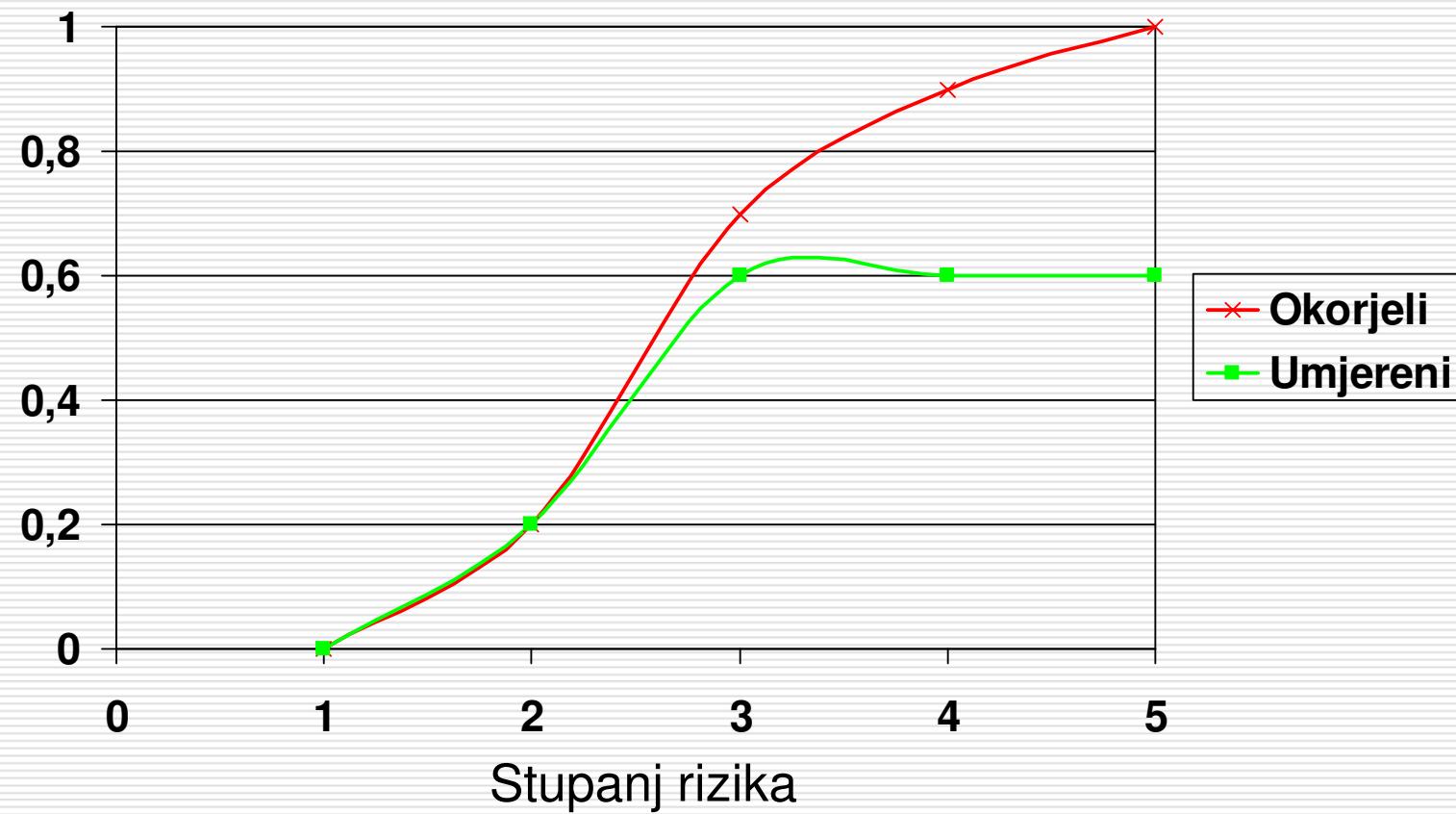
okorjeli pušač → visok rizik od raka

		Rizik				
		1	2	3	4	5
Cigaretta	0	0.0	0	0	0	0
	2	0.0	0.1	0.1	0.1	0.1
	4	0.0	0.2	0.6	0.6	0.6
	6	0.0	0.2	0.7	0.8	0.8
	10	0.0	0.2	0.7	0.9	1.0

		Rizik				
		1	2	3	4	5
Cigaretta	0	0	0	0	0	0
	2	0	0.1	0.1	0.1	0.1
	4	0	0.2	0.6	0.6	0.6
	6	0	0	0	0	0
	10	0	0	0	0	0
	Max	0	0.2	0.6	0.6	0.6

min (0.7, 0.1)

## Primjer. Grafički prikaz implikacija



# Neizrazita logika

---

- Viševrijednosna logika (npr. Lukasiewicz, 1930. - trovrijednosna logika)
  - $T = \{0, 0.5, 1\}$  (klasična l.  $T = \{0, 1\}$ )
  - Neizrazita logika
  - Vrijednost istinitosti "X je A" je dana preko funkcije pripadnosti  $\mu_A$
  - Primjer: osoba je visoka.
  - Operacije u *fuzzy* logici definirane su preko operacija s *fuzzy* skupovima (min, max, komplement).
-

# Operacije u neizrazitoj logici

---

- Modifikatori (koncentracija, dilatacija)
  - Modus ponens
  - Modus tolens
  - Silogizam
  - De Morganov zakon je modificiran
  - Neizrazita logika operira s neizrazitim propozicijama, vezama i pravilima zaključivanja.
-

# Pravila, zaključivanje, fuzifikacija i defuzifikacija

---

Neizraziti sustav se sastoji od tri dijela

- Neizrazite ulazne i izlazne varijable i njihove vrijednosti
  - Neizrazita pravila
  - Neizrazite metode zaključivanja, s fuzifikacijom i defuzifikacijom
-

# Neizrazita pravila

---

- Oblik pravila Zadeh-Mamdani

IF  $x$  is  $A$  THEN  $y$  is  $B$

$x$  is  $A$  i  $y$  is  $B$  su dvije neizrazite propozicije

$x$  i  $y$  su neizrazite varijable definirane nad skupovima  $U$  i  $V$

$A$  i  $B$  su neizraziti skupovi definirani preko njihovih funkcija članstva

$\mu_A$  i  $\mu_B$

- Oblik pravila Takagi-Sugeno

IF  $x$  is  $A$  and  $y$  is  $B$  THEN  $z$  is  $f(x, y)$

za aproksimaciju funkcija.

- Primjer

IF  $x$  is  $A$  and  $y$  is  $B$  then  $z = 5x - 2y + 3$

---

## Neizrazita pravila

---

- Oblik pravila Takagi-Sugeno

IF  $x$  is A and  $y$  is B THEN  $z$  is  $f(x, y)$

za aproksimaciju funkcija.

- Primjer

IF  $x$  is A and  $y$  is B then  $z = 5x - 2y + 3$

---

## Zaključivanje

---

- Temelji se na podudaranju: podudaranje nove činjenice sa skupom pravila  $R_i$  ( $i = 1, 2, \dots, n$ ).
  - Neizrazito zaključivanje koristi neizrazite relacije implikacije i operatore, te operatore koji povezuju neizrazita pravila
  - Proces zaključivnja dovodi do novih činjenica
-

# Strategije zaključivanja

---

- Moguć je velik broj strategija zaključivanja
- Npr. *modus ponens*
  - IF x je A THEN y je B
  - IF x je A' THEN y je B'

# Fuzifikacija, vrednovanje pravila i defuzifikacija

---

- Postupak kada su ulazni podaci izraziti, a također se očekuje da izlazne varijable budu izrazite.
- Metoda "fuzifikacije, vrednovanja pravila i defuzifikacije" nad pravilima tipa:

IF  $x_1$  is  $A_1$  and  $x_2$  is  $A_2$  THEN  $y$  is  $B$

---

## Definicija pojmove

---

- Fuzifikacija je proces nalaženja stupnja članstva  $\mu_{A1}(x_1)$  i  $\mu_{A2}(x_2)$  ukoliko podaci  $x_1$  i  $x_2$  pripadaju neizrazitim pravilima.
- Vrednovanje pravila - nakon fuzifikacije kod rada sa skupovima i funkcijama članstva  $\mu_{A1}(x_1)$  i  $\mu_{A2}(x_2)$  dobivamo izlaznu funkciju od  $B'$ . Npr.

$$B' = B \min\{\mu_{A1}(x_1), \mu_{A2}(x_2)\}, \text{ minimum}$$

---

## Definicija pojmove

---

- Defuzifikacija je proces računanja jednostavne numeričke vrijednosti na temelju funkcije članstva za tu varijablu
- Dvije metode
  - težište (COG)

$$y' = \frac{\sum \mu_i(x_i) \times x_i}{\sum \mu_i(x_i)}$$

- sredina od maksimuma (MOM)
-

# Primjer neizrazitog zaključivanja

---

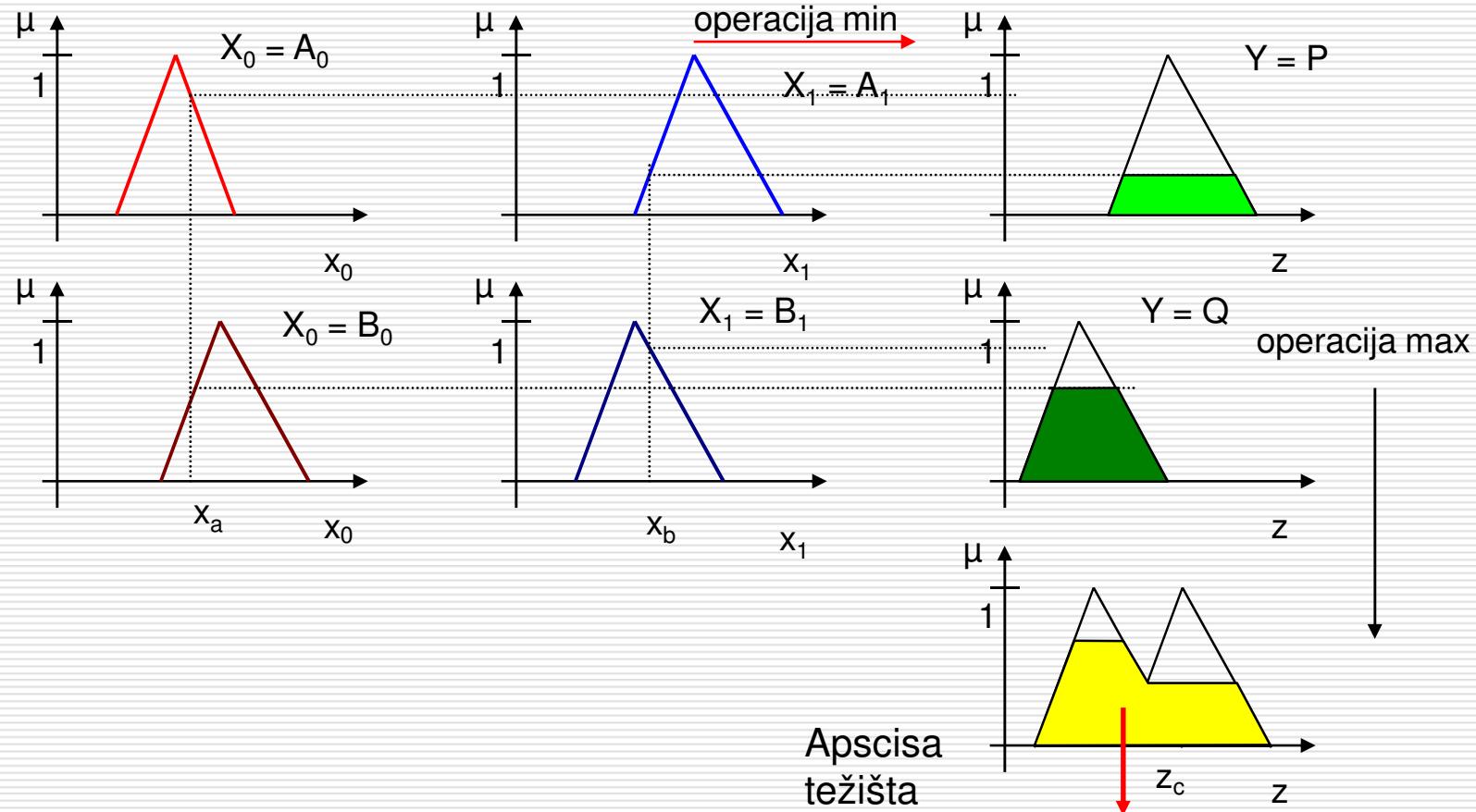
- Postoji skup pravila

**Pravilo 1:** Ako je  $((X_0 = A_0) \wedge (X_1 = A_1))$  onda je  $(Y = P)$

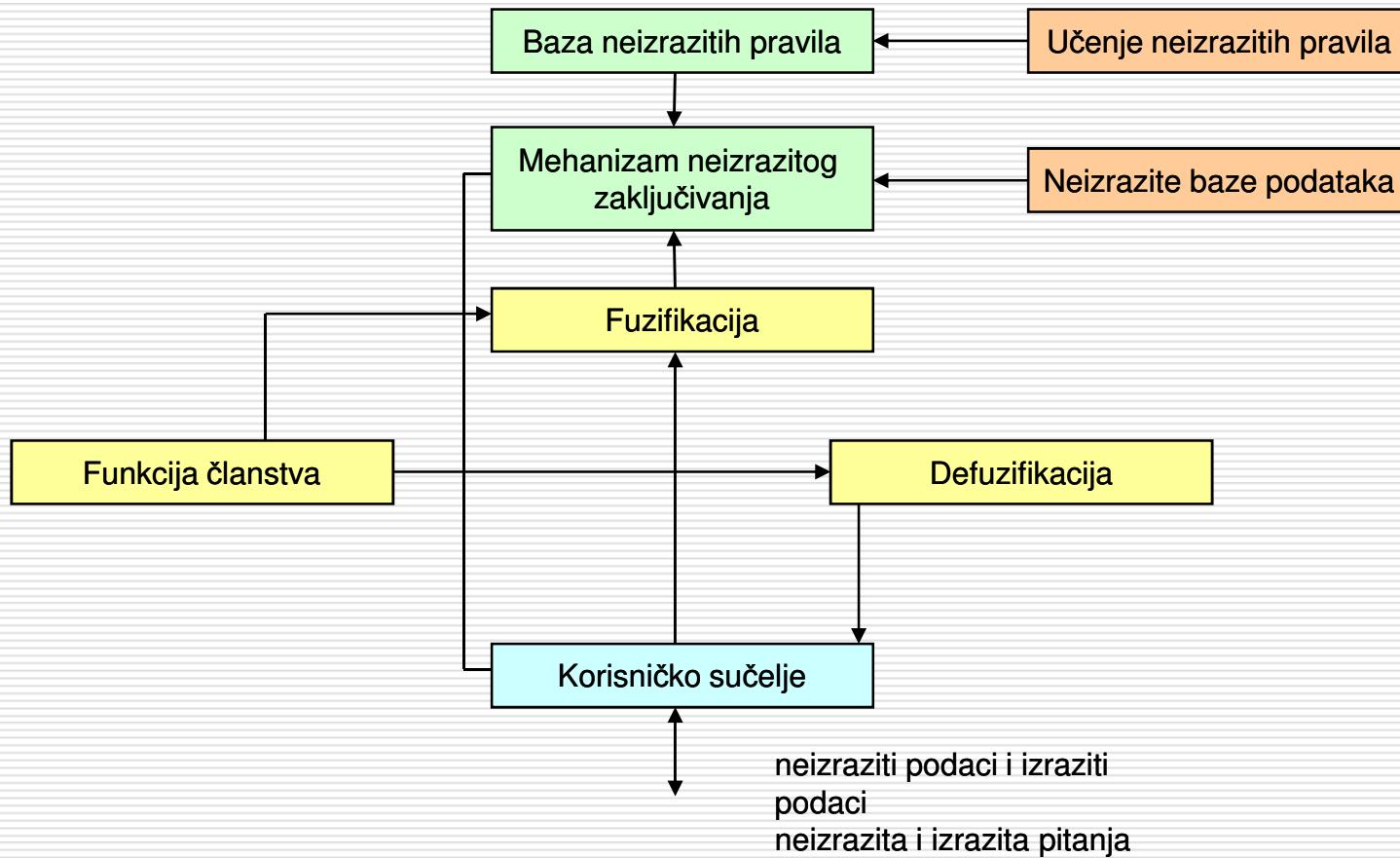
**Pravilo 2:** Ako je  $((X_0 = B_0) \wedge (X_1 = B_1))$  onda je  $(Y = Q)$

---

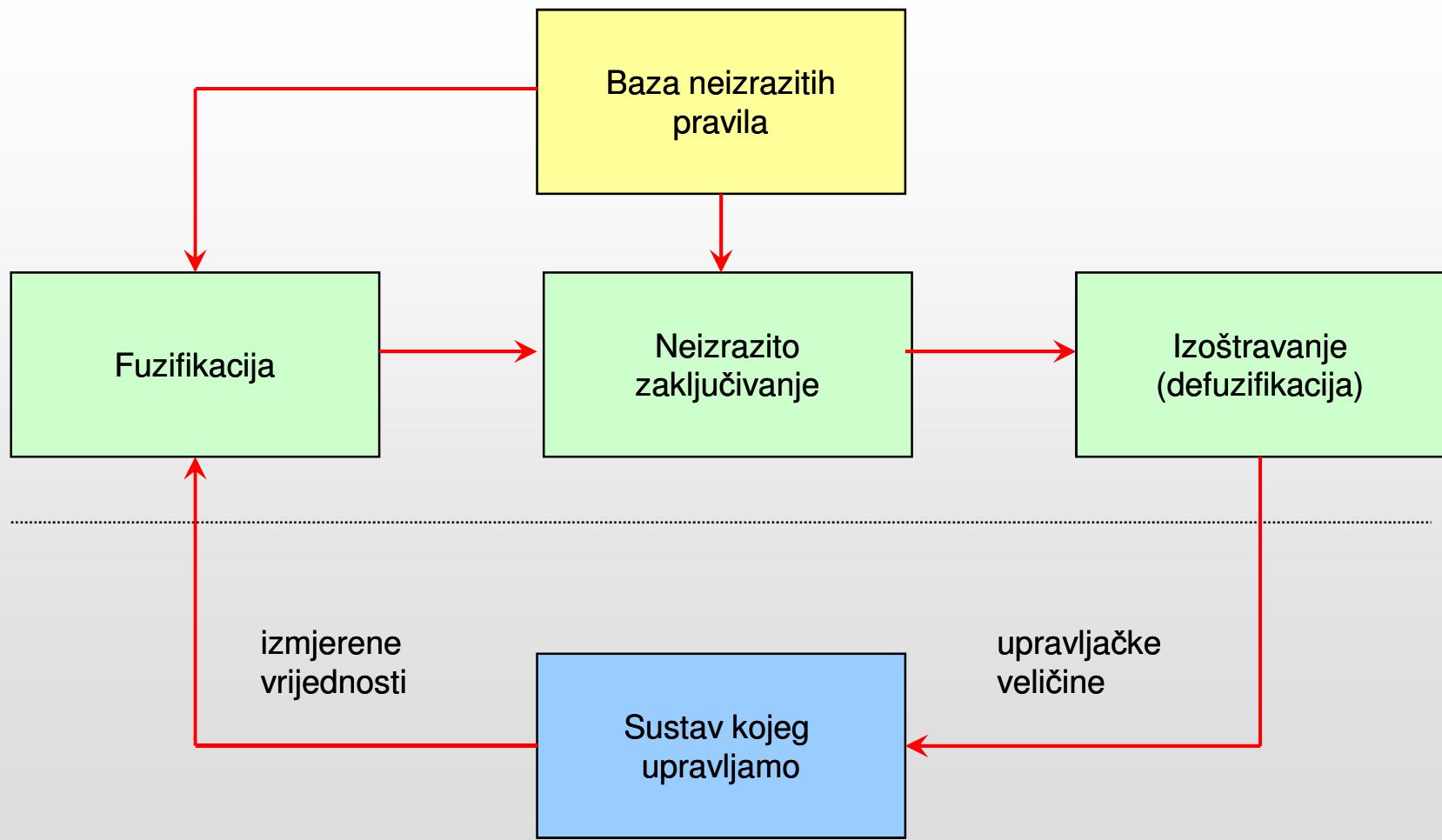
# Neizrazito zaključivanje - primjer



# Neizraziti ekspertni sustavi



# Neizrazito upravljanje neizrazitim sustavom



# Dizajn neizrazitih sustava

---

Faze dizajna neizrazitih sustava:

1. Identificiranje problema i izbor vrste neizrazitog sustava.
  2. Definiranje ulaznih i izlaznih varijabli i njihovih funkcija pripadnosti.
  3. Artikuliranje skupa heurističkih pravila.
  4. Izbor metoda neizrazitog zaključivanja.
  5. Eksperimentiranje s prototipom.
-

# Primjer dizajna fuzzy sustava

---

Faze dizajna neizrazitih sustava:

- 1. Identificiranje problema i izbor vrste neizrazitog sustava.**
    - Potrebno je izraditi jednostavan sustav koji će zainteresiranim mladim ljudima koji se žele baviti sportovima omogućiti da brzo pronađu sport u kojem bi bili uspješni. Treba napraviti za stotinjak sportova i disciplina. Problem će se riješiti preko neizrazite logike.
    - Primjer određivanje uspjeha desetobojca u disciplini bacenje kugle na temelju konstitucije.
-

# Primjer dizajna fuzzy sustava

---

Faze dizajna neizrazitih sustava:

2. Definiranje ulaznih i izlaznih varijabli, veličina i njihovih funkcija pripadnosti.

- Ulazne varijable su visina i težina, na temelju čega se određuje očekivani uspjeh u bacanju kugle, duljina u metrima. Te varijable se pretvaraju u neizrazite varijable.
  - visina (m) - rast: srednji (oko 180 cm), visok (oko 190), vrlo visok (više od 200)
  - težina (kg) - težina1: srednja (oko 80), teška (oko 90), vrlo težak (100 i više)
  - duljina hica (m) - uspjeh: osrednji (oko 16 m), dobar (17 m), izvrstan (18 m).
-

# Dizajn neizrazitih sustava

---

Faze dizajna neizrazitih sustava:

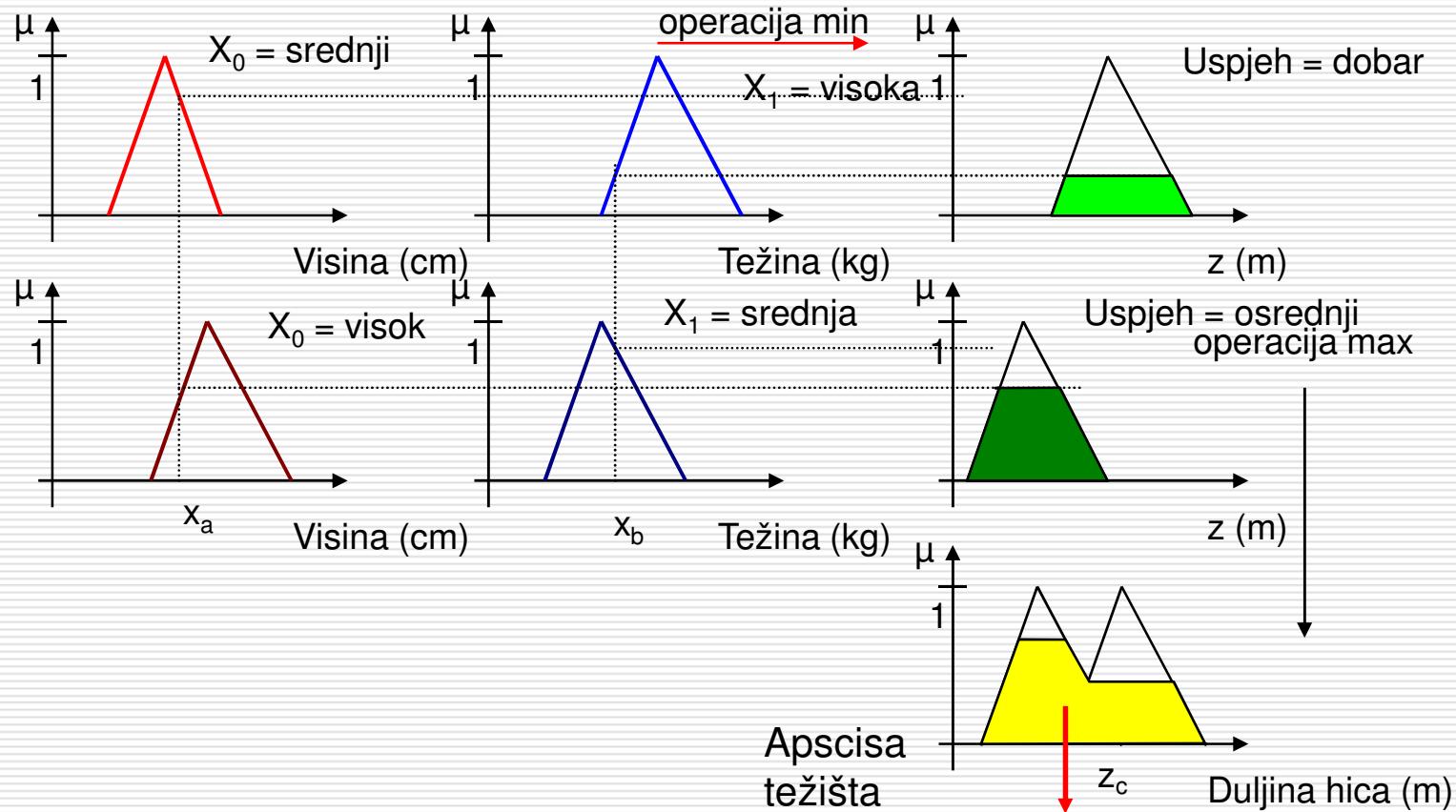
3. Artikuliranje skupa heurističkih pravila.

- Ako je rast srednji i težina teška tada uspjeh dobar.
- Ako rast visok i težina srednja tada uspjeh osrednji.

4. Izbor metoda neizrazitog zaključivanja.

Pogledati sliku.

# Primjer. Bacanje kugle



# Dizajn neizrazitih sustava

---

Faze dizajna neizrazitih sustava:

## 5. Eksperimentiranje s prototipom.

- Provjerite primjer: mladić visok 187 cm i težak 84 kg postiže duljinu hica od 16,66 m.
-

## Primjeri primjene

---

- Prepoznavanje uzorka i klasifikacija
  - Klasterifikacija
  - Prepoznavanje slike, signala i govora
  - Predviđanje
  - Upravljanje, nadzor, dijagnoza i planiranje
  - Optimizacija i odlučivanje
  - Ciljano e-poslovanje
-

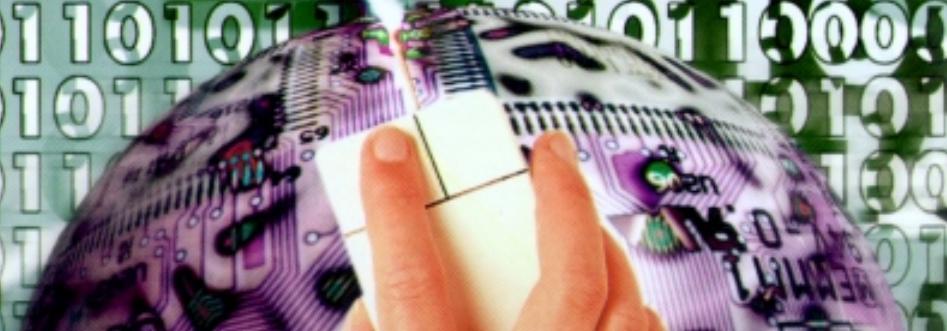
# INTELIGENTNI SUSTAVI U POSLOVNOM MODELIRANJU

---

Prof. dr. sc. Božidar Kliček  
Sveučilište u Zagrebu  
Fakultet organizacije i informatike, Varaždin



# Stabla odlučivanja i neuronske mreže



# Uvod

- 
- Prevladavanje problema prikupljanja znanja
  - Strojno učenje:
    - neuronske mreže
    - indukcija pravila (stabla odlučivanja)
  - Neuronsko računalstvo
    - alternativa Von Neumannovim računalima
  - Omogućuje uspješno rješavanje nekih problema obrade informacija
-

# Osnovne karakteristike

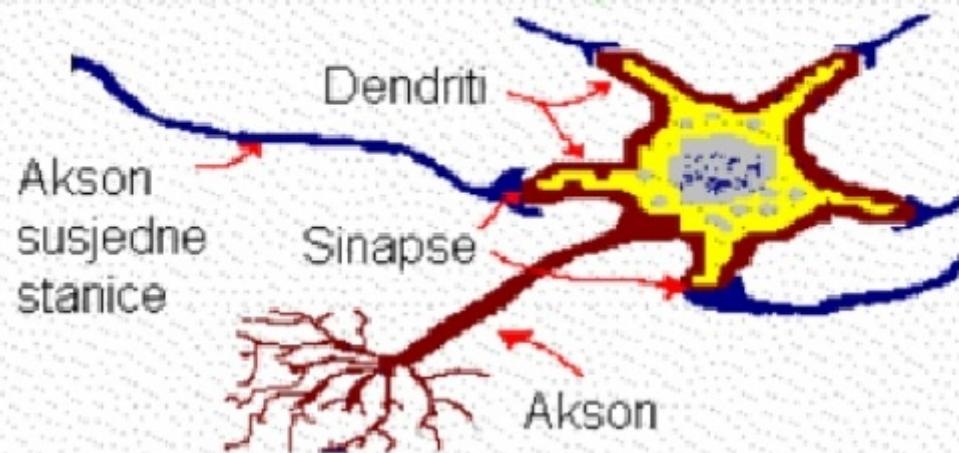
- Razlika NM (neuronski mreža) od ostalih informacijskih tehnologija:
  - NM su adaptivne (sposobnost učenja)
  - unapređuju se s iskustvom
  - NM su prirodno masivno paralelne
- Zanimanje za NM nastalo je zbog:
  - velike snage obrade pojedinih vrsta informacija čak i jednostavnijih bića
  - težna za razumijevanjem bioloških sustava računalnim modeliranjem mozga

# Teorijski temelji

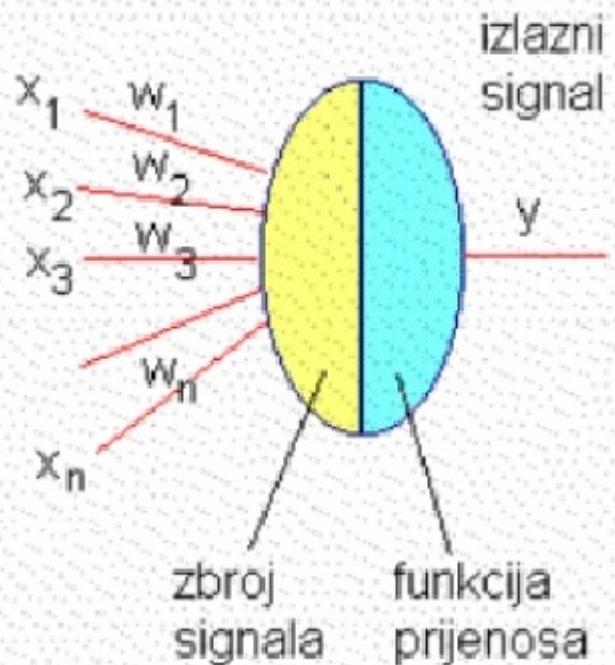
- Analogija s mozgom
- Neuronske mreže su računalne strukture modelirane prema biološkim procesima
- Biološki neuron je inspirirao razvoj umjetnih, računalnih neuronskih mreža

# Biološki i računalni neuron

Biološki neuron



Računalni neuron



# Neuron

- 
- Neuron - stanica u organizmu specijalizirana za obradu informacija
  - Elementi:
    - soma (jezgra)
    - akson (pridružen somi), električki aktivan, stvara pulseve
    - dendriti, dobivaju ulaze od ostalih neurona
    - sinapse (specijalizirani kontakti)
  - Veličina neurona: 100 µm u promjeru
-

# Neuron

- 
- Ljudski mozak - približno  $10^{11}$  neurona
  - Neuron - 1000 do 100,000 sinapsi
  - Ljudski organizam -  $10^{14}$  do  $10^{16}$  sinapsi
  - Okidanje neurona otprilike na 100 Hz
  - Ukupno  $10^{16}$  do  $10^{18}$  veza u sekundi
-

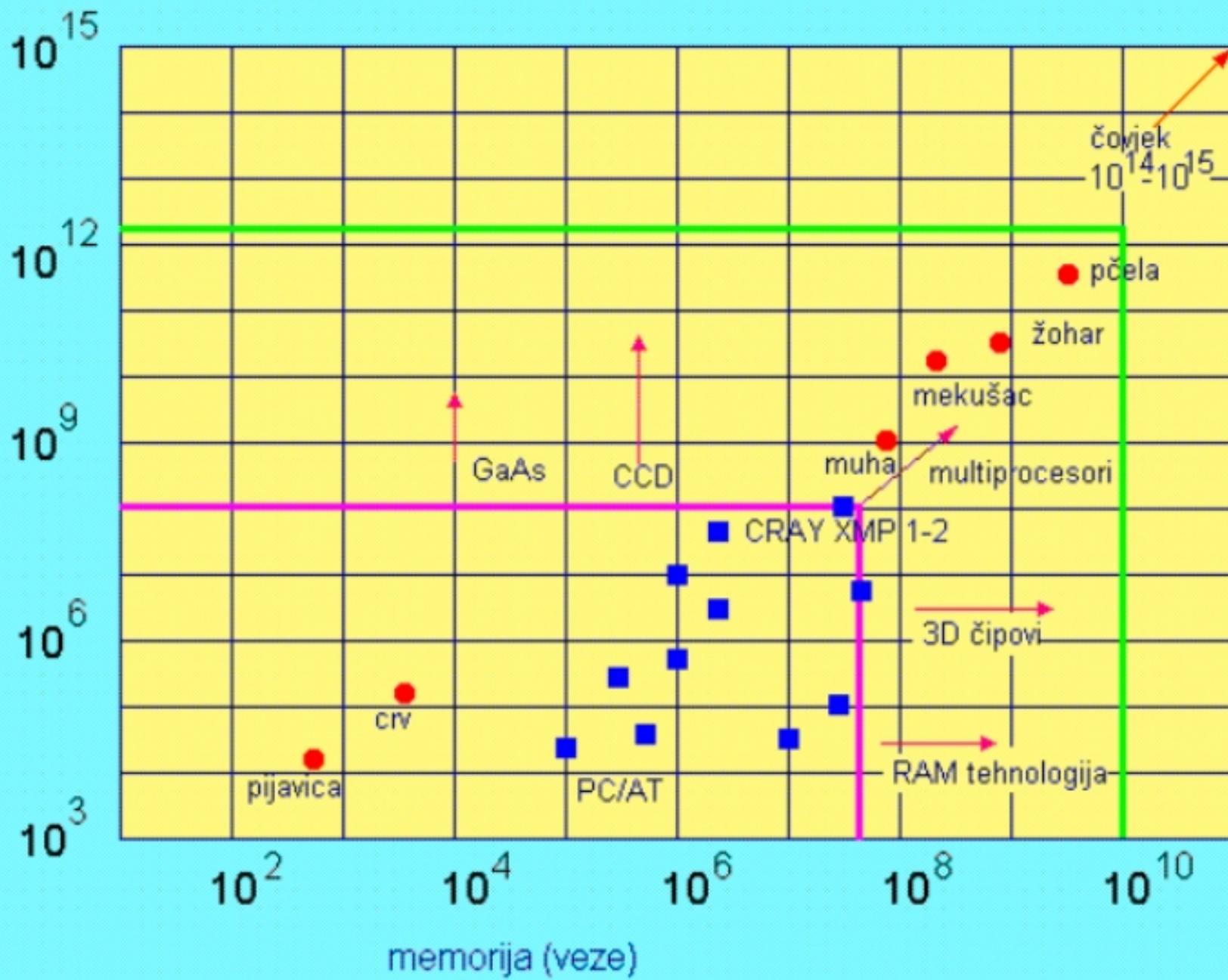
# Obrada signala u neuronima

---

- Neuroni primaju signale ostalih neurona
  - Neki se signali pojačavaju, a drugi oslabljuju
  - Svi pristigli signali se zbrajaju, kada zbroj pristiglih signala postigne neku razinu (prag) dolazi do okidanja naurona
  - Rezultat obrade u obliku signala se prosljeđuje dalje drugim neuronima na obradu
-

brzina (veza / s)

## USPOREDBA ŽIVIH I UMJETNIH SUSTAVA

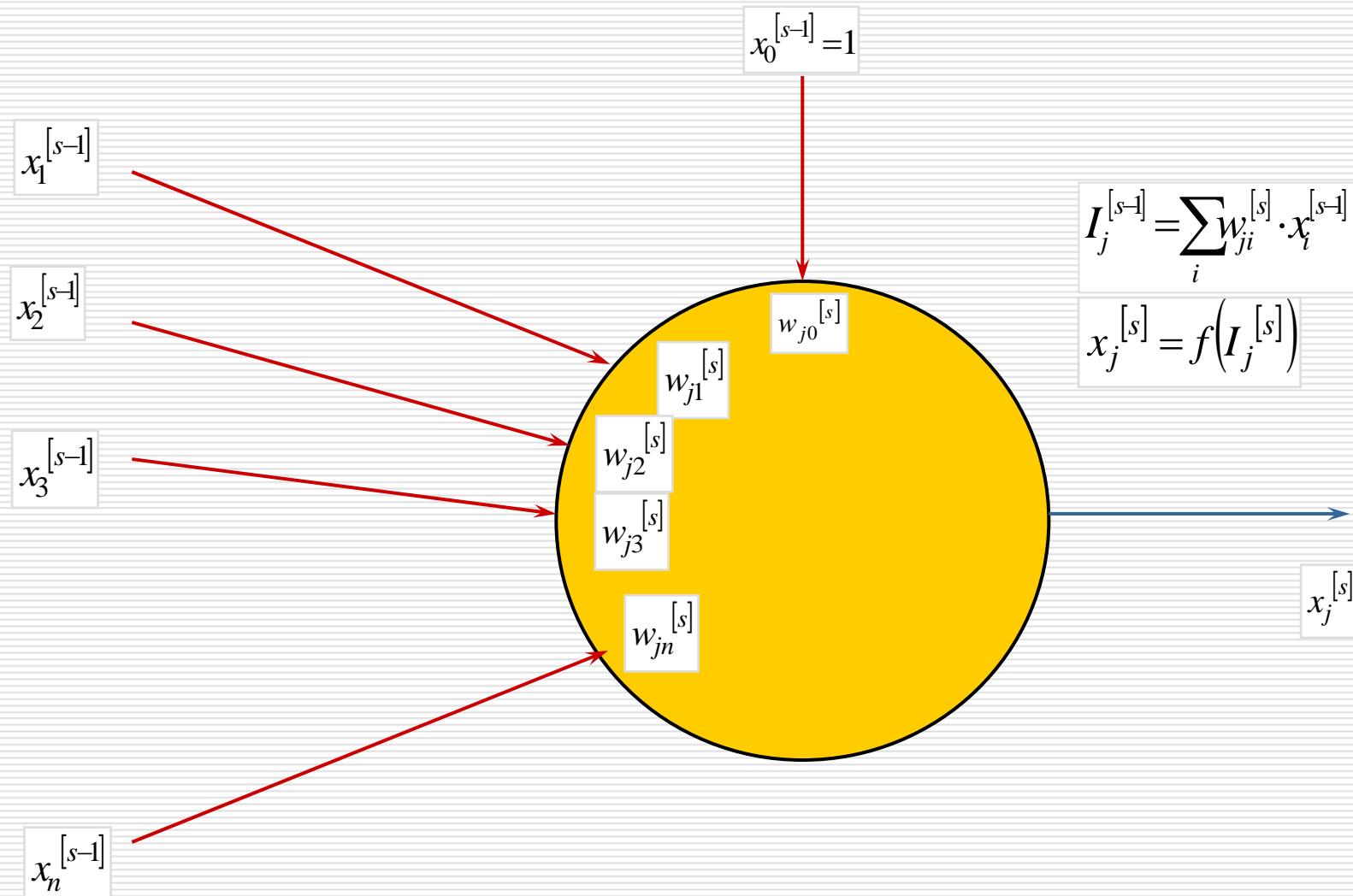


# Umjetni neuron

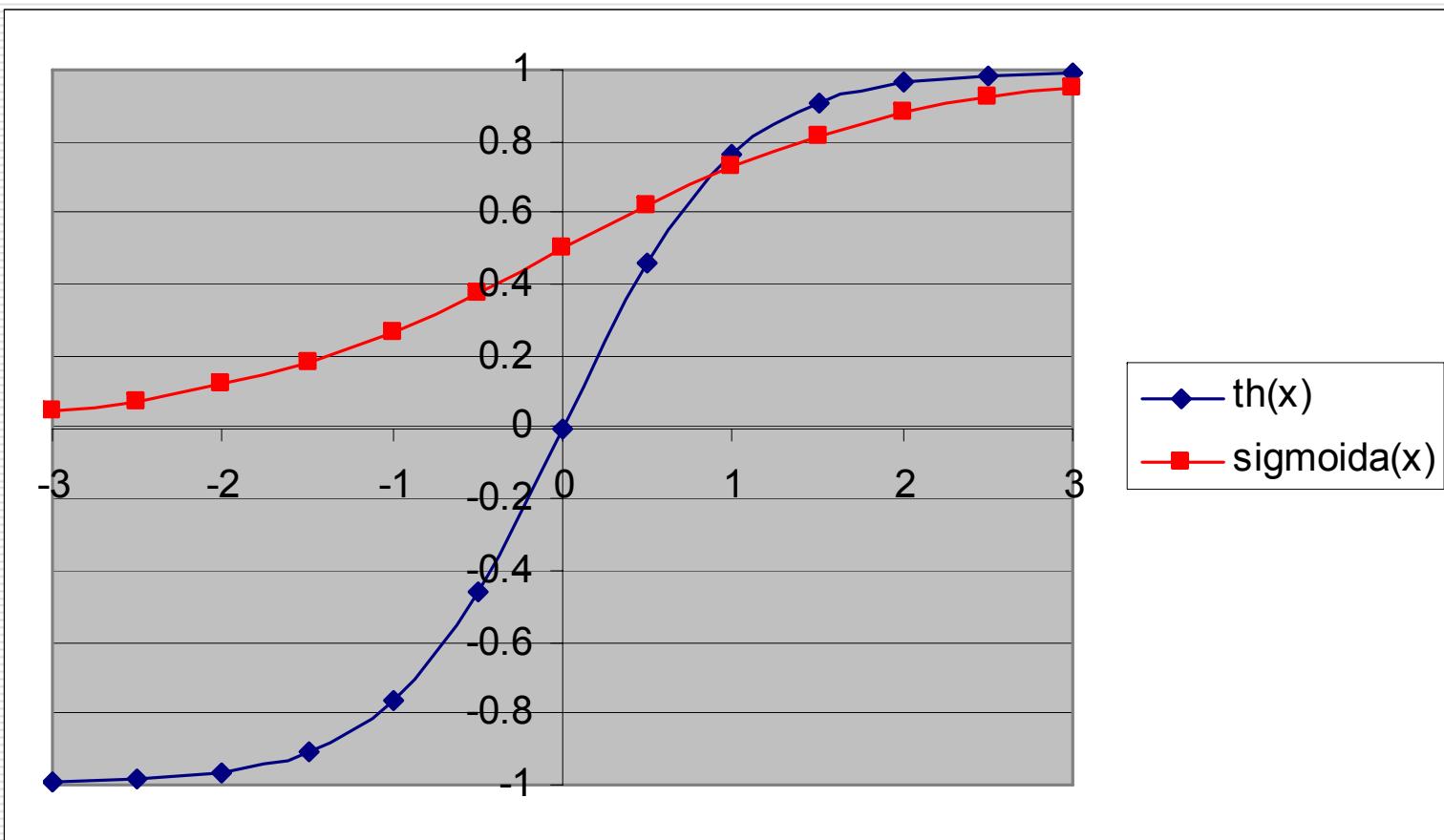
---

- Element obrade (PE - Processing Element)
  - Akumiliranje iz ostalih elemenata obrade u jedan zbroj signala
  - Zbroj signala s težinskim vrijednostima (interna aktivacija)
  - težine veza simuliraju biološke sinapse
  - funkcija prijenosa - sigmoida ili tangens hiperbolni
-

## Tipični element obrade kod mreže “širenje unatrag”



# Tipične funkcije prijenosa



# Neuronska mreža

---

- Kod učenja, težine se modificiraju (obično iterativno), da se postigne traženi izlaz.
  - Pravilo učenja - algoritam koji se koristi za podešavanje težina.
  - Širenje greške prema natrag (*back propagation*) je najpopularnije pravilo učenja.
  - Najšire korišteni model neuralne mreže je širenje prema natrag.
  - Širenje prema natrag (*back propagation*) se odnosi na algoritam za podešavanje težina veza u višestrukim slojevima mreže.
-

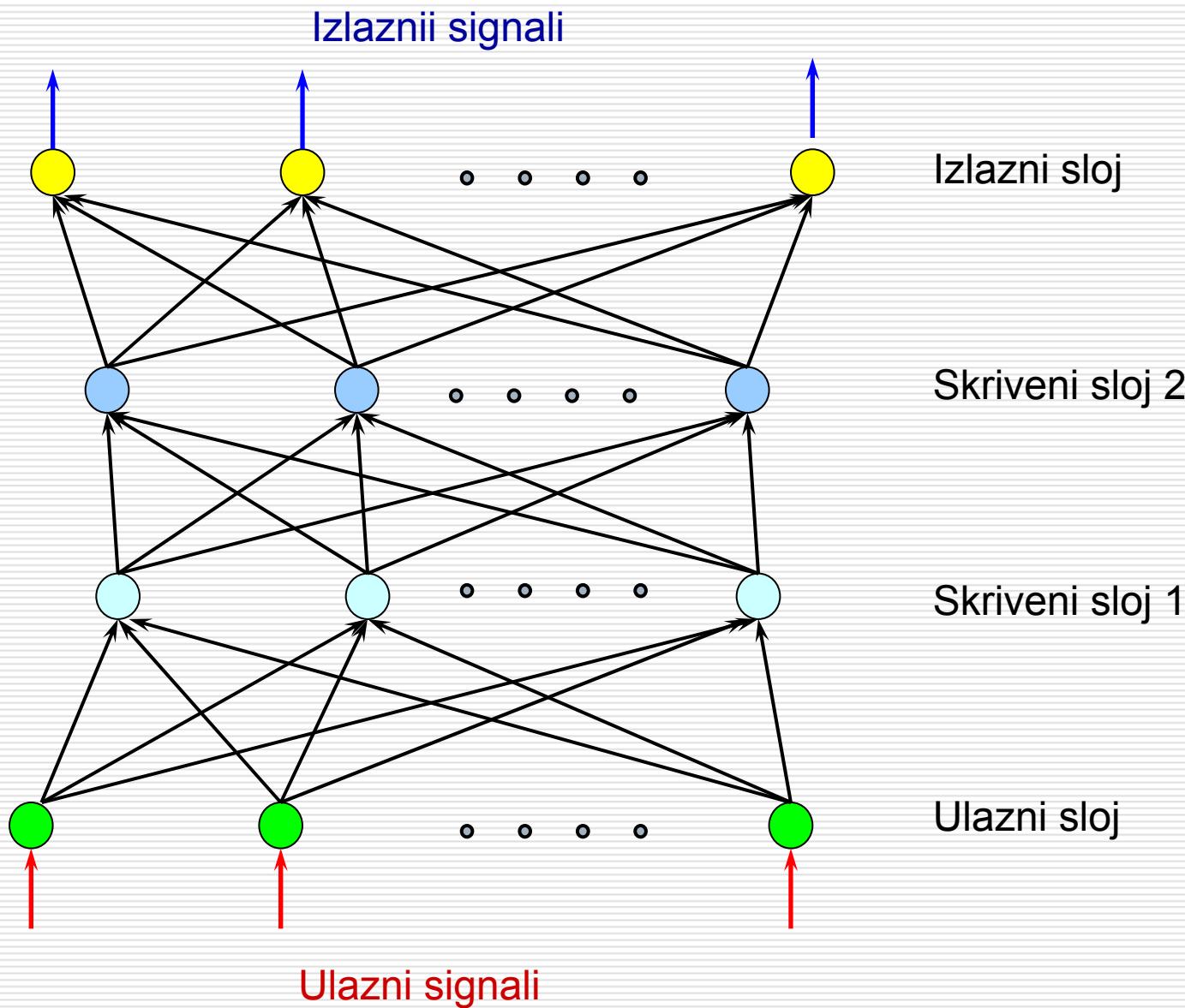
# Neuronska mreža

---

□ Elementi obrade su smješteni u slojevima:

- jedan ulazni sloj
  - jedan ili više međuslojeva
  - jedan izlazni sloj
-

## Tipična mreža “širenje unatrag”



# Neuronska mreža

$w_{jo}^{[s]}$

- Težina j-tog neurona u s-tom sloju prema nultom neuronom
- postoje odgovarajuće težine veza koje oslabljuju neurone

$x_j^{[s]}$

- trenutačno izlazno stanje j-tog neurona u s-tom sloju

# Neuronska mreža

$W_{ji}^{[s]}$

- Težina veze koja povezuje i-ti neuron u (s -1) - vom sloju sa j-tim nevronom u s-tom sloju

$I_j^{[s]}$

- interna aktivacija I ili zbroj ulaza pomnožen s težinama za j-ti neuron u s-tom sloju

# Neuronska mreža

- Svaki izlaz množimo s težinom veze i sve to zbrajamo

$$x_j^{[s]} = f\left( \sum w_{ji}^{[s]} \cdot x_i^{[s-1]} \right) = f(I_j^{[s]})$$

# Neuronska mreža

---

- Ulazne vrijednosti kod neuralne mreže širimo prema naprijed.
  - Učenje neuralne mreže
    - na neuralnu mrežu se dostavljaju ulazi prema uzorcima gdje za ulaze znamo kakvi će biti izlazi. Na temelju ulaza ona mora znati izlaz
    - kada NM nisu dobro naučene za ulaze daju nedovoljno dobre izlaze
-

# Neuronska mreža

- Trebamo definirati globalnu funkciju greške NM
  - uvodimo zato funkciju  $E$  kao globalnu funkciju greške neuronske mreže
  - od nje se traži da bude diferencijabilna s obzirom na  $w_i^{[s]}$
  - u dokazivanju koristimo tzv. kritični parametar koji zovemo  $e \rightarrow$  mjera lokalne greške koju definiramo za svaki neuron

$$e_j^{[s]} = \frac{-\partial E^{[s]}}{\partial I_j}$$

- ovisnost greške  $e$  j-tog neurona u s-tom sloju

# Umjetni neuron

$$e_j^{[s]} = f'(I_j^{[s]}) \cdot \sum_k (e_k^{[s+1]} \cdot w_{jk}^{[s+1]})$$

$$f(z) = (1.0 + e^{-z})^{-1}$$

$$f'(z) = f(z) \cdot (1.0 - f(z))$$

# Umjetni neuron

---

$$e_j^{[s]} = x_j^{[s]} \cdot (1.0 - x_j^{[s]}) \cdot \sum_k (e_k^{[s+1]} \cdot w_{kj}^{[s+1]})$$

- Vidimo kolika je mjera lokalne greške u sloju [s] u odnosu na sloj iznad njega [s+1]
  
  - Učenje NM je prilagođenje mjera težina veza  
Δ - (delta) korekcija težine veza
-

# Umjetni neuron

## □ Formula korekcije težine veza

$$\Delta w_{ij}^{[s]} = \left( \frac{-\partial E^{[s]}}{\partial I_j} \right) \cdot \left( \frac{-\partial I_j^{[s]}}{\partial w_{ij}^{[s]}} \right) = -e_j^{[s]} \cdot x_i^{[s-1]}$$

$$\Delta w_{ij}^{[s]} = lcoef \ e_j^{[s]} \cdot x_i^{[s-1]}$$

# Umjetni neuron

- Formule za globalnu grešku neuronske mreže i lokalnu grešku neurona

$$E = 0.5 \cdot \sum_k ((d_k - o_k)^2)$$

$$e_k^{(0)} = -\frac{\partial E^{(0)}}{\partial I_k} = -\frac{\partial E}{\partial o_k} \cdot \frac{\partial o_k}{\partial I_k} = (d_k - o_k) \cdot f'(I_k)$$

# Umjetni neuron

---

- Momentni član za korekciju nestabilnosti kod učenja:

$$\Delta w_{ji}^{[s]} = lcoef \cdot e_j^{[s]} \cdot x_i^{[s-1]} + momentum \quad \Delta w_{ji}^{[s]}$$

- Različite formule za globalnu grešku

$$E = \left( \frac{1}{3} \right) \cdot \sum_k |(d_k - o_k)|^3$$

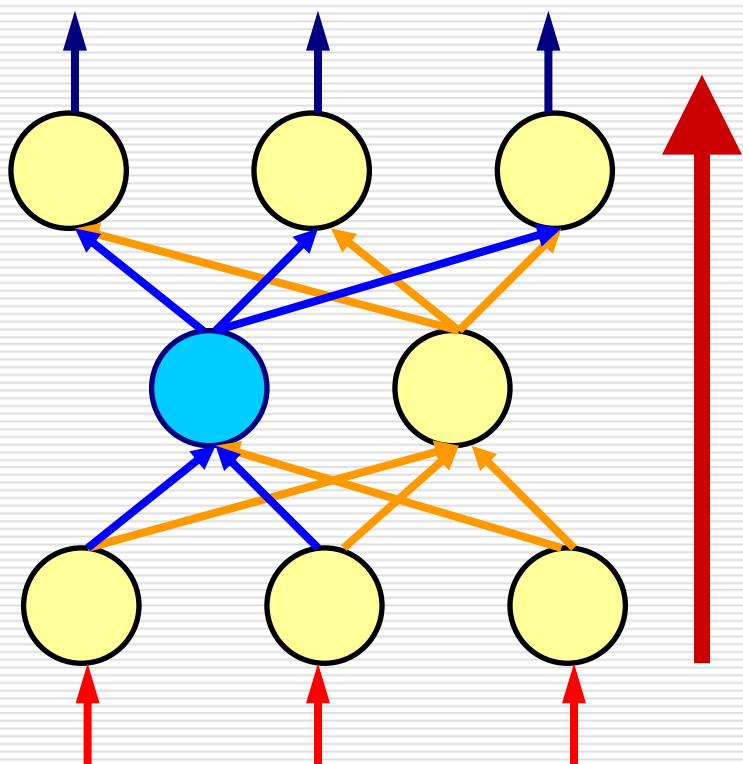
$$E = \left( \frac{1}{4} \right) \cdot \sum_k (d_k - o_k)^4$$

# Standardni algoritam - širenje unatrag

- Za dani ulazni vektor  $\underline{i}$  i izlazni vektor  $\underline{o}$  treba:
  1. Staviti  $\underline{i}$  na ulazni slog mreže. Prenositi tu vrijednost do izlaznog sloja i ostvariti izlazni vektor  $\underline{o}$ .
  2. Za vrijeme prenošenja ulaznog vektora, određuju se vrijednosti  $I_j$  i  $x_j$  svakog neurona u mreži.
  3. Za svaki element obrade u izlaznom sloju računati skaliranu lokalnu grešku po i računati delta težine prema.
  4. Za svaki sloj  $s$ , počevši od predzadnjeg sloja prije izlaznog, i sloja neposredno nakon ulaznog, računati skaliranu lokalnu grešku prema i delta težinu prema.
  5. Obnoviti sve težine veza u mreži dodavanjem delta težina prijašnjim vrijednostima.

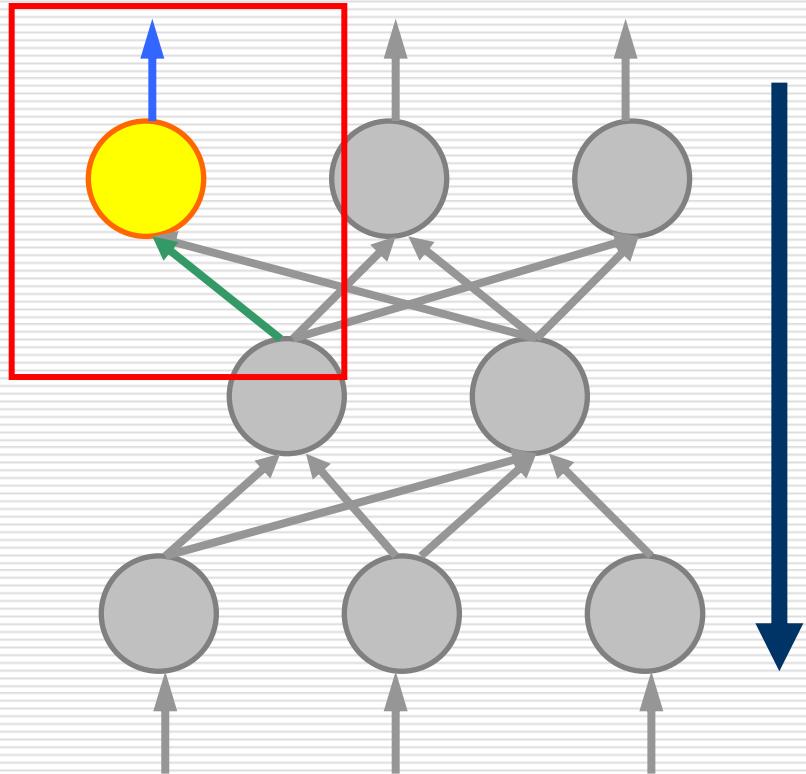
# Obrada u neuronskoj mreži

---



Veličine na ulazu u neuronsku mrežu se prosljeđuju u neurone, zbrajaju, te prosljeđuju neuronima u višim slojevima.

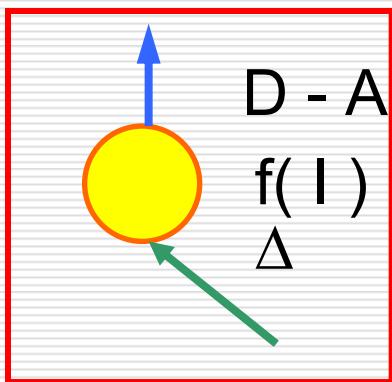
# Učenje neuronske mreže



Kod učenja greške iz gornjih slojeva se prosljeđuju na niže slojeve (prema ulazu) te korigiraju težine veza.

# Učenje

---



Greška nekog neurona u izlaznom sloju se računa po formuli ( $D$  - traženi izlaz iz neurona,  $A$  - dobiveni izlaz iz neurona):

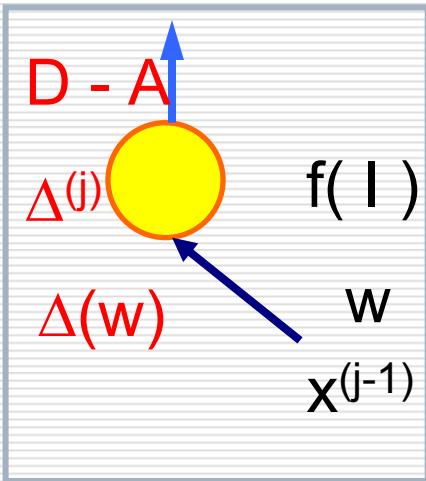
$$\Delta = f'(I) * (D - A)$$

Derivacija funkcije tangens hiperbolni  $f'$  se računa po formuli:

$$f'(I) = (1 + f(I)) * (1 - f(I))$$

# Učenje

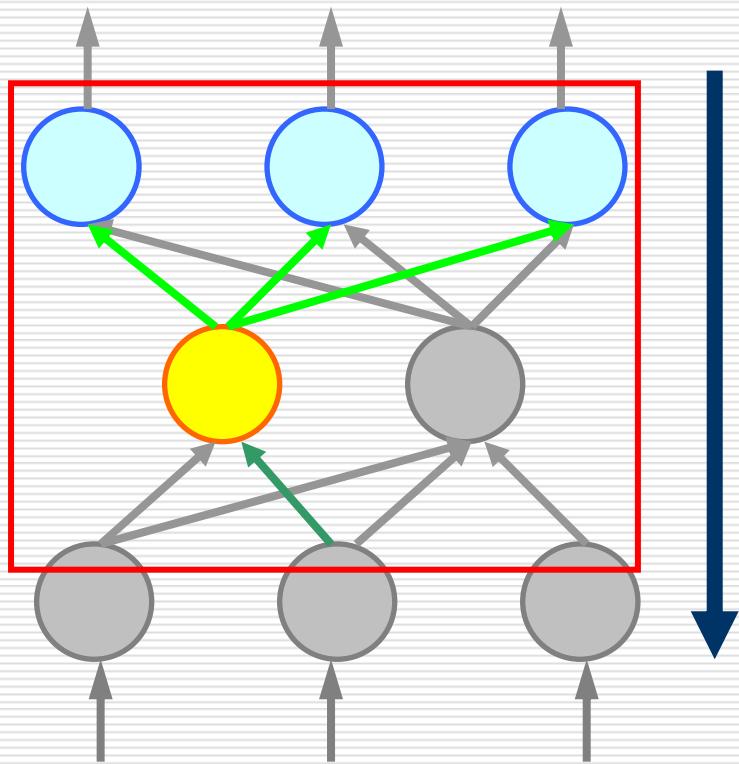
---



Korekcija težine veze se računa po formuli:

$$\Delta(w) = L_{\text{coef}} * \Delta^{(j)} * x^{(j-1)}$$

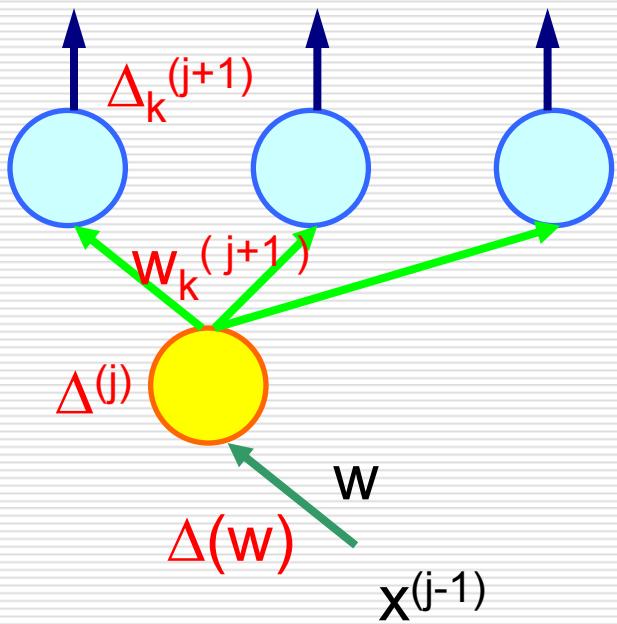
# Učenje neuronske mreže



Kod učenja greške iz gornjih slojeva se prosljeđuju na niže slojeve (prema ulazu) te korigiraju težine veza.

# Učenje

---



Greške neurona u **ostalim slojevima**, koji nisu izlazni računaju se po formuli:

$$\Delta^{(j)} = f'(\mathbf{l}) * \sum \Delta_k^{(j+1)} * w_k^{(j+1)}$$

Formula za korekciju težine veze je ista:

$$\Delta(w) = L_{\text{coef}} * \Delta^{(j)} * x^{(j-1)}$$

# Temeljne karakteristike neuronskih mreža

---

- učenje na temelju primjera
    - Hebbianovo učenje
    - učenje delta pravilom
    - kompetitivno učenje (u. natjecanjem)
  - distribuirana asocijativna memorija
  - tolerancija grešaka
  - prepoznavanje uzorka
  - sinteza
-

## Primjeri uspješne primjene:

- Detekcija eksploziva kod provjere prtljage na aerodromima (Shea i Lin)
- Identifikacija oblaka uporabom podataka satelita (Lee i Sengupta)
- Određivanje orijentacije satelita pomoću zvijezda (Alveda i Martin)
- Istovarivanje kamiona (Nguyen i Widrow)
- Obrada signala (Kwan i Lee)
- Dijagnostika kod mlaznih motora (Dietz, Kiech i Ali)
- Prepoznavanje govora (Franzini, Witbrock, Lee)
- Interpretacija sonara (Gorman i Sejnovski)
- Strojno gledanje (Glover i Rosenberg)
- Pretvaranje engleskog teksta u glasove (Sejnowski i Rosenberg)

# Tehnologije za implementaciju neuralnih računalnih sustava

---

- programski simulatori (P3, NeuralWorks)
  - sklopovski ubrzivači (Mark III i Mark IV, Network Emulation Processor NEP, Balboa, Delta-II procesor za pokretni zarez)
  - silikonski čipovi
  - optički procesori
-

# Izbor modela neuronske mreže

---

- Značenje pojedinih stavki u tablici:
    - Vrsta problema
      - Classify Klasifikacija
      - Predict Predviđanje/sinteza funkcija
  - Objasniti tablicu
-

# Izbor modela neuronske mreže

---

- Značenje pojedinih stavki u tablici:
  - Karakteristike mreže
    - Novelty Detector      Mogućnost pronalaženja podataka izvan domene treniranih podataka
    - Zahtjev za memoriju      Kompaktnost; brzi poziv
    - Brzina učenja      Učinkovitost on-line učenja
    - Modelling Ability      Praktični opseg mogućnosti modeliranja mreže
    - Differantiability      Diferencijabilnost *feedforward* funkcije

# Prednosti i nedostaci mreže *back propagation*

---

- Prednosti:
  - ima dodatne slojeve koji dopuštaju da se rezultat jednog sloja dodatno obrađuje, uređuje i stvara kompleksni sustav
- Nedostaci:
  - dugotrajno treniranje i osjetljivost na početne vrijednosti težina
  - algoritmi treniranja mreže su dugotrajni i ne osiguravaju konvergenciju

# Prednosti i nedostaci mreže *back propagation*

---

## □ Nedostaci:

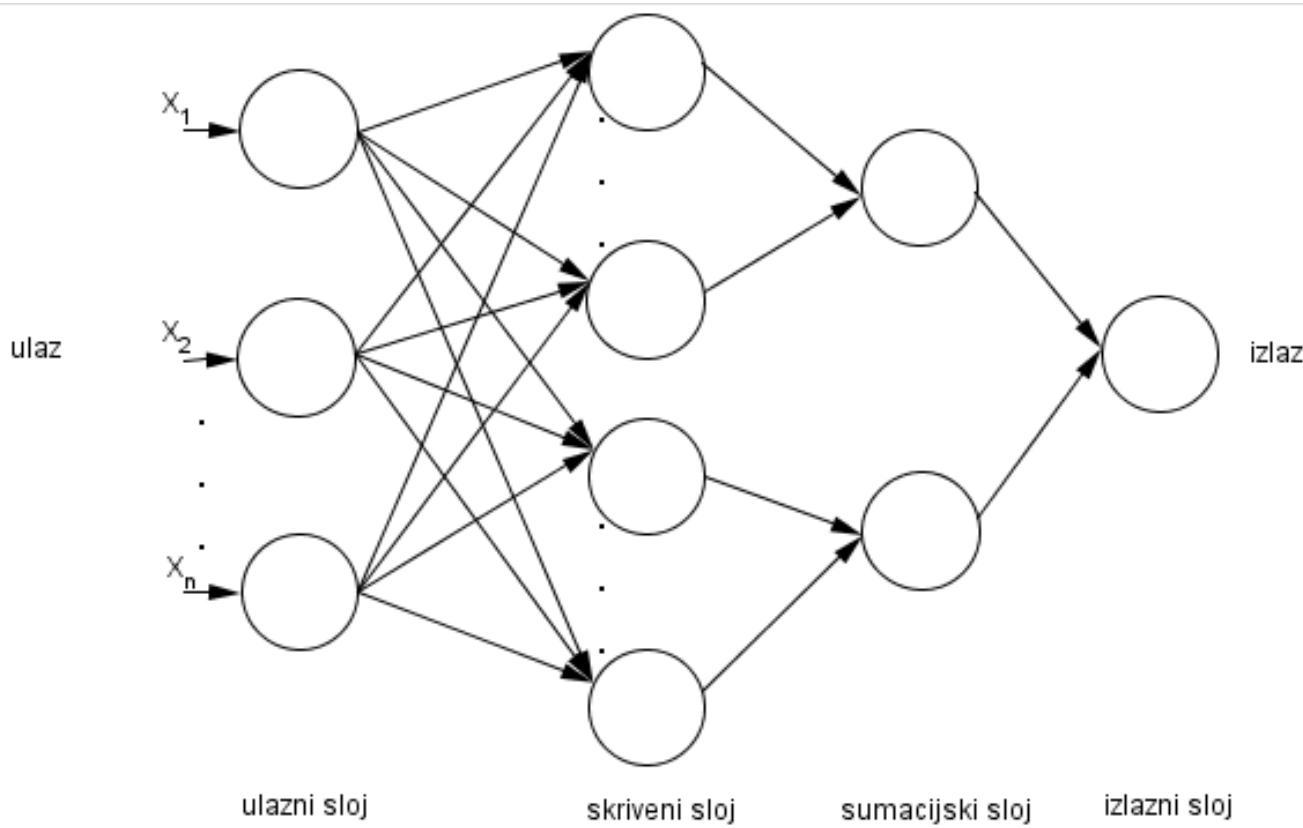
- ukoliko mreža ne sadrži dovoljno skrivenih neurona, nije dovoljno složena za rješavanje problema aproksimacije funkcije i podilazi podaccima te proizvodi pretjeranu grešku na izlazu
  - ukoliko mreža ima previše skrivenih neurona tada postoji mogućnost pretjeranog prilagođavanja , gdje se mreža dobro uvježbava na uzorku za treniranje, a slabo na uzorku za testiranje
  - teško je predvidjeti grešku testiranja iz treniranih podataka
  - određivanje optimalnog broja treniranja je teško, a rješenje ovog problema uključuje zaustavljanje treniranja kada greška validacije počinje rasti
-

# Vjerojatnostna mreža

---

- Vjerojatnostna mreža (PNN probabilistic neural netowork, Donald Specht 1988.) koristi statistički pristup, tzv. Bayesove klasifikatore i Parzenove procjenjivače, koji računaju funkcije gustoće vjerojatnosti koje zahtjeva Bayesova teorija razmatrajući relativne vjerojatnosti događaja i koristeći dane informacije, kako bi poboljšala predviđanje i minimizirala očekivanu grešku.
  - Svaki element se trenira jednom, a za učenje se koristi nadgledano učenje.
-

# Arhitektura vjerojatnosne mreže



## Opis vjerojatnostne mreže

---

- Ulazni sloj sadrži onoliko elemenata koliko ima parametara za opis objekta koji se klasificira
  - Ulazni sloj je u potpunosti povezan sa skrivenim slojem, koji se sastoji od uzorka za treniranje
  - Izlazni sloj sadrži onoliko analiziranih elemenata koliko ima kategorija koje se određuju
  - Skriveni sloj nije u potpunosti povezan s izlaznim slojem (svaki neuron skrivenog sloja povezan je s jednim neuronom izlaznog sloja)
  - Skriveni sloj je implementaciju verzije Bayesovog klasifikatora, gdje su ovisne funkcije gustoće vjerojatnosti kategorije
-

## Opis vjerojatnostne mreže

---

- Svaki element skrivenog sloja trenira se jednom kako bi generirao visoku vrijednost izlaza kad se ulazni vektor usporedi s vektorom treniranja
  - Svaka kategorija u koju se mogu svrstati ulazni podatci predstavljena je u izlaznom sloju gdje se klasifikacija obavlja po principu „pobjednik uzima sve“
-

# Matematički model vjerojatnostnne mreže

---

- Na ploči.
-

# Karakteristike vjerojatnostne mreže

---

## Prednosti:

- Faza treniranja vjerojatnostne mreže je znatno brža, jer se treniranje većinom provodi u jednom prolasku.
- Vjerojatnostna mreža dozvoljava proširivanje skupa za treniranje u bilo kojem trenutku, a cijela se mreža ne mora ponovno trenirati

## Nedostatak:

- Vjerojatnostna mreža može postati jako velika i spora prilikom treniranja velikog uzorka, pa time postane nepraktična za rješavanje problema klasifikacije
-

# Mreža učeće vektorske kvantizacije

---

- **Mreža učeće vektorske kvantizacije** (learning vector quantization LVQ, Kohonen, 1986. godine), temelji se na Kohonovim slojevima koji imaju sposobnost kategoriziranja objekata u kategorije po sličnosti.
  - Predloženo je više modifikacija tog algoritma (LVQ1, LVQ2, LVQ2.1, LVQ3, LVQ4 i druge), čime se željelo poboljšati utvrđivanja klasifikatora.
  - Svrha LVQ je definiranje klasa u području ulaznih podataka, a namijenjena je isključivo za klasifikaciju ili metodu prepoznavanja.
  - LVQ je hibridna mreža, jer može koristiti nadgledano i nenadgledano učenje za klasifikaciju (neuronima mreže može biti pridodana oznaka kategorije)
-

# Arhitektura mreža učeće vektorske kvantizacije

---

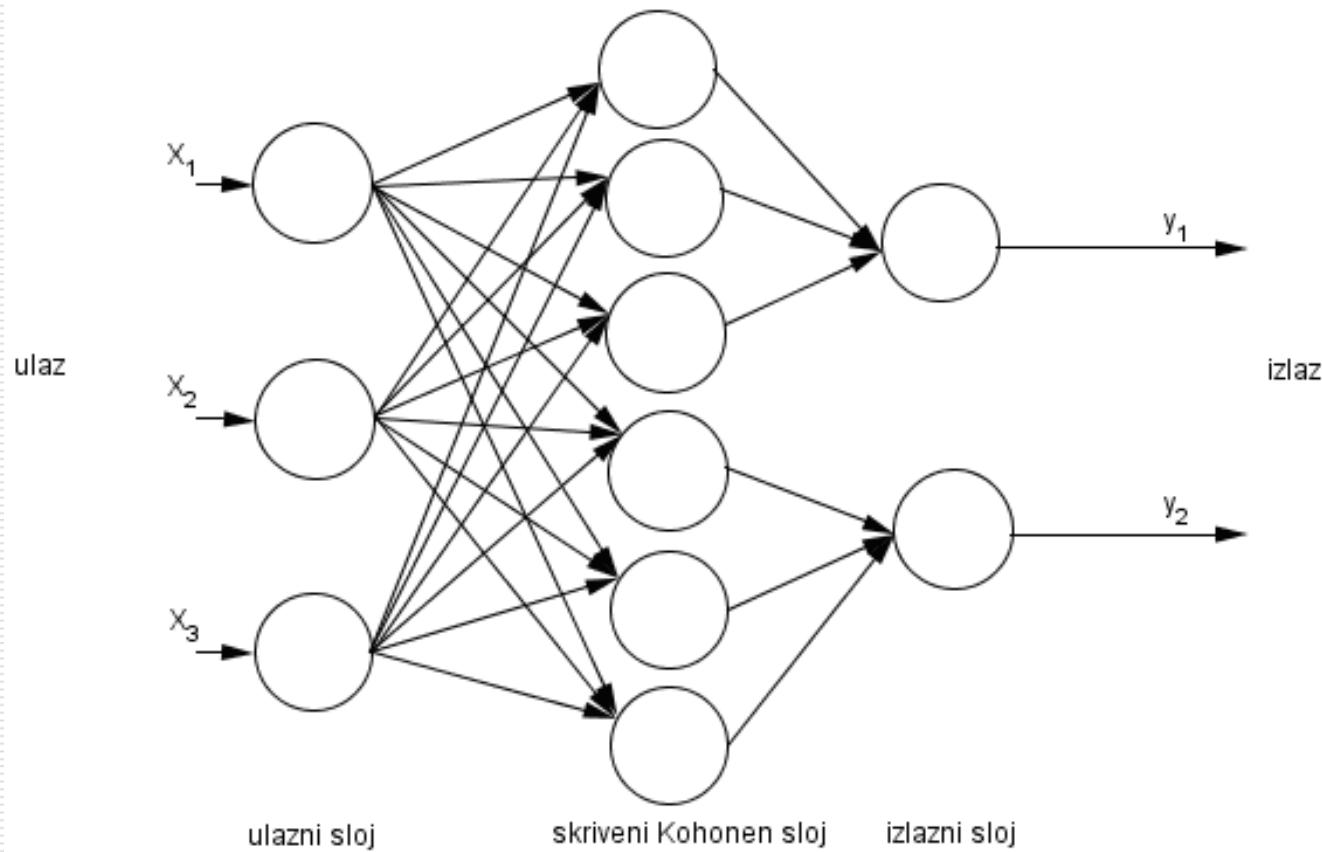
- Arhitektura mreže sastoji se od:
  - ulaznog sloja,
  - Kohonovog sloja i
  - izlaznog sloja

# Arhitektura mreža učeće vektorske kvantizacije

---

- Ulazni sloj** je u potpunosti povezan sa skrivenim Kohonovim slojem
  - Kohonenov sloj** grupira određeni broj analiziranih elementa za svaku klasu ovisno o složenosti odnosa ulazno-izlaznih elemenata
  - Izlazni sloj** sadrži onoliko elemenata koliko ima različitih kategorija.
  - Vektorska kvantizacija je tehnika koja istražuje strukturu ulaznih vektora zbog kompresije podataka.
  - Većinom svaka klasa ima isti broj elemenata u svim slojevima.
  - Svaki neuron ulaznog sloja pridružen je nekoj kategoriji.
-

# Arhitektura LVQ mreže



# Učenje mreže učeće vektorske kvantizacije

---

- Prilikom faze treniranja nadgledana mreža koristi Kohonenov sloj tako da izračuna udaljenost vektora treniranja do svakog analiziranog elementa.
  - Najbliži element omogućuje samo jedan izlazni element za cijeli sloj te ukoliko je taj element u očekivanoj klasi treniranog vektora, pojačava se prema vektoru treniranja, a ukoliko nije u očekivanoj klasi tada se težine koje ulaze u obrađivani element uklanjaju od vektora treniranja i ta operacija zove se odbijanje.
  - Prilikom treniranja mreže, pobjednički neuron mreže, koji je i najbliži neuron u ulaznom dijelu uzorka treniranja, pomiče se prema tom uzorku treniranja i sa sobom povlači susjedne neurone što dovodi do ravnomjerne distribucije treniranih podataka.
  - Ukoliko pobjednik nije u ispravnoj klasi, odmiče se od nje.
-

# Matematički model mreže učeće vektorske kvantizacije

---

- Na ploči.
-

# Prednosti i nedostaci mreže učeće vektorske kvantizacije

---

- Nedostaci LVQ arhitekture su:
  - za složene probleme klasifikacije koji imaju slične ulazne vektore ili objekte, mreža zahtjeva veliki Kohonenov sloj s mnogo analiziranih elemenata po klasi (može se smanjiti broj ulaznih parametara, tako da se pooštiri izbor).
  - kod jednostavnih oblika LVQ mreže neki analizirani elementi prečesto pobjeđuju dok drugi ne rade ništa (dodaje se tzv. mehanizam savjesti koji je proporcionalan razlici između frekvencije pobjeđivanja elementa i prosječne frekvencije pobjeđivanja analiziranih elemenata te se dodaje svakom analiziranom elementu, pa element koji često pobjeđuje bude kažnjen).
- LVQ se najčešće koristi za probleme klasifikacije
  - Korisnost vektorske kvantizacije koja čuva odnose između centara primjenjuje se i kod komunikacija, gdje šum dodan kodiranom vektoru može izmijeniti reprezentaciju.

# Mreža s radijalno zasnovanom funkcijom

---

- „Radijalno zasnovana funkcija“ (RBF) - funkcije koje koristi ova mreža su radijalno simetrične, odnosno svaki neuron proizvodi identičan izlaz koji se nalazi na određenoj radijalnoj udaljenosti od centra

## Arhitektura RBF mreže

---

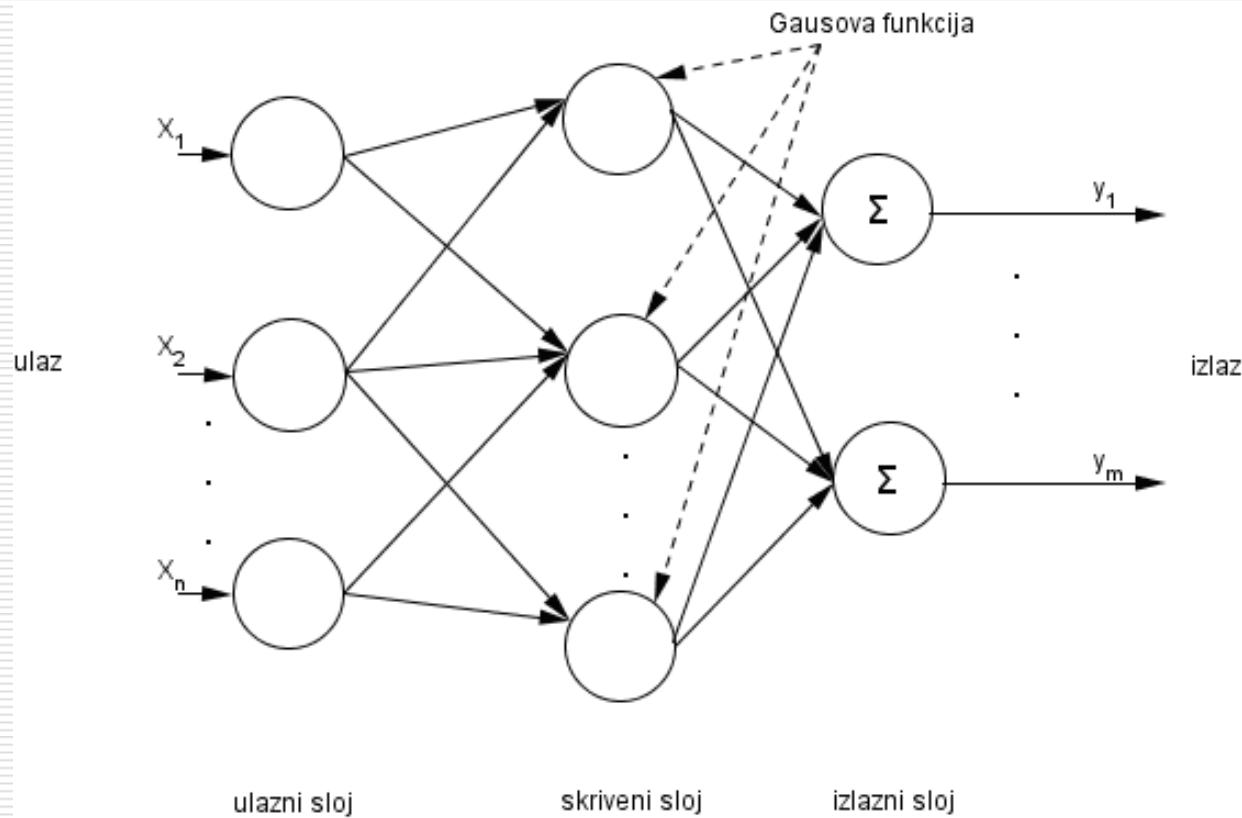
- Ulazni sloj ima funkciju primatelja ulaznih podataka, a u skrivenom sloju obavlja se nelinearna transformacija te neuroni u izlaznom sloju samo izračunavaju linearu kombinaciju izlaznih vrijednosti neurona u skrivenom sloju.
  - Mreža s radijalno zasnovanom funkcijom koristi radijalnu osnovu ili Gaussovu funkciju kao prijenosnu funkciju.
  - Arhitektura sadrži tri sloja neurona s potpuno različitim ulogama: ulazni sloj, skriveni sloj i izlazni sloj.
  - Uobičajeno je da postoji samo jedan skriveni sloj neurona čija se prijenosna funkcija odabire iz kategorije funkcija koje se zovu osnovne funkcije
-

## Arhitektura RBF mreže

---

- Parametri koji određuju učenje kod RBF su :
    - težine između osnovnih funkcija u skrivenom sloju i izlaznog sloja
    - pozicija centara
    - razmještaj skrivenih neurona.
  - RBF mreže su prikladne za aproksimaciju funkcija, jer je konvergencija brža zbog linearnosti parametara izlazne greške.
-

# Arhitektura RBF mreže



# Učenje RBF mreže

---

- Parametri RBF mreže mogu se razvrstati u dvije kategorije: parametre prvog sloja i parametre funkcije s radijalnim značajem (centri i standardne devijacije za Gaussove funkcije).
  - Treniranje RBF mreže uključuje dvije faze:
    1. svakom neuronu dodijeljeni su centar i dijametar
    2. podešava se vektor težine.
  - Najpopularniji način pronalaženja centara dijeljenje ulaznih vektora u skupine i pronalaženje centra svake skupine te tada lociranje neurona u skrivenom sloju.
-

## Prednosti i nedostaci RBF mreže

---

- Izvrsno aproksimiraju i mogu se istrenirati jednostavno i brzo, no daju slab odaziv u fazi ponavljanja zbog velikog broja neurona povezanih s drugim slojem.
  - Linearne težine povezane s izlaznim slojem mogu se obrađivati odvojeno od skrivenog sloja neurona.
  - Težine skrivenog sloja prilagođavaju se kroz nelinearnu optimizaciju, a težine izlaznog sloja kroz linearnu optimizaciju.
  - Ne preporuča se korištenje RBF za pronalaženje rješenja ukoliko imamo mali skup podataka.
  - RBF je idealna mreža za rješavanje klasifikacijskih problema.
-

# Izbor modela neuronske mreže

---

- Značenje pojedinih stavki u tablici:
  - Karakteristike mreže
    - *Novelty Detector* - mogućnost pronalaženja podataka izvan domene treniranih podataka
    - Zahtjev za memoriju - kompaktnost; brzi poziv
    - Brzina učenja - učinkovitost on-line učenja
    - *Modelling Ability* - praktični opseg mogućnosti modeliranja mreže
    - Differantiability - diferencijabilnost *feedforward* funkcije
    - Sparse - oskudnost podataka

Network Type	Problem Type		Data Constraints			
	Classify	Predict	Noisy	Many Inputs	Sparse	Non-stationary
Back-Propagation	A	B				
Counter-Propagation						
Fuzzy ARTMAP						
General Regression						
Logicon Projection						
LVQ						
Modular						
Probabilistic						
Radial Basis Function	A					
Reinforcement	A					
SOM/Output Map	A					

Network Characteristics				
Novelty Detector	Memory Requirements	Learning Speed	Modeling Ability	Differentiability
	C			
	C			
D	D			D

# Učenje

- 
- Postoje dvije vrste učenja:
    - Temeljeno na pridruživanju novih informacija k prethodno prikupljenom znanju.
    - Temeljeno na pronalaženju korisnih zakonitosti u podacima (*database mining* ili prekapanje po podacima).
-

# Stabla odlučivanja

- 
- Učenje izradom identifikacijskih stabala:
    - najšire primjenjena metoda
    - rezultat testova se ugrađuju u identifikacijsko stablo
    - iz stabala se mogu stvarati skupovi pravila tipa prethodnik - sljedbenih (uzorak-posljedica, premisa-konkluzija, prednjak-zaglavak)
-

# Indukcija

- Indukcijom se uporabom određenih primjera postižu opći zaključci.
- (Klaić: **Indukcija** je zaključivanje od pojedinačnih slučajeva na općeniti izvod, od pojedinačnih fakata na općenite.)
- Stablo odlučivanja je poredak testova koji određuju najprikladniji test kod svakog koraka analize.

# Stablo odlučivanja

---

- **Stablo odlučivanja** je semantičko stablo u kojem je svaki čvor povezan sa skupom mogućih odgovora.
    - svaki čvor (koji nije list) je povezan sa testom koji odvaja skup svojih mogućih odgovora u podskupove koji odgovaraju različitim rezultatima testa
    - svaka grana vodi k posebnom podskupu rezultata testova u slijedećem čvoru.
-

# Učenje gradnjom identifikacijskih stabala

---

- Metoda koja omogućuje računalima učenje, izradom testova u identifikacijskom stablu.
  - Transformacija identifikacijskih stabala u skup pravila prethodnik-sljedbenik.
- 
- Kako izraditi identifikacijsko stablo traženjem pravilnosti u podacima?**
-

## Primjer. Posljedice sunčanja

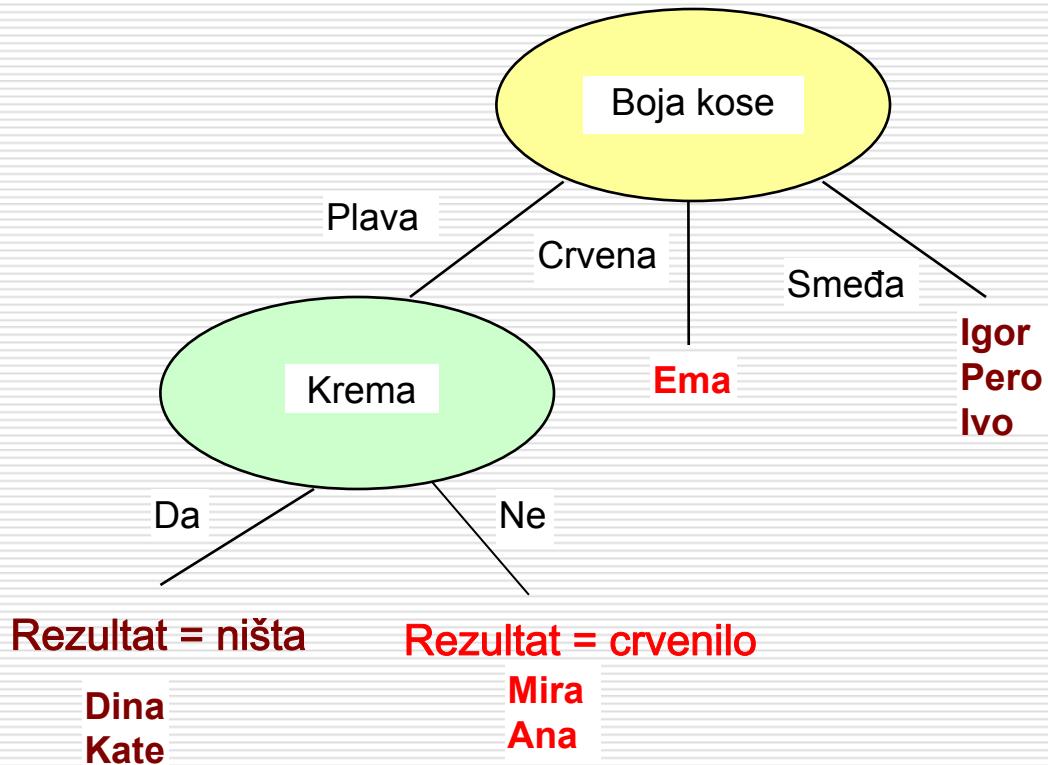
---

<b>Ime</b>	<b>Kosa</b>	<b>Visina</b>	<b>Težina</b>	<b>Krema</b>	<b>Rezultat</b>
<b>Mira</b>	plava	srednja	lagana	ne	crvenilo
<b>Dina</b>	plava	visoka	srednja	da	ništa
<b>Igor</b>	smeđa	niska	srednja	da	ništa
<b>Ana</b>	plava	niska	srednja	ne	crvenilo
<b>Ema</b>	crvena	srednja	teška	ne	crvenilo
<b>Pero</b>	smeđa	visoka	teška	ne	ništa
<b>Ivo</b>	smeđa	srednja	teška	ne	ništa
<b>Kate</b>	plava	niska	lagana	da	ništa

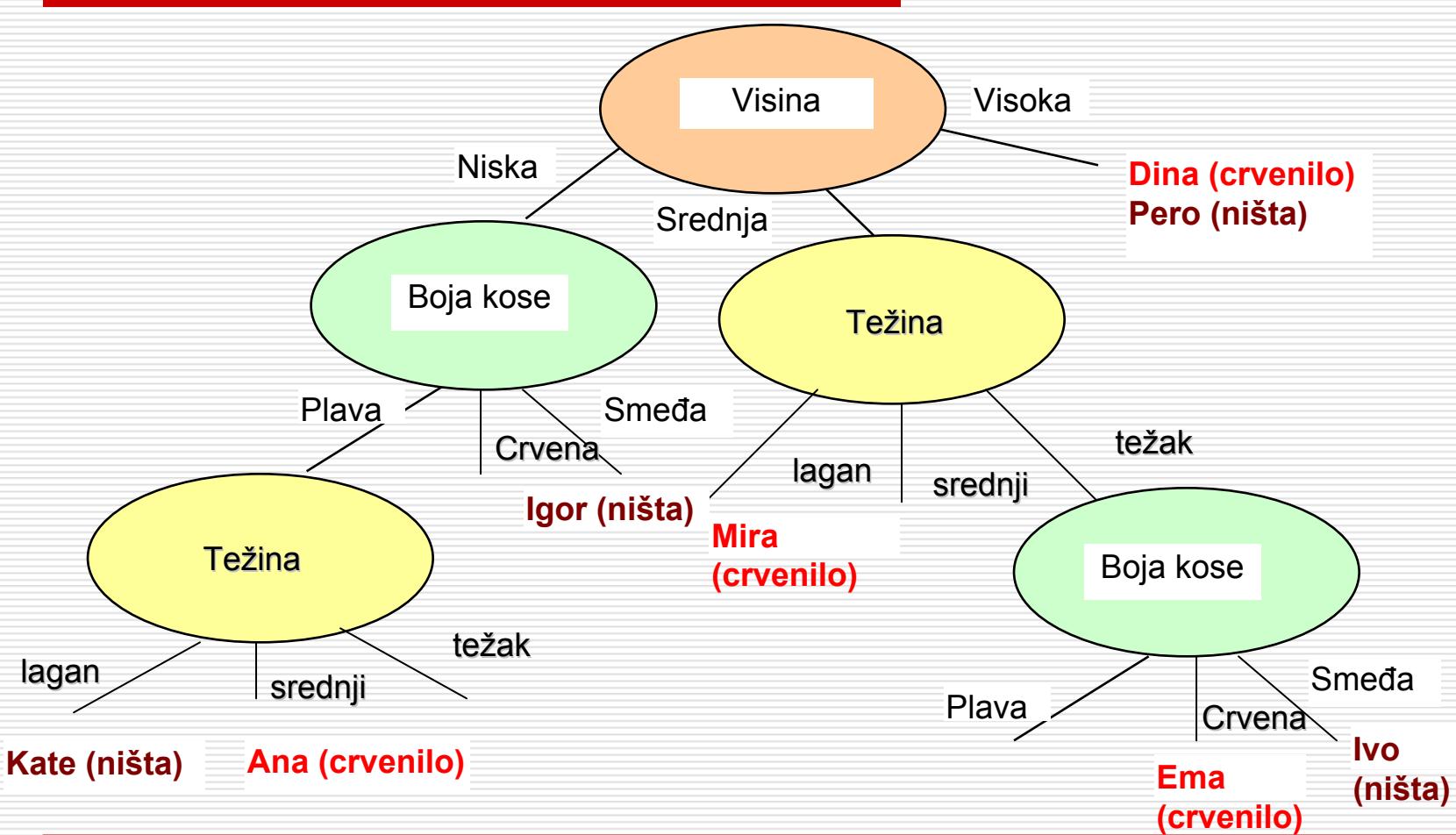
---

# Najjednostavnije stablo odlučivanja

---



# Slijedeće moguće stablo odlučivanja



# Učenje gradnjom identifikacijskih stabala

---

- Stvaranje različitih stabala odlučivanja:** neka su jednostavna, a druga komplikirana
  - Occamova britva** (William Occam - engl. filozof) - svijet je svojstveno jednostavan.
  - Occamova britva u učenju identifikacijskih stabala:** Postoji najmanje identifikacijsko stablo koje s najvećom vjerojatnošću ispravno identificira nepoznate objekte.
-

# Stvaranje najjednostavnijeg stabla

- Formula entropije. Mjerenje ukupnog nereda ili nehomogenosti iz baza podataka.

$$E = \sum_b (n_b/n_t) (\sum_c - n_{bc}/n_b \log_2(n_{bc}/n_b))$$

$E$  - (entropija) mjera nereda

$n_t$  - ukupan broj primjera u svim granama

$n_b$  - broj primjera u grani  $b$

$n_{bc}$  - je ukupan broj primjera u grani  $b$  klase  $c$ .

$$n_t = 13$$

Prva grana

$$n_1 = 7$$

$$n_{11} = 5$$

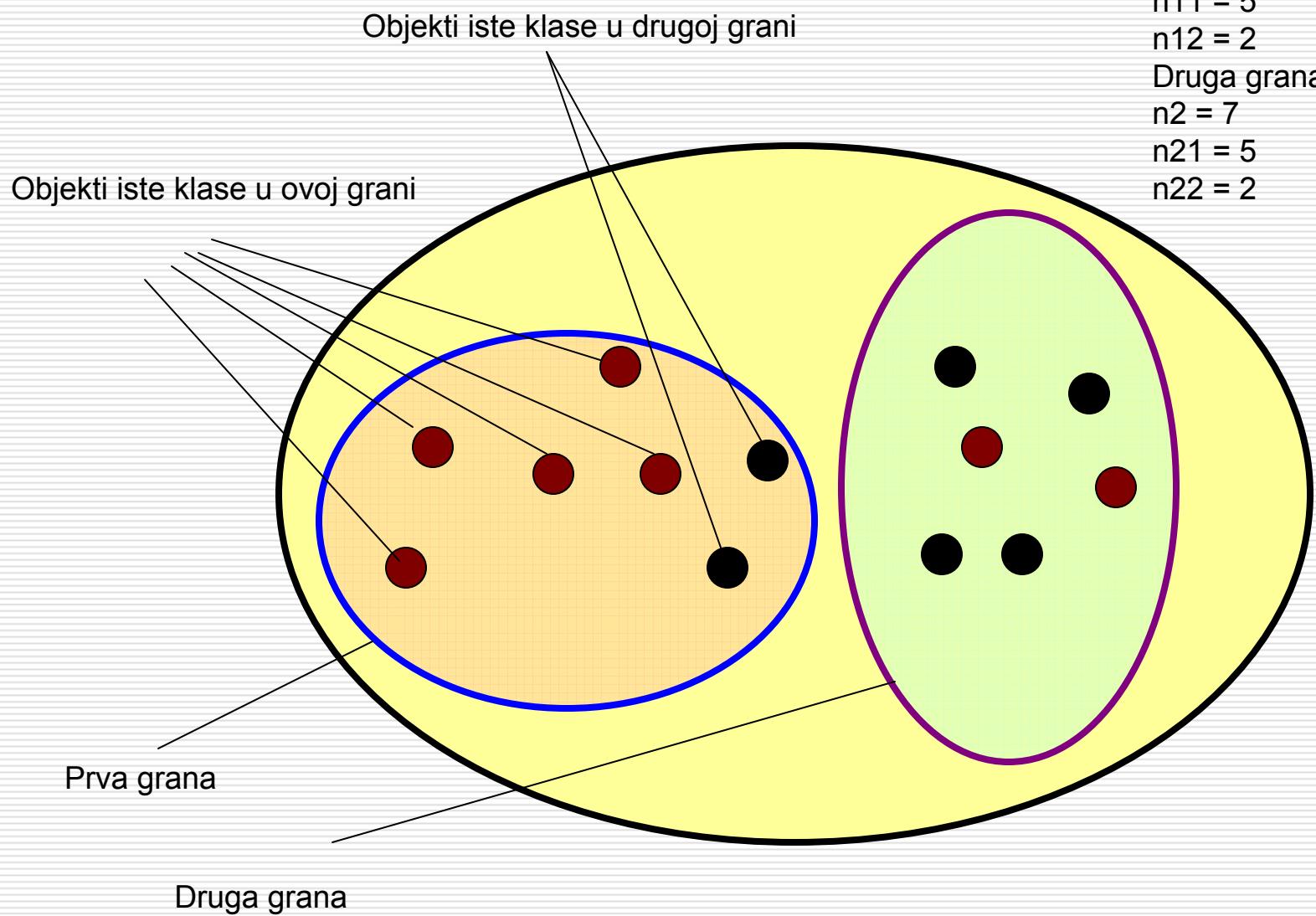
$$n_{12} = 2$$

Druga grana

$$n_2 = 7$$

$$n_{21} = 5$$

$$n_{22} = 2$$



# Učenje gradnjom identifikacijskih stabala

---

- Želimo formulu koja će dati veliki broj kada test daje nehomogeni skup i mali broj kada test daje homogeni skup.
-

# Kosa - najmanja entropija

<b>Ime</b>	<b>Kosa</b>	<b>Visina</b>	<b>Težina</b>	<b>Krema</b>	<b>Rezultat</b>
<b>Mira</b>	plava	srednja	lagana	ne	crvenilo
<b>Dina</b>	plava	visoka	srednja	da	ništa
<b>Ana</b>	plava	niska	srednja	ne	crvenilo
<b>Kate</b>	plava	niska	lagana	da	ništa

<b>Ime</b>	<b>Kosa</b>	<b>Visina</b>	<b>Težina</b>	<b>Krema</b>	<b>Rezultat</b>
<b>Igor</b>	smeđa	niska	srednja	da	ništa
<b>Pero</b>	smeđa	visoka	teška	ne	ništa
<b>Ivo</b>	smeđa	srednja	teška	ne	ništa

<b>Ime</b>	<b>Kosa</b>	<b>Visina</b>	<b>Težina</b>	<b>Krema</b>	<b>Rezultat</b>
<b>Ema</b>	crvena	srednja	teška	ne	crvenilo

# Krema - za plavokose

---

<i>Ime</i>	<i>Kosa</i>	<i>Visina</i>	<i>Težina</i>	<i>Krema</i>	<i>Rezultat</i>
<b>Mira</b>	plava	srednja	lagana	ne	crvenilo
<b>Ana</b>	plava	niska	srednja	ne	crvenilo

<i>Ime</i>	<i>Kosa</i>	<i>Visina</i>	<i>Težina</i>	<i>Krema</i>	<i>Rezultat</i>
<b>Dina</b>	plava	visoka	srednja	da	ništa
<b>Kate</b>	plava	niska	lagana	da	ništa

---

## ID3 (*Iterative Dichotomiser 3*) algoritam

---

- ID3 (*Iterative Dichotomiser 3*) je algoritam za generiranje stabla odlučivanja, otkriven od strane Rossa Quinlana. ID3 je prethodnik algoritma C4.5.
- Koraci algoritma:
  1. Preuzmi sve nekorištene atribute i izračunaj njihovu entropiju u odnosu na uzorke testova
  2. Izaberi atribut za koji je entropija minimalna (ili ekvivalentno, informacijska snaga je maksimalna)
  3. Kreiraj čvor koji sadrži taj atribut

## ID3 (*Iterative Dichotomiser 3*) algoritam

---

- Algoritam je sličan prije navedenom u prezentaciji.
- Karakteristike algoritma: Temeljen je na Occamovoj britvi: preferira manja stabla odlučivanja u odnosu na veća (jednostavnije teorije). Dakako, ne stvara uvijek najmanja stabla, dakle algoritam je heuristički.
- Occamova britva se formalizira uporabom koncepta informacijske entropije:

$$I_E(i) = - \sum_{j=1}^m f(i,j) \log_2 f(i,j).$$

## C4.5 algoritam

---

- C4.5 je algoritam za generiranje stabala odlučivanja koji je razvio Ross Quinlan, kao proširenje algoritma ID3. Stabla odlučivanja koje generira algoritam C4.5 se može koristiti za klasifikaciju, pa ga se smatra statističkim klasifikatorom.
- C4.5 gradi stabla od skupa podataka za trening na isti način kao i ID3, pomoću koncepta informacijske entropije. Skup podataka je skup  $S = s_1, s_2, \dots$  već klasificiranih uzoraka. Svaki uzorak  $s_i = x_1, x_2, \dots$  je vektor gdje  $x_1, x_2, \dots$  prikazuje vrijednosti atributa ili svojstava određenog uzorka. Podaci za trening se dodaju s vektorom  $C = c_1, c_2, \dots$  gdje  $c_1, c_2, \dots$  prikazuje klasu kojoj svaki uzorak pripada.

## C4.5 algoritam

---

- Na svakom čvoru stabla, C4.5 izabire jedan atribut podataka koji najučinkovitije razdvaja skup podataka na podskupove kojima se pridodaju jednoj ili drugoj klasi. Kriterij je normalizirana vrijednost informacije (za razliku od entropije) što dolazi od izbora nekog atributa za razdvajanje podataka. Atribut s najvećom informacijskom vrijednošću se odabire da dodnosi odluku. C4.5 se tada ponavlja na manjim podlistama.
- Algoritam ima nekoliko temeljnih slučajeva.
  1. Svi uzorci u listi pripadaju istoj klasi. U tom slučaju, jednostavno se stvara stablo lista za stablo odlučivanja koje kaže da se odabere ta klasa.
  2. Nijedno od svojstava ne omogućava informacijsku vrijednost. U tom slučaju, C4.5 kreira viši čvor odlučivanja korištenjem očekivane vrijednosti klase.
  3. Pojavljuje se slučaj klase koja se prije nije susrela. C4.5 kreira viši čvor odlučivanje na stablu korištenjem očekivane vrijednosti.

## C4.5 algoritam

---

- Opis algoritma:

1. Provjeri za temeljne slučajeve
  2. Za svaki atribut  $a$ 
    - Izračunaj normaliziranu informacijsku vrijednost od razdvajanja kod  $a$
  3. Neka je  $x$  najbolji atribut s najvišom normaliziranim informacijskom vrijednošću
  4. Kreiraj čvor odlučivanja koji odvaja na atributu  $x$
  5. Ponavljaj na podlistama dobivenim kod odvajanja kod  $x$ , i dodajte te čvorove kao djecu čvora  $x$
-

## C4.5 algoritam, prethodnici i sljedbenici

---

- C4.5 poboljšava prijašnji algoritam ID3 na slijedeći način:
  - Mogu se koristiti i kontinuirani i diskretni atributi. Kod rada s kontinuiranim atributima, C4.5 kreira prag i tada razdvaja listu na kod kojih su vrijednosti atributa ispod prava i one kod koji su jednaki ili veći od praga.
  - Podaci kod kojih nedostaju vrijednosti jednostavno se označavaju s ?. Nedostajuće vrijednosti atributa se jednostavno ne koriste u računanju informacijske vrijednosti ili entropije
  - Računanje atributa s različitim težinama (cijenom).
  - Primjena tehnike kljaštrenja - C4.5 se vraća natrag po stablu koje je bio kreirao i nastoji odstraniti one grane koje ne pomažu tako da iz zamjenjuje s čvorovima listovima.
-

## C4.5 algoritam, prethodnici i sljedbenici

---

- Quinlan je kreirao algoritam C5.0 i See5 kao poboljšanje od C4.5 na slijedeći način:
    - Brzina - C5.0 je nekoliko redova veličina brži od C4.5
    - Učinkovitiji je s memorijom nego C4.5
    - C5.0 generira manja stabla odlučivanja
    - Tehnike poboljšavanja stabala, što ih čini pouzdanijima
    - Moguće je davati različite težine pojedinim atributim
    - Odvajanje korisnih podataka od smeća da se smanji šum.
  - Algoritam C5.0 je poslovna tajna.
-

# CART algoritam za učenje stabala

---

- Stablo odlučivanja je tehnika rudarenja podataka koja ima strukturu u obliku stabla kojom reprezentira skup odluka.
  - Generira se kroz seriju odluka o podjeli grupe promatranja baziranim na određenim varijablama.
- 
- Autori **algoritma CART (Classification And Regression Trees)** su bili 1984. godina Leo Breiman, Jerome Friedman, Richard Olshen i Charles Stone
  - CART gradi binarno stablo odlučivanja koje sadrži dvije grane za svaki čvor odlučivanja i nastavlja dijeljenje sve dok pronađe nove dijelove koji povećavaju sposobnost podjele podataka u kategorije.
-

# CART algoritam za učenje stabala

---

- CART razmatra kako napraviti svaku podjelu, odlučiti kada je čvor stabla vrh i kako pridijeliti predviđenu kategoriju svakom završnom čvoru.
- U skladu s uzorkom za treniranje određuju se kategorije :
  - ukoliko se greške pogrešnog klasificiranja smatraju jednakim lošim tada je pravilo dodjeljivanja kategorije većinsko pravilo - završnom čvoru je dodijeljena ona kategorija koja ima najveći broj članova u uzorku za treniranje
  - ukoliko se greške pogrešnog klasificiranja ne smatraju jednakim lošim razmatraju se troškovi pogrešne klasifikacije, a kategorija dodijeljena završnom čvoru treba biti ona koja minimizira trošak pogrešne klasifikacije za sve slučajeve uzorka za treniranje

# CART algoritam za učenje stabala

---

- ukoliko se jednakо kazne svи tipovi pogrešne klasifikacije, a uzorak za treniranje ima različitu proporciju članova kategorija od one u koju se očekuje klasifikacija budućih slučajeva, tada se koriste težine kako bi minimizirale broj pogrešne klasifikacije za buduće slučajeve
  - ukoliko svи tipovi pogrešne klasifikacije nisu jednakо kažnjeni i uzorak za treniranje ima različitu proporciju članova kategorija od one u koju se očekuje klasifikacija budućih slučajeva, tada se koriste dva skupa težina od kojih jedan računa razne troškove pogrešne klasifikacije, a drugi se koristi za prilagođavanje omjera kategorija budućih promatranja koja su različita od proporcija kategorija uzorka za treniranje.
-

## GINI index i entropija

---

- Kao funkciju procjene gdje će biti podjela koriste se funkcije nečistoće koje mogu biti Gini indeks (CART algoritam) i funkcija entropije (informacijska vrijednsot ID3, C4.5 i C5.0):
- Gini indeks koji se za dani čvor  $c$  može zapisati na sljedeći način:

$$Gini(c) = 1 - \sum_j p_j^2$$

- gdje je  $p_j$  vjerojatnost kategorije  $j$  u  $c$
- i funkcija entropije koja se može zapisati kao:

$$Entropija(c) = - \sum_j p_j \log p_j$$

Detaljinije na ploči.

---

# Prednosti stabala odlučivanja

---

- Prednosti stabala odlučivanja su:
    - jednostavni za razumijevanje i objašnjavanje
    - traže jednostavnu pripremu podataka
    - mogu raditi kako s numeričkim tako i kategorijskim vrijednostima
    - koriste model bijele kutije (sve je razumljivo što je u njoj, za razliku od crne kutije)
    - modeli se mogu vrednovati statističkim tehnikama
    - robustni
    - brzi su (dobro se odvijaju s velikom količinom podataka u kratkom vremenu)
-

# Nedostaci stabala odlučivanja

---

- Nedostaci stabala odlučivanja su:
    - algoritmi učenja stabala odlučivanja su temeljeni na heuristici, gdje se lokalno optimalne odluke donose u svakom čvoru, što ne jamči dobivanje globalno optimalnog stabla stabla (problem učenja nekog optimalnog stabla odlučivanja je poznat kao NP-kompletan)
    - problem pretreniranosti, jer se može kreirati presloženo stablo koji ne jamči da su podaci dovoljno dobro generalizirani
    - neke je koncepte vrlo teško naučiti (npr. funkcija XOR), i tada su prikazi vrlo veliki
-

# Usporedba nekoliko principa

---

- Usporedimo nekoliko različitih principa učenja iz podataka
  - Stabla odlučivanja
  - Linearna regresija
  - Neuronske mreže

# Linearna regresija

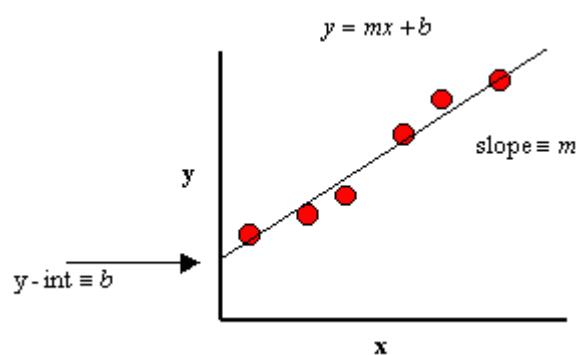
---

- Jednostavna teorija
  - Izvori znanja: podaci
  - Jednostavan i primjenjiv linearni model
-

# Linearna regresija

- Primjer podataka i njihova interpretacija preko linearne funkcije
- Prikazujemo podatke preko konstante  $m$  (nagib) and  $b$  (y-odsječak) u jednadžbi  $y = mx + b$

x	y
1.0	2.6
2.3	2.8
3.1	3.1
4.8	4.7
5.6	5.1
6.3	5.3



$$m = \frac{n\sum(xy) - \sum x \sum y}{n\sum(x^2) - (\sum x)^2}$$

$$b = \frac{\sum y - m \sum x}{n}$$

$$r = \sqrt{\frac{n\sum(xy) - \sum x \sum y}{\left[n\sum(x^2) - (\sum x)^2\right] \left[n\sum(y^2) - (\sum y)^2\right]}}$$

# Linearna regresija

---

- Traženje najjednostavnijeg pravca, koji aproksimira  $n$  točaka podataka  $\{y_i, x_i\}$ , gdje je  $i = 1, 2, \dots, n$ .

$$y = \alpha + \beta x,$$

Find  $\min_{\alpha, \beta} Q(\alpha, \beta)$ , where  $Q(\alpha, \beta) = \sum_{i=1}^n \hat{\varepsilon}_i^2 = \sum_{i=1}^n (y_i - \alpha - \beta x_i)^2$

$$\begin{aligned}\hat{\beta} &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{\sum_{i=1}^n x_i y_i - \sum_{i=1}^n x_i \sum_{j=1}^n y_j / n}{\sum_{i=1}^n (x_i^2) - (\sum_{i=1}^n x_i)^2 / n} \\ &= \frac{\bar{xy} - \bar{x}\bar{y}}{\bar{x^2} - \bar{x}^2} = \frac{\text{Cov}[x, y]}{\text{Var}[x]} = r_{xy} \frac{s_y}{s_x},\end{aligned}$$

$$\hat{\alpha} = \bar{y} - \hat{\beta} \bar{x},$$

# Linearna regresija

## □ Primjer u Excelu

		A11	▼	=	=COUNT(B3:B8)						
		A	B	C	D	E	F	G	H	I	J
1											
2		x	y	xy	$x^2$	$y^2$					
3		1.0	2.6	2.6	1.0	6.8					
4		2.3	2.8	6.44	5.3	7.8					
5		3.1	3.1	9.61	9.6	9.6					
6		4.8	4.7	22.56	23.0	22.09					
7		5.6	5.1	28.56	31.4	26.0					
8		6.3	5.3	33.39	39.7	28.1					
9											
10	n	$\Sigma x$	$\Sigma y$	$\Sigma (xy)$	$\Sigma (x^2)$	$\Sigma (y^2)$					
11	6	23.1	23.6	103.16	110.0	100.4					
12											
13		$(\Sigma x)^2$	$(\Sigma y)^2$								
14		533.61	556.96								
15											
16	slope, m =	0.5842									
17	y-int, b =	1.6842									
18	r =	0.9741									

$$=(A11*D11-B11*C11)/(A11*E11-B^2)$$

$$=(C11-C16*E11)/A11$$

$$=(A11*D11-B11*C11)/SQRT((A11*E11-B14)*(A11*E11-C14))$$

# Linearna regresija

---

- Primjer u Excelu, primjena gotovih funkcija
  - Slope, m: `=SLOPE(known_y's, known_x's)`
  - y-intercept, b: `=INTERCEPT(known_y's, known_x's)`
  - Correlation Coefficient, r: `=CORREL(known_y's, known_x's)`
  - R-squared, r<sup>2</sup>: `=RSQ(known_y's, known_x's)`

	A	B	C	D	E	F
1						
2	x	y		Slope, m, =	0.5842	
3	1.0	2.6		=SLOPE(C3:C8,B3:B8)		
4	2.3	2.8				
5	3.1	3.1		y-intercept, b, =	1.6842	
6	4.8	4.7		=INTERCEPT(C3:C8,B3:B8)		
7	5.6	5.1				
8	6.3	5.3		Correlation, r, =	0.9741	
9				=CORREL(C3:C8,B3:B8)		

# Zaključivanje temeljeno na slučajevima

---

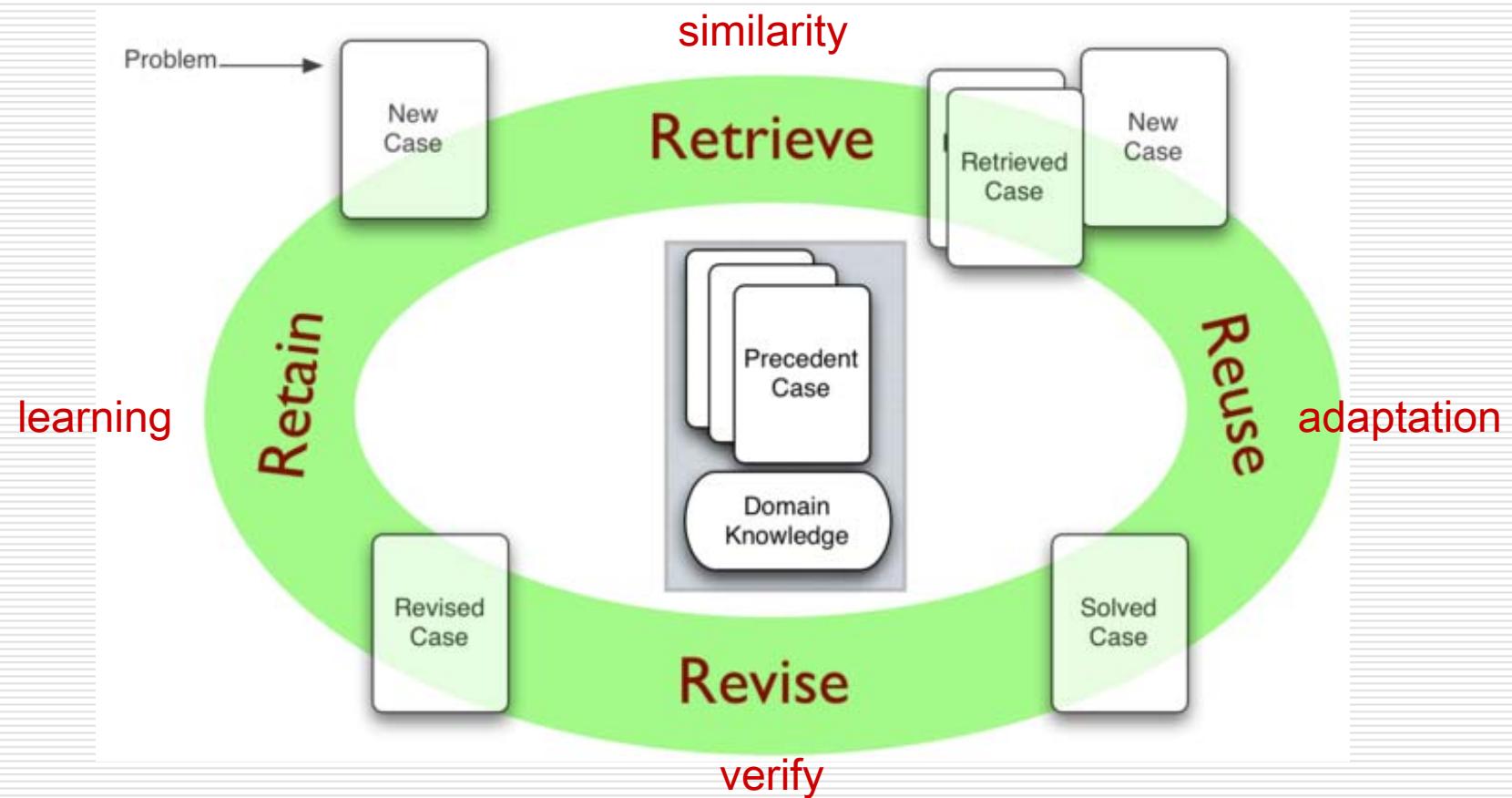
- Zaključivanje temeljeno na slučaju (Case-based reasoning, CBR) je paradigma za rješavanje problema koja se u svojim temeljima razlikuje od ostalih pristupa umjetne inteligencije.
1. Umjesto da se isključivo oslanja na opće znanje iz problemske domene ili izradi veza preko generaliziranih odnosa između deskriptora problema i zaključaka, CBR koristi specifično znanje situacija o kojima postoji prethodno iskustvo s određenim problemima (slučajevima, cases). Novi se problem rješava pronalaženjem rješenja sličnog problema u prošlosti i njegovim korištenjem u sličnoj problemskoj situaciji.
  2. Pristup CBR je inkrementalni poduprijeto učenje, budući da se novo iskustvo zadržava svaki put kada se riješi neki problem, i čini ga se trenutačno dostupnim za buduće probleme.

# Koraci zaključivanja temeljenog na slučajevima

---

- Koraci zaključivanja temeljenog na slučajevima su:
    1. Prikaz (Representation)
    2. Pretraživanje (Retrieval)
    3. Ponovna uporaba (Reuse)
    4. Revizija (Revision)
    5. Retencija (Retention)
-

# Zaključivanje temeljeno na slučajevima



## Koraci CBR

---

- Prikaz (Representation).** Sustavi CBR mogu prikazati najraznovrsniji raspon tipova podataka za sve tehnologije prikaza znanja. (Npr. neuronske mreže su prikladne samo za numeričke podatke, sustavi temeljeni na znanju numeričke i simboličke, CBR može obuhvataiti bilo koji tip podataka.) CBR može raditi sa slijedećim tipovima podataka: numerički, simbolički, kategoriski, hijerarhijski i tekstualni. Bilo koji tip podataka koji može provoditi uspoređivanje podudaranja jednog primjera s drugim se može koristiti (Npr. mnogi komercijalni alati rade samo podudaranje teksta.)
-

## Koraci CBR

---

- **Pretraživanje (Retrieval)** je proces traženja slučajeva u bazi slučajeva koji se najbolje podudara s poznatom trenutačnom situacijom. Trenutačna informacija se prikazuje kao novi slučaj s mnogo nedostajućih informacija. To je najbitniji korak u CBR metodi.
  - **Ponovna uporaba (Reuse)** je korak gdje se sustav slučajevi koji se podudaraju uspoređuju s novim slučajevima da oblikuju predloženo rješenje novog problema (slučaja).
-

## Koraci CBR

---

- Revizija (Revision)** je testiranje predloženog rješenja da se osigura da je predloženo rješenje prikladno i pouzdano. Rezultat je rješenje koje se potvrdili testiranjem.
  - Retencija (Retention)** je spremanje novih slučajeva za buduću uporabu. Najveća prednost CBR-a u odnosu na tehnologije zaključivanja (STZ ili KBS) je da se novo znanje kontinuirano i jednostavno dodaje da spremi iskustvo. To je tako jednostavno kao spremanje novog sloga u bazu.
-

# Zaključivanje temeljeno na slučajevima

---

- **Zaključivanje temeljeno na slučajevima** (Case Based Reasoning) je najprimjenjivije za probleme gdje ekspert treba objasniti kako riješiti problem davanjem primjera. Lakše je kada postoji baza prošlih slučajeva, ali se također može izvući iz samo nekoliko slučajeva.

# Zaključivanje temeljeno na slučajevima

---

- Područje gdje su se CBR pristupi pokazali uspješni su:
  - Call centri / Help Desks:** Kada agenti više razine s naprednim stručnim znanjem dodaju nove riješene slučajeve dodaju druge razine dodaju nove riješene slučajeve u bazu podataka, to stručno znanje odmah postaje dostupno manje iskusnim agentima. Mnogi se problemi rješavaju lakše na nižoj razini, jer postoji već spremljeno iskustvo.
  - Planiranje, rasporedi, sklapanje i dizajn:** Lista materijala i troškova se može generirati za novi projekt na temelju uspoređivanja sa specifikacijama prošlog projekta.
  - Dijagnoza:** skup simptoma se uspoređuje s postojećim primjerima (npr. kvarovi na strojevima, problemi s procesima u proizvodnji, medicinska dijagnostika).
  - Pravno zaključivanje.** U anglosaksonskom pravu zaključivanje se temelji na prošlim slučajevima. CBR koristi prošle slučajeve i njihovu ovisnost.
-

## Zaključivanje temeljeno na slučajevima

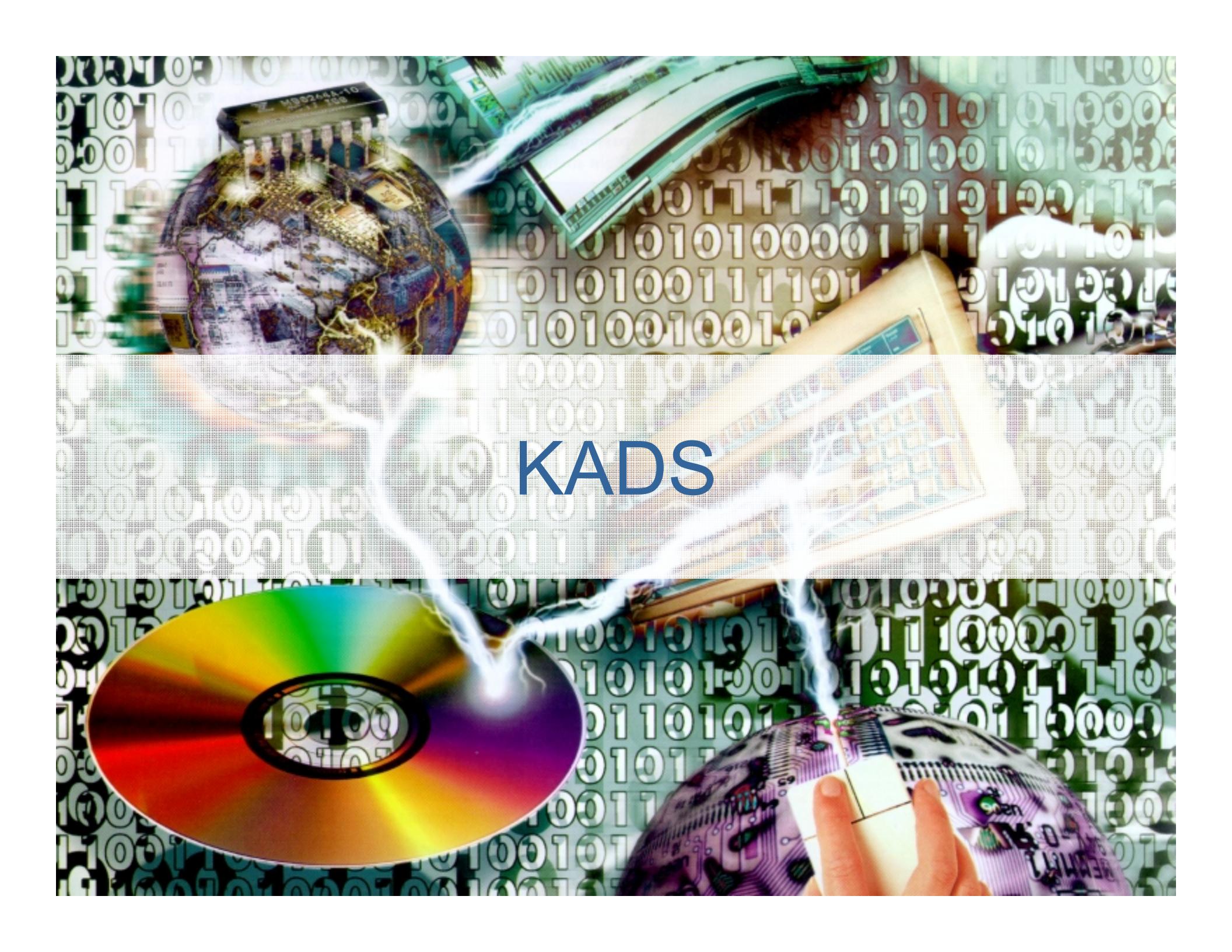
---

- Primjer: Savjeti za ribolov.
  - [http://bassexpert.com/basspro\\_demo.htm](http://bassexpert.com/basspro_demo.htm)
-

# INTELIGENTNI SUSTAVI

---

Prof. dr. sc. Božidar Kliček  
Sveučilište u Zagrebu  
Fakultet organizacije i informatike, Varaždin



# KADS

# METODOLOGIJA KADS



# Nastanak metodologije

---

- započela kao dio projekta ESPRIT 1983.
  - AI istraživači su pokazivali malo interesa za probleme metodologije
  - tada su bile uhodane tehnologije: brzo prototipiranje, LISP strojevi, Ijuske
  - ZAKLJUČAK: stukturni razvoj je jednako važan za ekspertne sustave kao i za klasične informacijske sisteme
-

# Temeljne postavke

---

- KADS je najopsežnija metodologija danas.
  - Razvoj ekspertnog sustava je aktivnost **modeliranja**, a ne samo punjenje sustava znanjem eksperata.
  - KBS je računalni model traženog ponašanja koji može odraziti aspekte ponašanja nekog eksperta.
  - KBS nije funkcionalni i behavioristički ekvivalent eksperta.**
-

# KADS

- 
- KADS je nastao 80 godina a označava Knowledge Acquisition Design Strategy
  - Iz nje se razvio CommonKADS, ali i niz drugih metodologija (npr. MIKE, Model Based Incremental Knowledge Engineering)
  - Ostale metodologije: DESIRE, PROTEGE-II, VITAL, TASK, RDR, GEMINI, RUDE, POLITE, PROforma, itd.
-

# Znanje u kontekstu

---

- Peter Drucker: Znanje je jedini značajan resurs danas. (U knjizi Post-Capitalist Society, 1993).
  - James Brian Quin: načinio je opsežnu studiju ključne uloge znanja u modernoj organizaciji u knjizi Intelligent Enterprise (1992).
  - Nova disciplina: inženjerstvo znanja.
-

# Znanje u kontekstu

---

- Podaci su neinterpretirani signali koji stalno stižu u naša osjetila.
  - Informacija je podataka opskrbljen značanjem.
  - Znanje je svrha i kompetencija data informaciji, a daje mogućnost za pokretanje akcija.
-

# Principi Common KADS-a

---

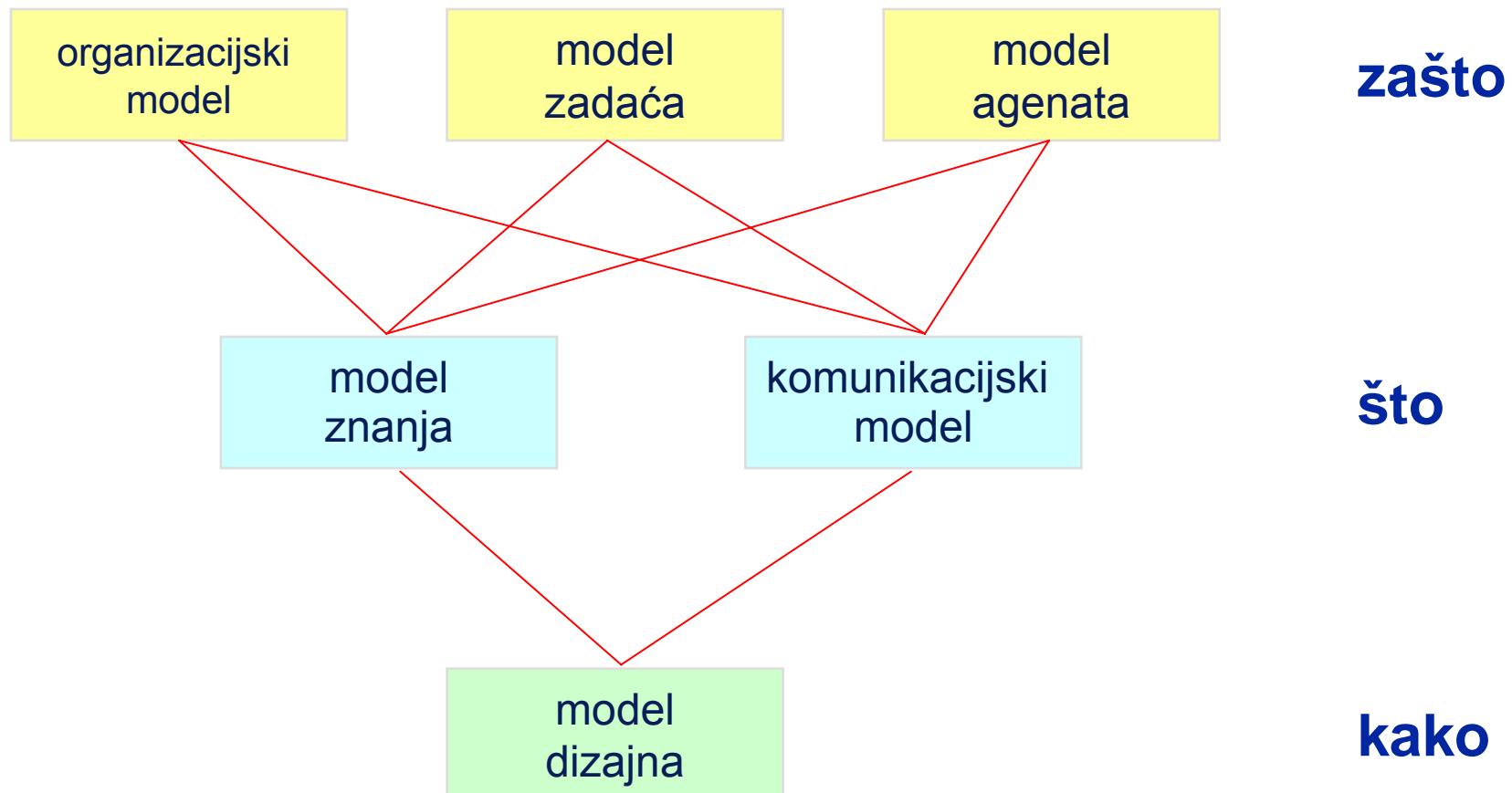
- Inženjerstvo znanja nije neki oblik izvlačenja znanja iz njegove glave, već se sastoji od konstrukcija različitih aspekta modela ljudskog znanja.
  - U modeliranju znanja najprije se treba koncentrirati na konceptualnoj strukturi znanja i ostaviti detalje programiranja za kasnije.
  - Znanje ima stabilnu internu strukturu koja se može analizirati preko dostiživih specifičnih tipova i uloga znanja.
  - Projektu znanja se moraju upravljati učenjem iz iskustva na kontroliranom spiralni način.
-

# Modeli KADS-a

- 
- Postoje tri grupe modela:
  - Zašto? Zašto je sustav temeljen na znanju rješenje. Koji su troškovi i organizacijski utjecaji? Organizacijski kontekst.
  - Što? Priroda i struktura znanja.
  - Kako? Kako se znanja implementira i s kojim arhitekturom. Tehnički aspekti problema.
-

# Modeli KADS-a

- 
- model stručnog znanja
  - model organizacije
  - model zadaća
  - model agenata
  - komunikacijski model
  - model dizajna
-



**zašto**

**što**

**kako**

# Organizacijski model

---

- Organizacijski model podržava analizu glavnih svojstava organizacije, u svrhu otkrivanja problema i mogućnosti primjene ove tehnologije, ostvarivanje svojstava i prosuditi utjecaj organizacije na sustav koji se planira graditi.
-

# Model zadaća

---

- Poslovni proces sastavljen je od zadaća. Ovaj model analizira opći pregled zadaća u organizaciji, ulaze i izlaze, preuvjete i kriterije, kao i resurse te kompetenciju.
-

# Model agenata

- 
- Agenti su izvršitelji zadaća. Mogu biti ljudi ili informacijski sustavi, odnosni bilo koji entiteti koji mogu izvršiti neku zadaću. Ovaj model opisuje karakteristike agenata, kompetencije, nadležnost za djelovanje, te ograničenja. Također daju se komunikacijske veze između agenata.
-

# Model znanja

---

- U detalje eksplicira vrstu i strukturu znaja koje se koristi za izvršavanje zadaće. To je opis koji ne ovisi o tehnologiji implementacije, već je na jednom općem nivou razumljivom za ljude.
  - Na taj način mogu komunicirati s ekspertima i korisnicima.
-

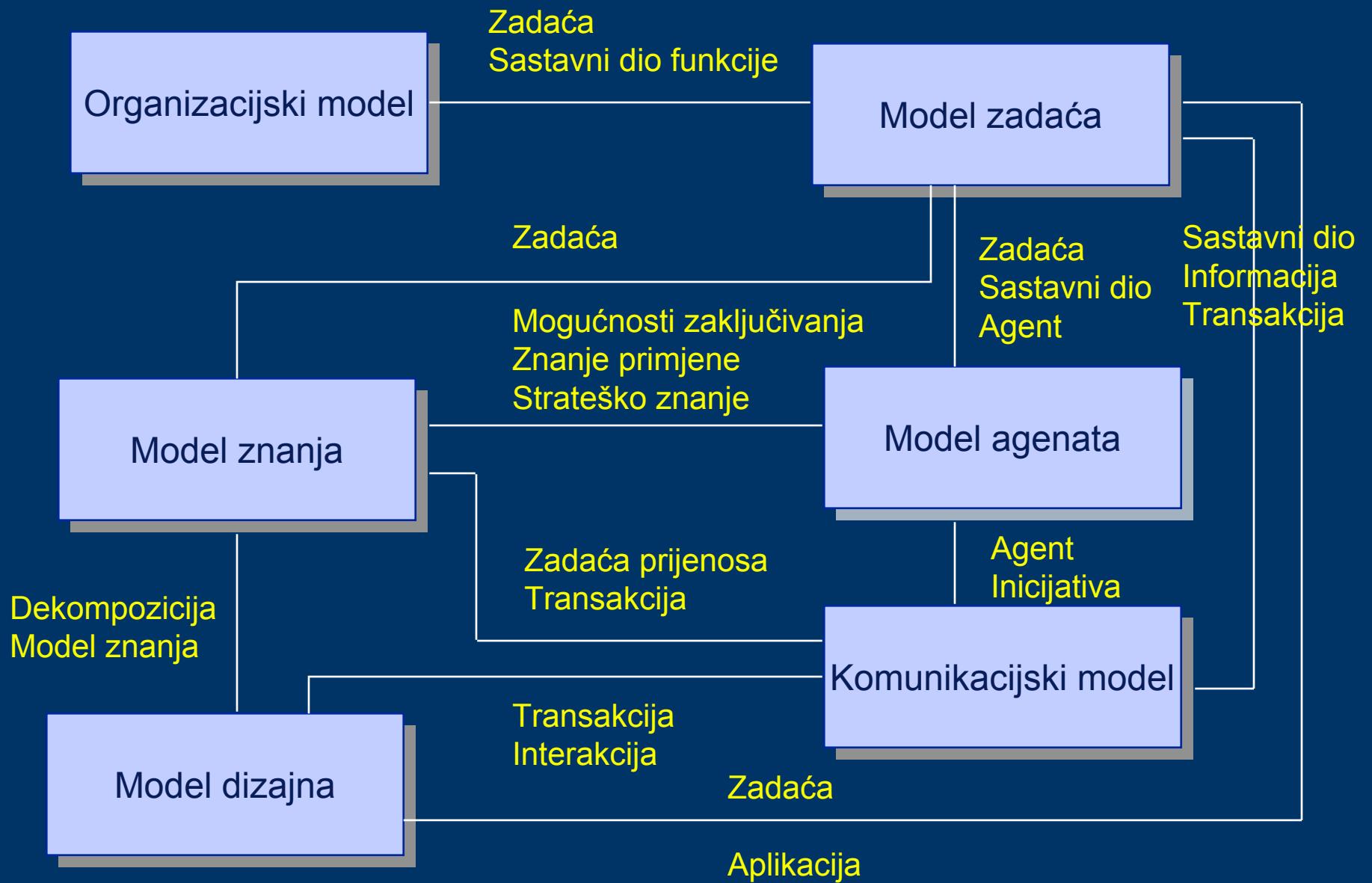
# Komunikacijski model

---

- Opisuju komunikaciju između agenata. Na konceptualnoj razini neovisnoj o implementaciji.
-

# Model dizajna

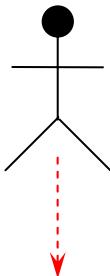
- 
- Model kojeg gradi Common KADS su ustvari specifikacije zahtjeva za sustav temeljen na znanju. Ovaj model dizajna daje tehničke specifikacije u terminima arhitekture, platformu implementacije, softverske module, konstrukte prikaza, računalne mehanizme potrebe za implementacije funkcija u modelima znanja i komunikacije.
-



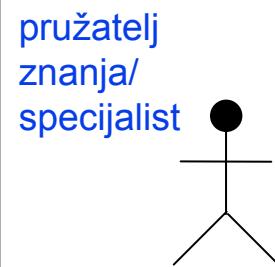
# Principi upravljanja projektom

---

- aktivnosti upravljanja projektom i razvoja su odvojene
  - upravljanje projektom je predstavljeno preko modela aktivnosti upravljanja koji komunicira s radom razvoja kroz stanja modela pridružena modelima KADS-a
  - razvojni proces je ciklički (kao Boehmov spiralni model)
-



menađer znanja  
definira strategiju znanja  
započinje projekt  
pospješuje distribuciju znanja



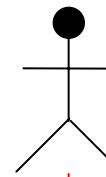
pružatelj  
znanja/  
specijalist

prikuplja znanje od

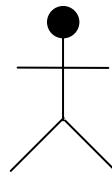
vrednuje

inženjer znanja

upravlja



voditelj  
projekta

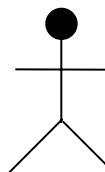


korisnik znanja

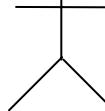
koristi

Sustav temeljen  
na znanju

dizajnira i implementira



dostavlja  
modele analize

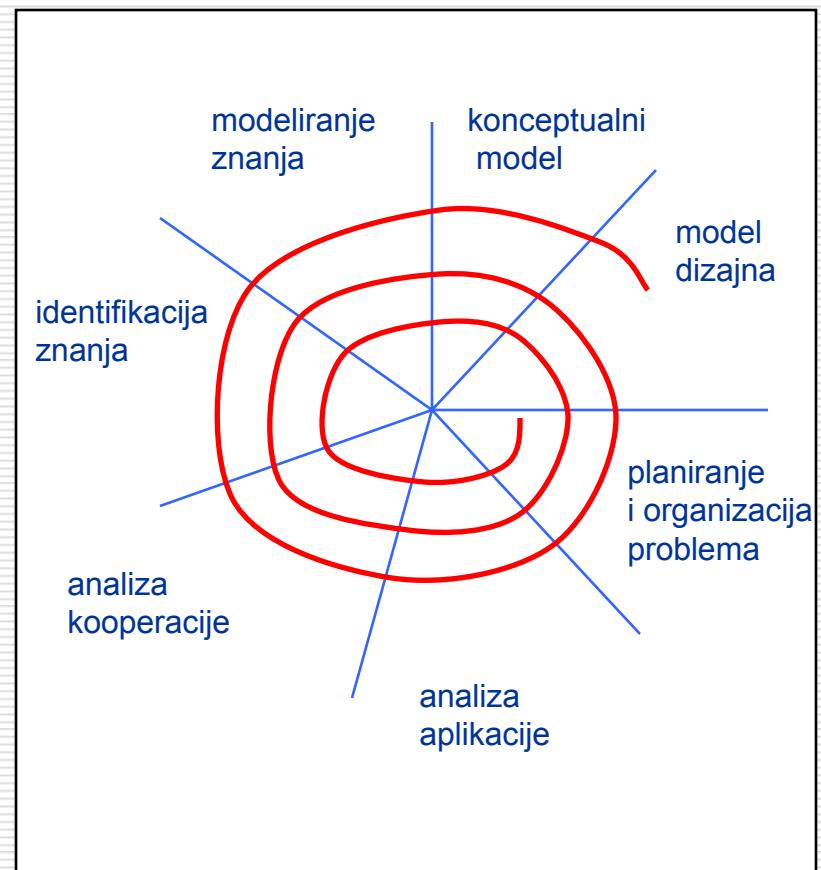


dizajner sustava

Uloge u procesu upravljanja i inženjerstvu znanja

# Spiralni životni ciklus kod KADS-a

- planiranje i organizacija problema
- analiza aplikacije
- analiza kooperacije
- identifikacija znanja
- modeliranje znanja
- konceptualni model
- model dizajna



# Upravljanje projektom

## Odredite ciljeve

- Shvatite tekuću situaciju

## Identificirajte rizike

- Neslaganje sa strategijom tvrtke
- Opis problema je nepotpun

## Pregledajte ciljeve

## Definirajte stanja ciljnog modela

OM: opis problema = vrednovan

## Aktivnosti razvoja plana

## Razvoj

opis tekućeg modela → opis novog modela

## Kontrola kvalitete

OM: problem = vrednovan

OM: problem = opisan

OM: proces = opisan

OM: struktura = opisana

TM: vr. opterećenje = opisano

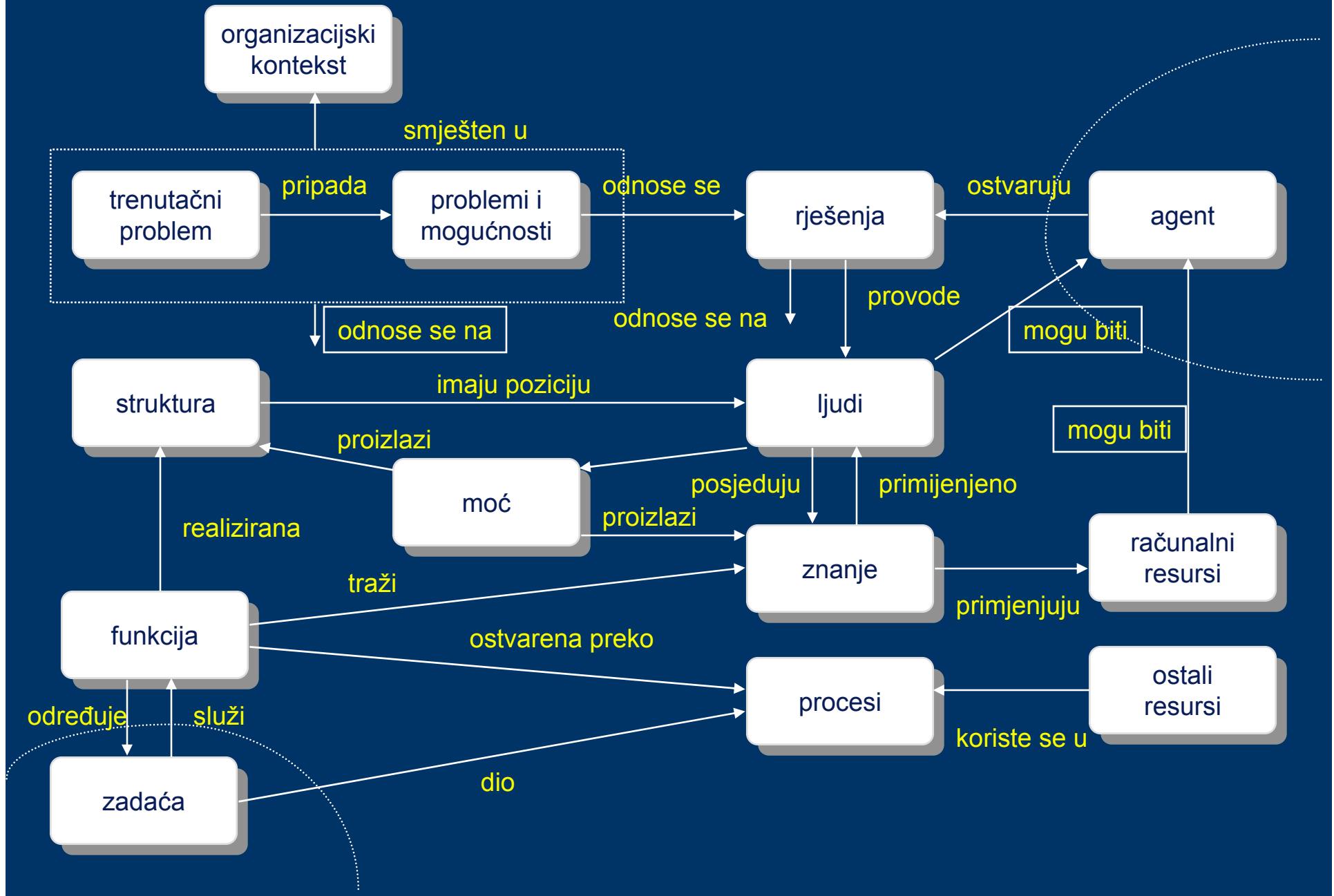
TM: dekompozicija = opisana

OM = organizacijski model, TM = model zadaća

# Modeli KADS-a

- 
- Dekompozicija procesa na različite modele je način bavljenja s kompleksnočcu ekspertnog sustava.
  - Model je apstrakcija koja je koncentrirana na određena ključna svojstva područja koja se modelira.
-

# Predlošci organizacijskog modela



# Modeli KADS-a

---

## Organizacijski model

- prvi korak u CommonKADS-u
  - ciljevi su otkriti :
    - probleme,
    - mogućnosti i
    - moguće utjecaje razvoja KBS sustava.
-

# Modeli KADS-a

---

## Organizacijski model

- socio-organizacijska analiza okoline
  - model visoke razine funkcija, zadaća i problema u organizacijama
  - prosudba učinaka uvođenja ekspertnih sustava u organizaciju
-

# Modeli KADS-a

## Aplikacijski model

- područja aplikacije
- probleme koje sustav treba riješiti
- cjelokupna funkcija i svrha sustava.

# Modeli KADS-a

---

## Model zadaća

- dekompozicija sustava na seriju zadaća i podzadaća koje sustav treba ostvariti
  - zadaća je definirana s ulazom i rezultatima (ulazno-izlazne specifikacije)
  - zadaće se pridjeljuju agentima (ekspertni sustavi, korisnici itd.)
  - ograničenja i specifični zahtjevi za zadaće.
-

# Modeli KADS-a

---

## Model kooperacije

- Model kooperacije - kako sustav i korisnik rade zajedno na razini zadaća i vrše interakciju u različitim režimima rada.
  - složeni skup sučelja i interakcija
  - znatno složenije nego za tradicionalnu informacijsku tehnologiju.
-

# Modeli KADS-a

---

## Model stručnog znanja

- funkcionalna specifikacija rješavanja problema ekspertnog sustava
  - daje ponašanje i tipove znanja koji su uključeni
-

# Modeli KADS-a

---

## Model stručnog znanja

- proces identifikacije znanja se provodi prije stvaranja modela stručnog znanja
  - istraživanje područja prije izgradnje modela
  - prikupljaju se podaci prije gradnje modela.
-

# Modeli KADS-a

---

## Model stručnog znanja

- Podaci i zadaće se dobivaju različitim tehnikama (strukturirani intervjuji, racionalna analiza zadaća, analiza tijeka rada i analiza pohrane)
  - rječnici i leksikoni, dokumentiranje područja
-

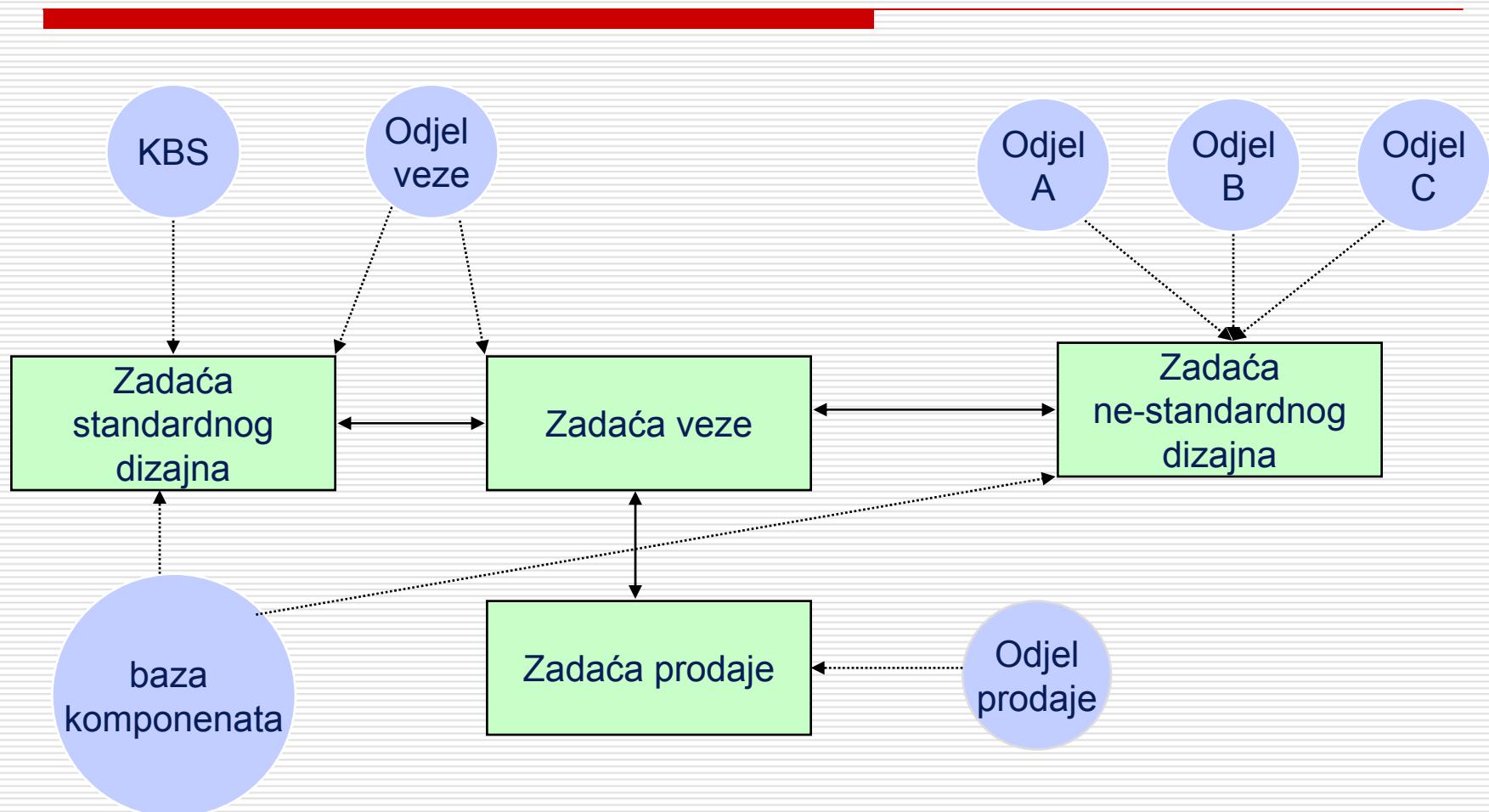
# Modeli KADS-a

---

## Model stručnog znanja

- modeliranje znanja u četiri razine
  - (dijagrami stabla, stabla odlučivanja, Ijestve, dijagrami modeliranja podataka, protokoli glasnog razmišljanja i simulacija scenarija)
-

# Primjer modela zadaća



# Vrste znanja

---

- Znanje domene
  - Znanje zaključivanja
  - Znanje zadaća
  - Strateško znanje
-

# Vrste znanja

---

## Znanje domene

- identifikacija koncepata, svojstava, odnosa struktura
  - koncepti su stvari ili objekti u domeni znanja
  - svojstva su atributi koncepata
  - strukture skupine složenih objekata.
-

```
C1: leftplatformedge_lefthoistwaywall =
    openinghoistwayleftspec - carreturnleft

C2: leftplatformedge_lefthoistwaywall >= 8

C3: counterwtovertravel =
    toplandtobeam - (deflsheavep - counterwtfoot + 
    counterwtbuffblockh + counterwtbuffheight +
    counterwtrunby + (counterwtframeh - pitdept))

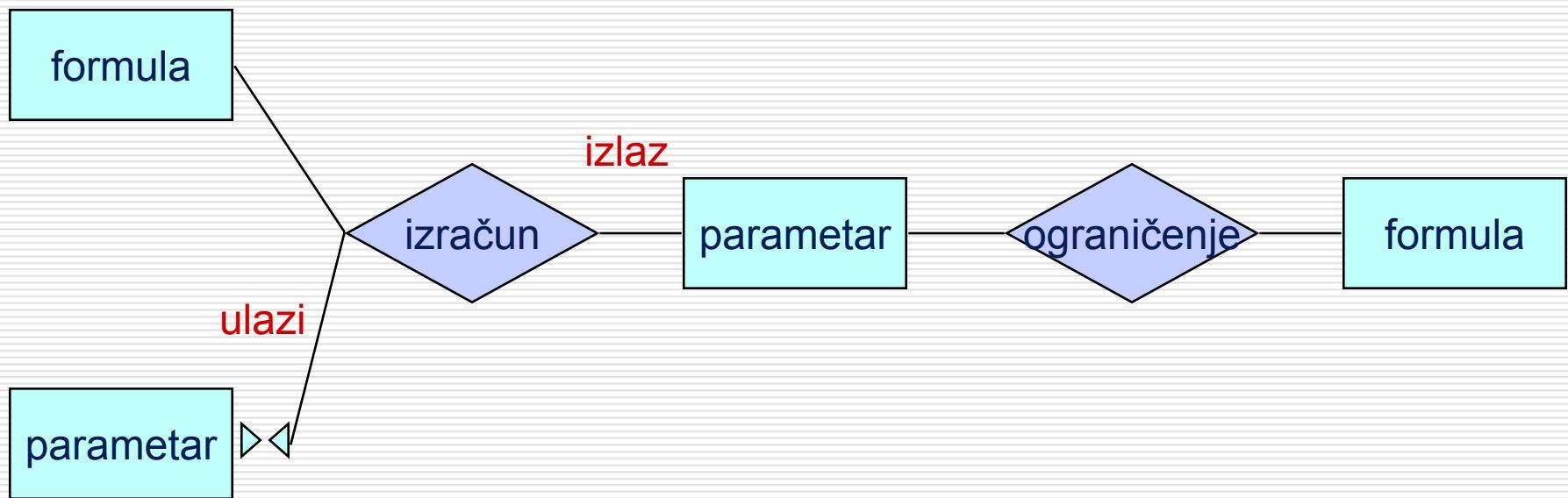
C4: counterwtovertravel >= carrunby + (1.5 *
    (carbufferstroke +6))

C5: IF machine-model is-one-of {"machine28", "machine38"}
    THEN 3 >= noohoistcables >= 6

C6: cwt_to_hoistway_rear >= cwt_ubracket_protrusion + 0.75

C7: IF machine-model = "machine18" AND elevatorspeed = 200
    THEN machine_efficiency = 0.78
```

# Struktura znanja



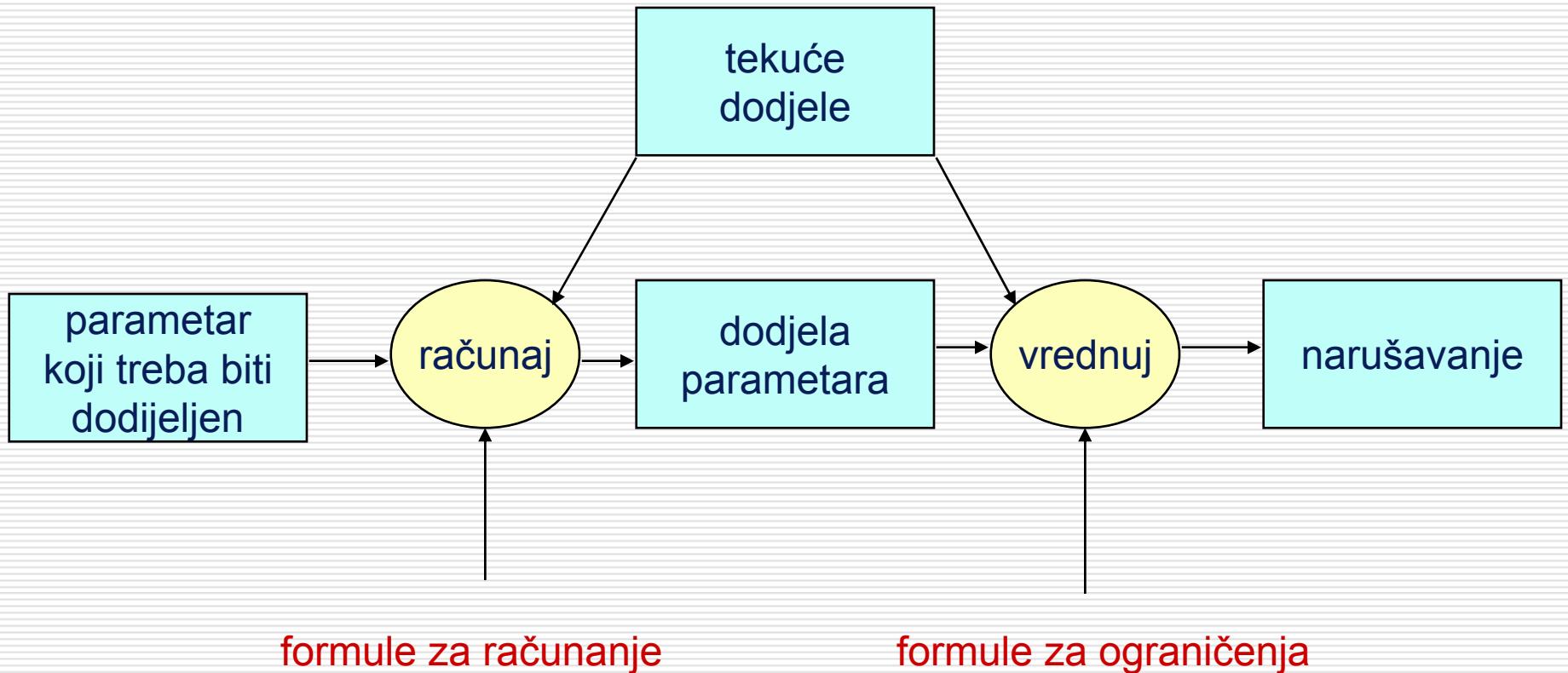
# Vrste znanja

---

## Znanje zaključivanja

- koristi znanje domene da zaključi ili stvori novu informaciju
  - odvajanje znanja domene od znanja zaključivanja
  - prikaz na što općenitijoj razini da se može koristiti i drugdje
  - ostvarivanje ponovne uporabivosti znanja.
-

# Primjeri zaključivanja



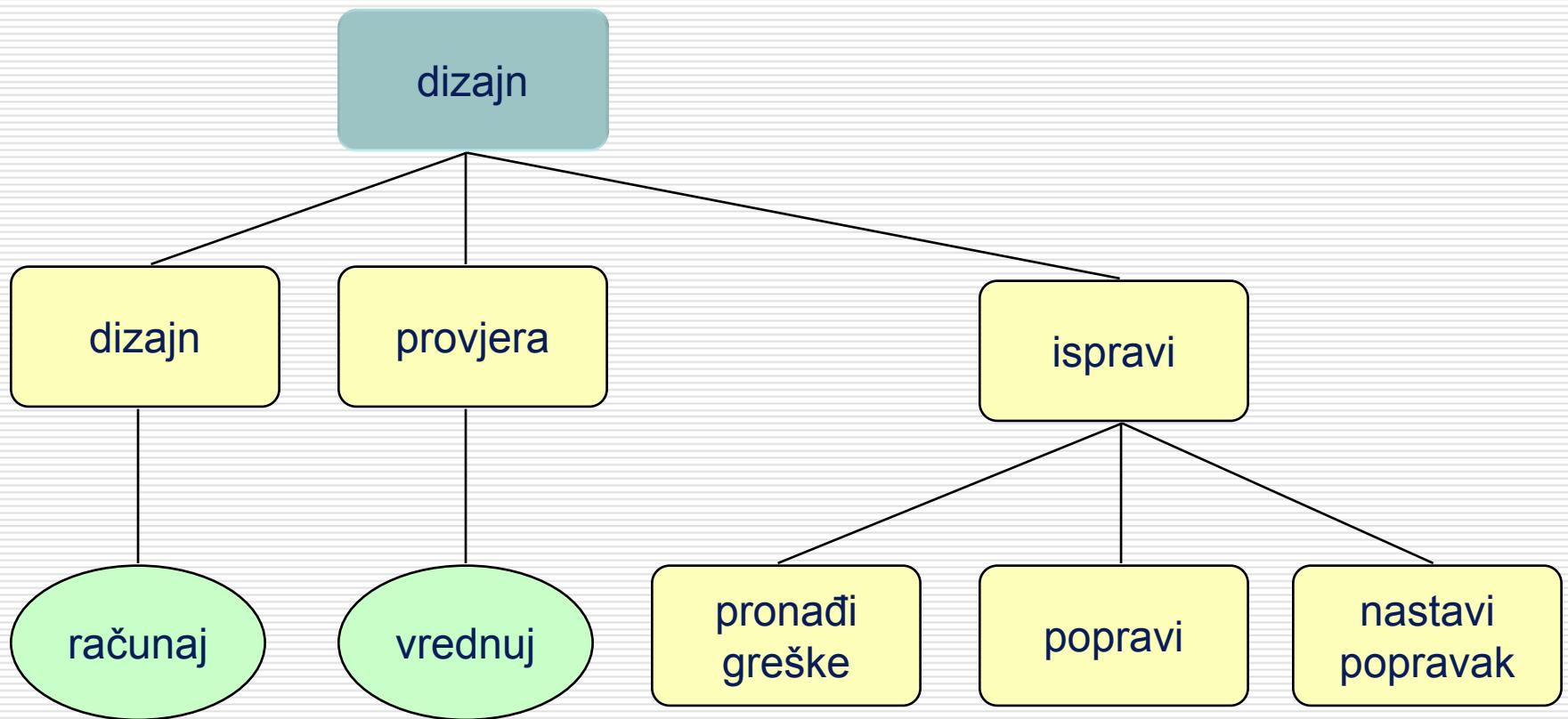
# Vrste znanja

---

## Znanje zadaća

- kako se izvori elementarnog znanja koriste za ostvarivanje pojedinih ciljeva više razine (zadaće)
  - zadaće se dekomponiraju u podzadaće - zadaće zaključivanja, rješavanja problema ili zadaće transfera (s vanjskim agentom)
  - struktura zadaća i podzadaća je opisana pomoću strukturiranog jezika
-

# Dekompozicija zadaća



# Vrste znanja

---

## Strateško znanje

- identificira strategije, ciljeve, pravila visoke razine i zadaće koje su značajne za rješavanje pojedinog problema.
-

**task** parametric-design

**task-definition**

**goal:** “find a design that satisfies a set of constraints”;

**input:**

skeletal design: “the set of system parameters to which values need to be assigned”;

requirements: “the set of initial parameter/value pairs”;

**output:**

design: “final set of assigned parameters”;

**task-body**

**type:** composite;

**sub-tasks:** init, propose, verify, revise;

**additional-roles:**

extended-design: “current set of assigned parameters”;

design-extension: “proposed new element of the extended module”;

violation: “violated constraint”;

**control-structure:**

parametric design(skeletal design + requirements → design) =

init(requirements → extended-design)

REPEAT

propose(skeletal-design + extended-design → design-extension)

extended-design := design-extension ∪ extended-design

verify(design-extension + extended-design → violation)

IF “some violation”

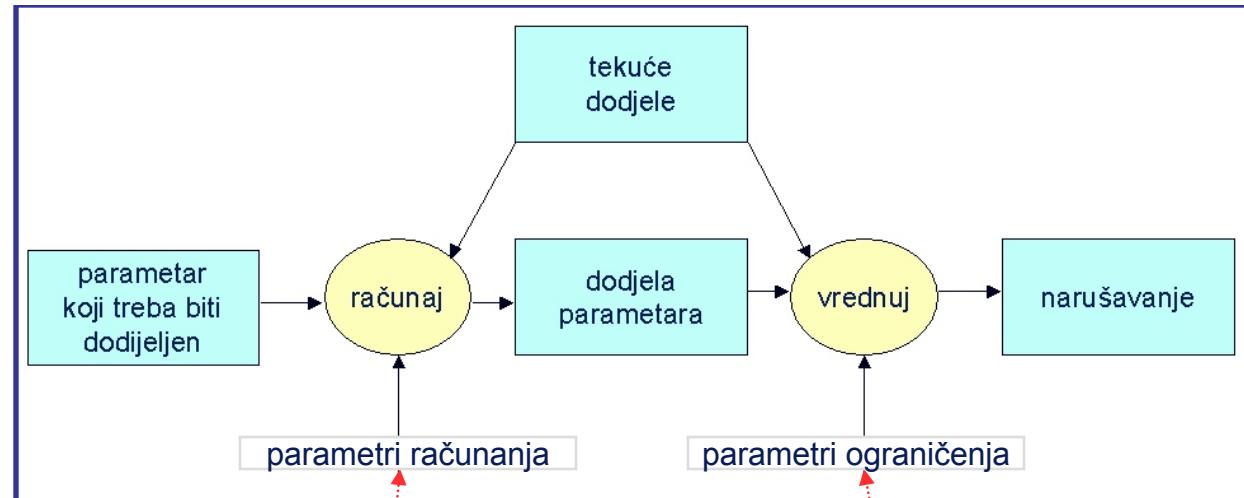
THEN revise(extended-design + violation → extended-design)

UNTIL “*a value has been assigned to all parameters in the skeletal-design*”

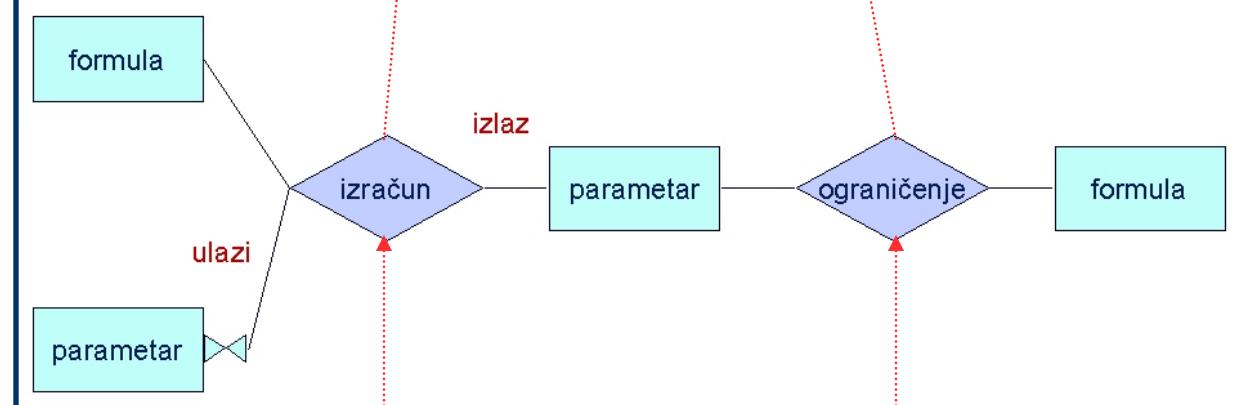
design := extended-design;

**end**

## Struktura zaključivanja



## P&R ontologija



## Ontologija parametarskog dizajna



## Baza znanja domene

C1:  
leftplatformedge\_lefthoistwaywall  
=  
openinghoistwayleftrightspec  
- carreturnleft

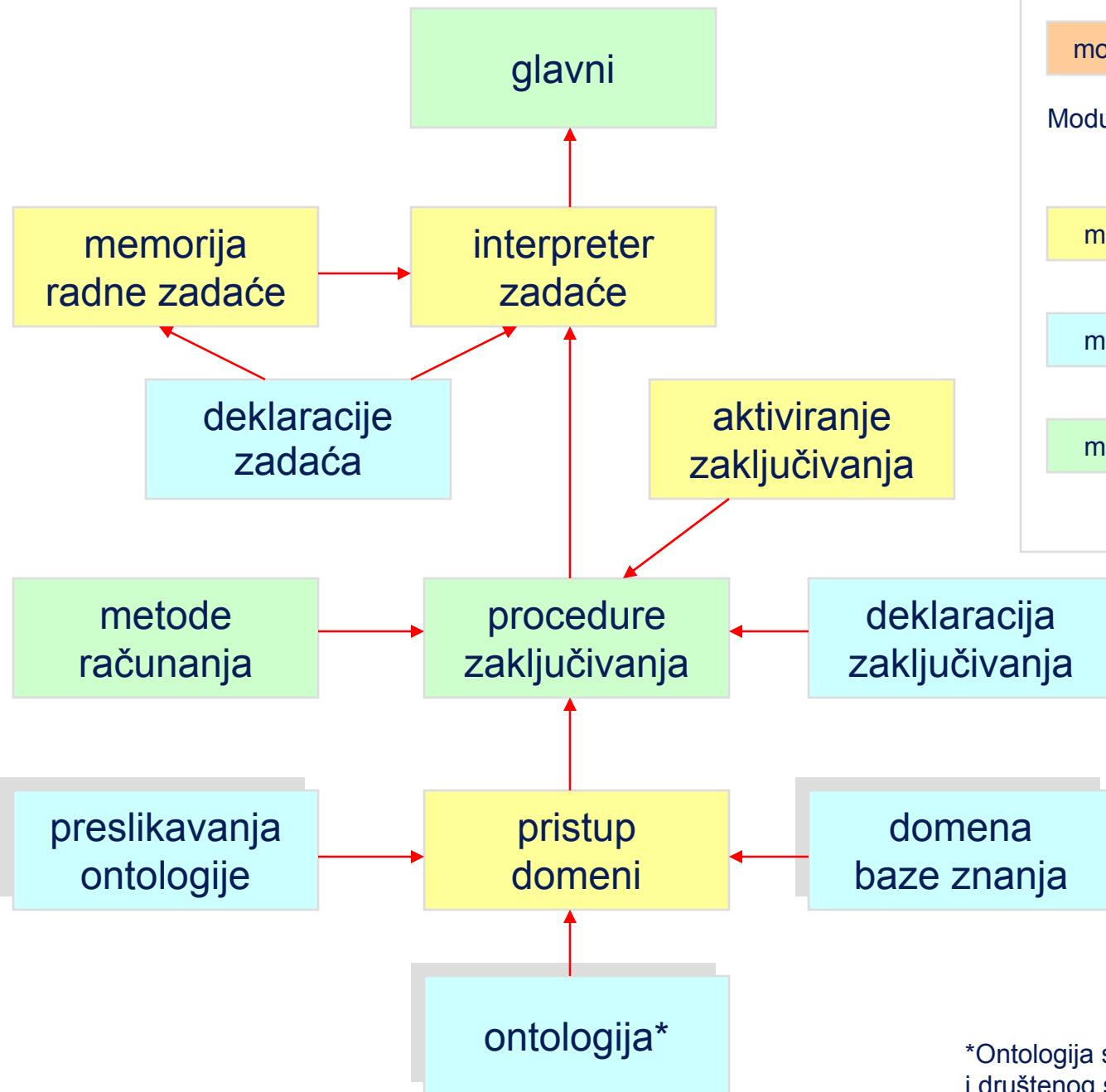
C2:  
leftplatformedge\_lefthoistwaywall  
>= 8

# Modeli KADS-a

---

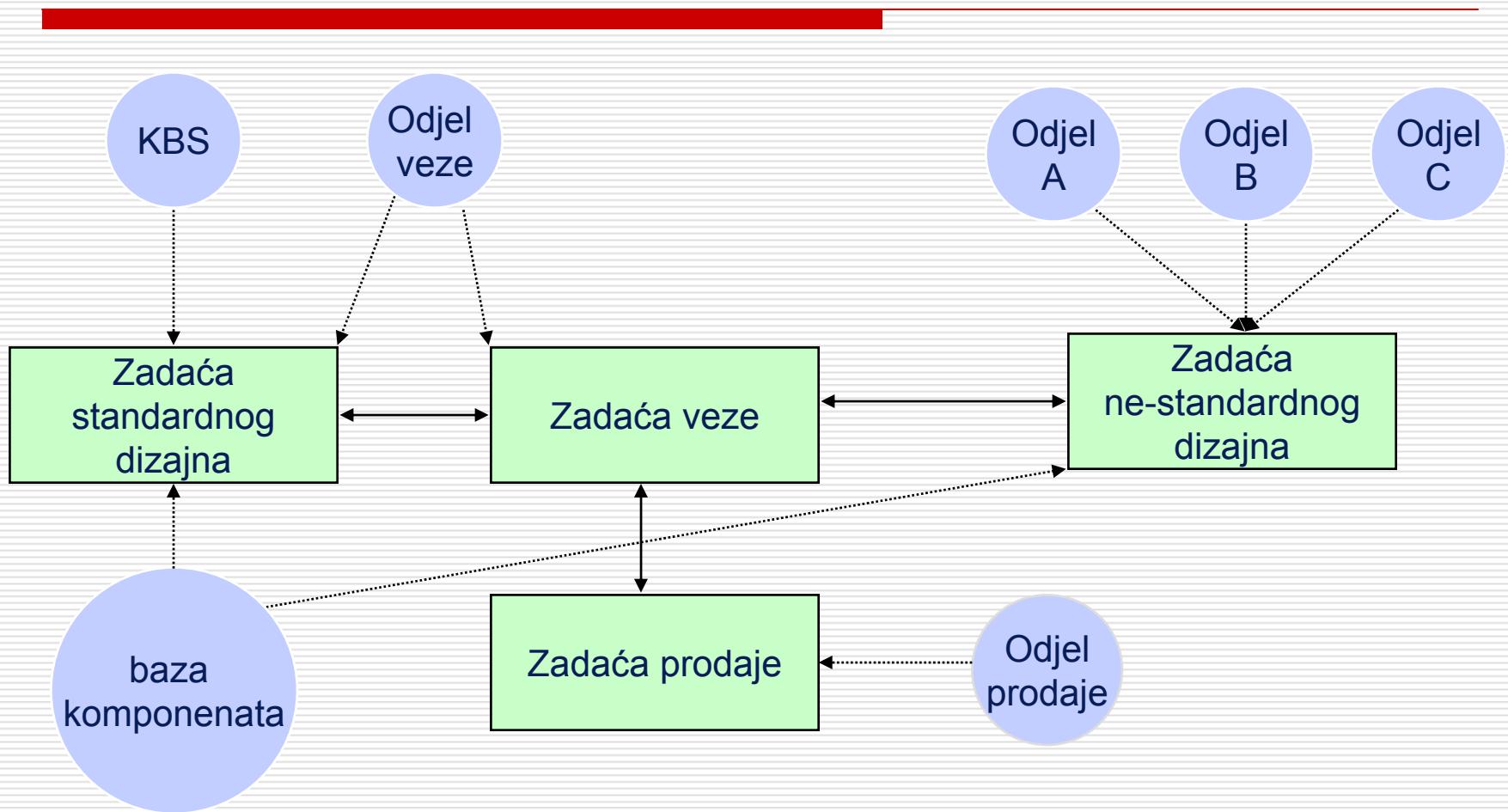
## Model dizajna

- tehnika računanja i prikaza u konceptualnom modelu
  - ograničenja strojne i programske opreme i vanjskog svijeta
  - slično je odvajanju logičkog i fizičkog modela kao u informacijskim sustavima
  - proces nije automatski niti dobro definiran
-



\*Ontologija se bavi temeljnom prirodom fizičkog i društvenog svijeta i načinu na koji oni djeluju.

# Arhitektura sustava



# Perspektiva

---

- modeliranje omogućuje podjelu svijeta u upravljive dijelove i izbor pogleda koji su važni za rizike u razvojnem procesu
  - predlošci modela su srž metodologije
  - uvođenje ontologije kao mehanizma apstraktnog opisa strukture i rječnika u domeni znanja
  - povezivanje znanja domene i zaključivanja
-

# Perspektiva

---

## sličnost

KADS-a i programskog inženjerstva: pogled podataka, funkcija i upravljanja

## razlike

izražajnost modeliranja znanja je veća od tehnika ER ili OR

struktura zaključivanja je vrlo ograničavajuća u KADS-u

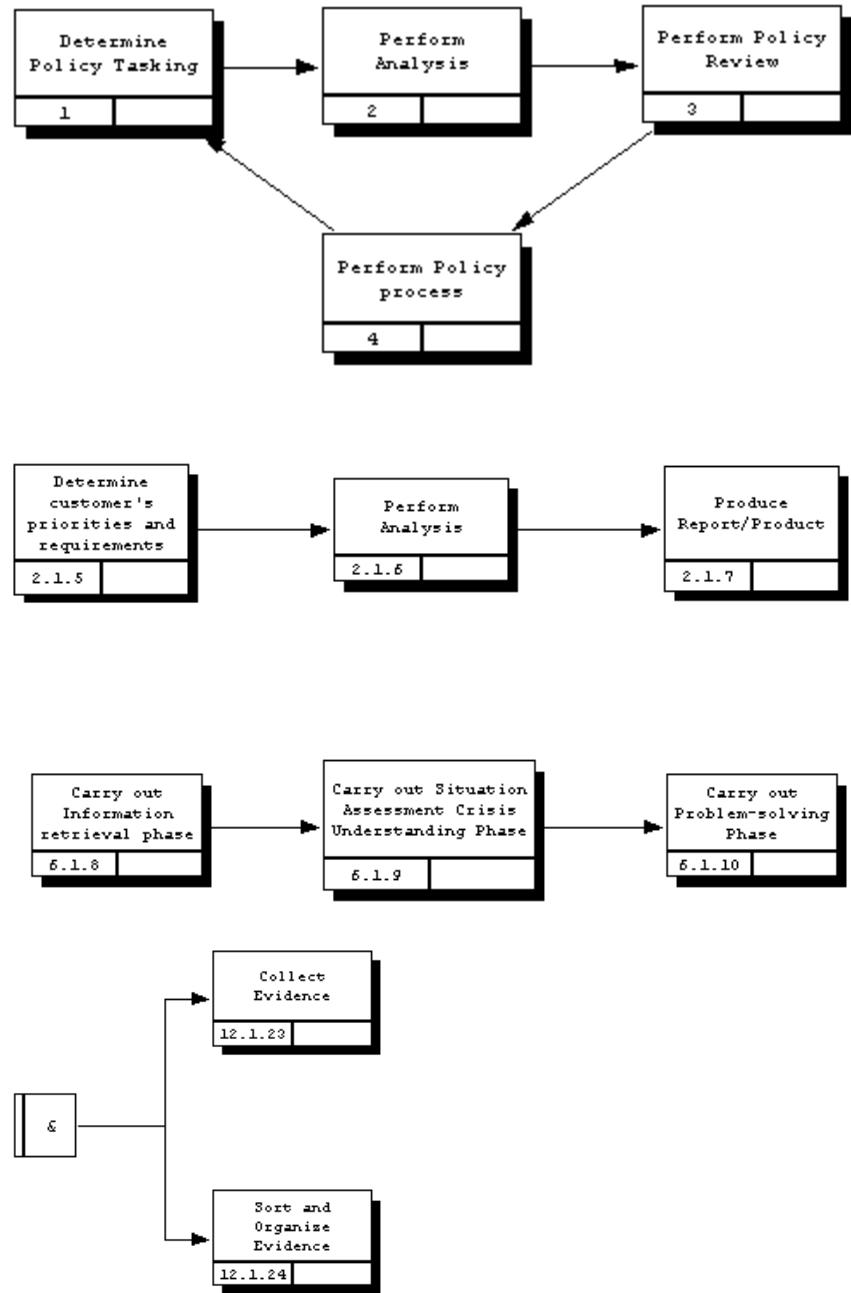
indirektna priroda veze između znanja i zaključivanja

---

# Contents

- Top Level

- Analysis
  - Perform Analysis
    - Information retrieval phase
      - Develop Problem Structure
        - Bound Problem
        - Establish Criteria for Information Gathering
      - Gather Information
        - Collect Evidence
    - Situation Assessment Crisis Understanding Phase
      - Develop understanding of problem
        - Determine context
          - Determine domestic context
          - Determine Political/ Social/ Economic Factors
          - Determine regional/global context
        - Assess problem
        - Evaluate evidence
      - Assess implications
        - Determine Other Impacts
    - Problem-solving Phase
      - Generate Action Options
      - Evaluate Action Options
        - Determine Evaluation Criteria
        - Analyze options using appropriate methodology
      - Draw Conclusions



# Alati

---

- Izbor ovisi o zahtjevima se organizacije u kojoj će se koristiti.
- Slijedeći faktori se uzimaju u obzir:
  - Da li je važnije koristiti CommonKADS ili KADS metodologiju.
  - Da li će se razvijati svi CommonKADS modeli ili samo model znanja.
  - Da li je podrška za implementaciju važna kao i za analizu i dizajn.

# Alati

---

## KADSTOOL

+

- pruža najbolje mogućnosti u grafičkom prikazu modela,
- podržava mogućnosti modela znanja koje su predstavljene u CommonKADS-u.

-

- ne podržava razinu modela ontologije u domenskom znanju, kao ni jedan drugi model u CommonKADS-u.

# Alati

---

- OpenKADS

+

- podržava podjelu domenskih rječnika u direktorije
- više mogućnosti eksplicitno namjenjenih za podršku više korisnika.

-

- pošto se zasniva na originalnoj KADS metologiji, nedostaje mu podrška za brojne mogućnosti uvedene u CommonKADS-u.
-

# Alati

---

- CommonKADS Workbench
    - +
    - podržava sve CommonKADS modele
    - pruža podršku za upravljanje projektima CommonKADS-a
  - 
  - konačna verzija kasni za ostalim alatima
-

# Alati

---

- VITAL workbench

+

- pruža dobre mogućnosti za prikupljanje znanja, te podršku dizajnu i implementaciji sa vizualizacijom uključenih procesa
- pruža podršku za upravljanje projektom

-

- bazira se na drugoj metodologiji i ne daje potporu mnogim mogućnostima CommonKADS-a
  - hoće li ikad postati komercijalno dostupan?
-

# Alati

---

- PC-PACK**
  - integrirani skup alata baze znanja koji podržava prikupljanje, analizu i vrednovanje znanja.
  - Pruža sedam vrsta alata: "Protocol Tool", "Ladder Tool", "Diagram Tool", "Matrix Tool", "Annotation Tool", "Publisher Tool" i "Diagram Template Tool "
-

# UML

---

- CommonKADS se temelji na objektno-orientiranom razvoju i za izradu modela koristi UML notaciju.
- UML je grafički jezik za vizualizaciju, specifikaciju, konstruiranje i dokumentiranje komponenti programskog sustava, kao i za poslovno modeliranje i druge neprogramske sustave.
- Počeli su ga razvijati Grady Booch i Jim Rumbaugh 1994 godine, kombinirajući notacije iz svojih dviju poznatih metoda:
  - „The Booch i
  - OMT“ (Object Modeling Technique) metoda.

# UML

---

- UML je „jezik“ za specifikaciju, a NIJE metoda ili procedura.
  - Koristi se da definira softverski sustav, za detaljizaciju konstrukata u sustavu, za dokumentaciju i izgradnju.
  - Može se koristiti na različite načine za podršku metodologija za razvoj softvera, ali sam po sebi on ne specificira ni metodologiju, ni proces.
-

# UML

---

- Tri su glavna elementa jezika:
    - građevni blokovi (*Building Blocks*),
    - pravila (*Rules*) koja diktiraju kako se ti građevni blokovi mogu staviti zajedno,
    - neki uobičajni mehanizmi podržani UML-om (*Common Mechanisms*).
-

# UML

---

- Tri su klasifikacije UML dijagrama:
    - Dijagrami ponašanja
    - Dijagrami interakcije
    - Dijagrami strukture.
-

# UML

---

- *Dijagrami ponašanja* predstavljaju tip dijagrama koji opisuje ponašanje sustava ili poslovnog procesa.
  - Uključuju:
    - dijagram aktivnosti,
    - dijagram stanja stroja i dijagram slučajeva korištenja
    - dijagrame interakcije.
-

# UML

---

- *Dijagrami interakcije* predstavljaju podskup dijagrama ponašanja, a naglasak je stavljen na interakcije.
- Uključuju:
  - dijagram komunikacije,
  - dijagram pregleda interakcija,
  - dijagram toka i
  - dijagram vremena.

# UML

---

- Dijagrami strukture predstavljaju tip dijagrama na kojima su prikazani elementi strukture koji su neovisni o vremenu.
- Uključuju:
  - dijagram klase,
  - dijagram kompozitne strukture,
  - dijagram komponenata,
  - dijagram uvođenja,
  - dijagram objekata i
  - dijagram paketa.

# UML

---

- CommonKADS koristi slijedeće UML dijagrame:
    - dijagram aktivnosti,
    - dijagram klasa,
    - dijagram slučajeva korištenja.
  - Svojstvo „proširivosti“ čini UML najviše favoriziranom tehnikom u modeliranju znanja!
-

# INTELIGENTNI SUSTAVI

---

Prof. dr. sc. Božidar Kliček  
Sveučilište u Zagrebu  
Fakultet organizacije i informatike, Varaždin

# Inteligentni agenti

# INTELIGENTNI AGENTI



# Uvod

---

- 60-ih godina prošlog stoljeća, unutar umjetne inteligencije, počinje razvoj *intelligentnih sustava*.
  - U novije vrijeme pojavljuje se paradigma prilagodljivih, distribuiranih intelligentnih sustava.
  - Okosnicu koncepta intelligentnih agenata čini način distribucije problema u kojem skupina specijaliziranih, samostalnih, intelligentnih komponenti surađuje u njegovu rješavanju.
-

# Agent - definicija

---

- Agent* - poslovođa; posrednik; predstavnik firme; zastupnik, tajni policajac (Anić)
- Agent* - softver koji zna raditi stvari koje biste vjerojatno uradili sami da imate vremena. (Teda Selker, IBM)
- Agent* - računalni proces koji implementira funkcionalnost aplikacija da djeluju i komuniciraju samostalno (Foundation for Intelligent Physical Agents- FIPA)
- In artificial intelligence, an **intelligent agent** (IA) is an autonomous entity which observes and acts upon an environment (i.e. it is an agent) and directs its activity towards achieving goals (i.e. it is **rational**).[1]

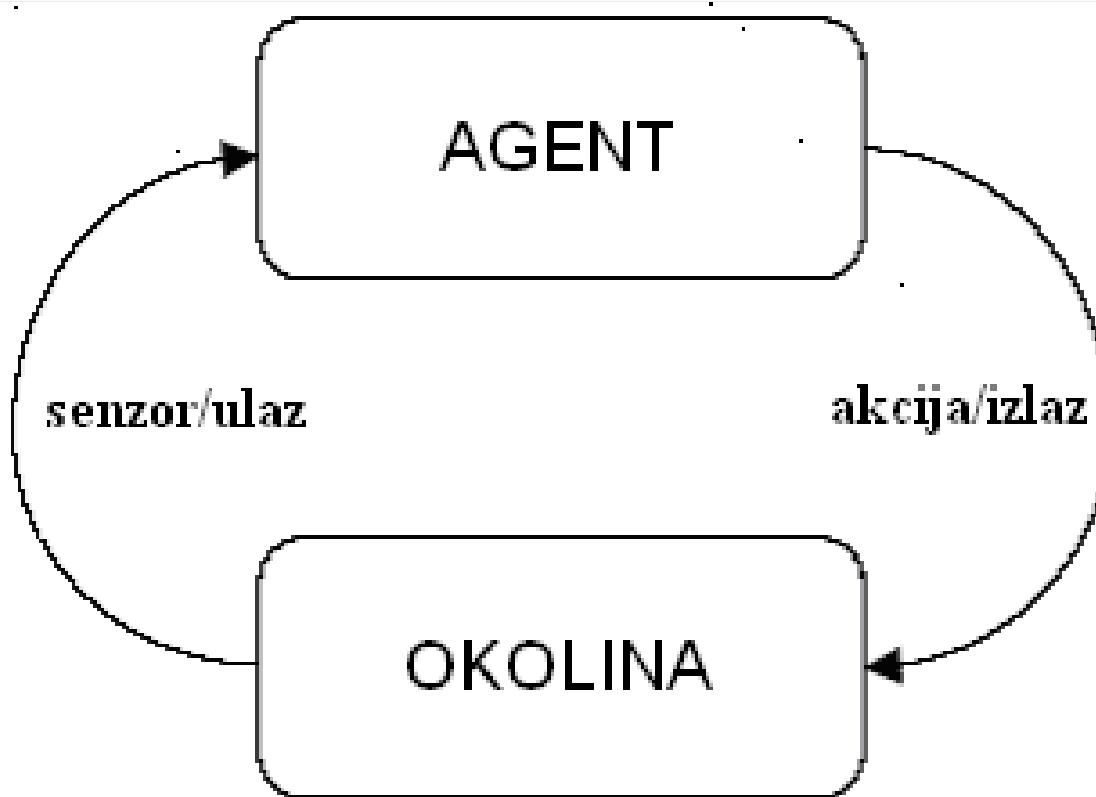
# Inteligentni agent

---

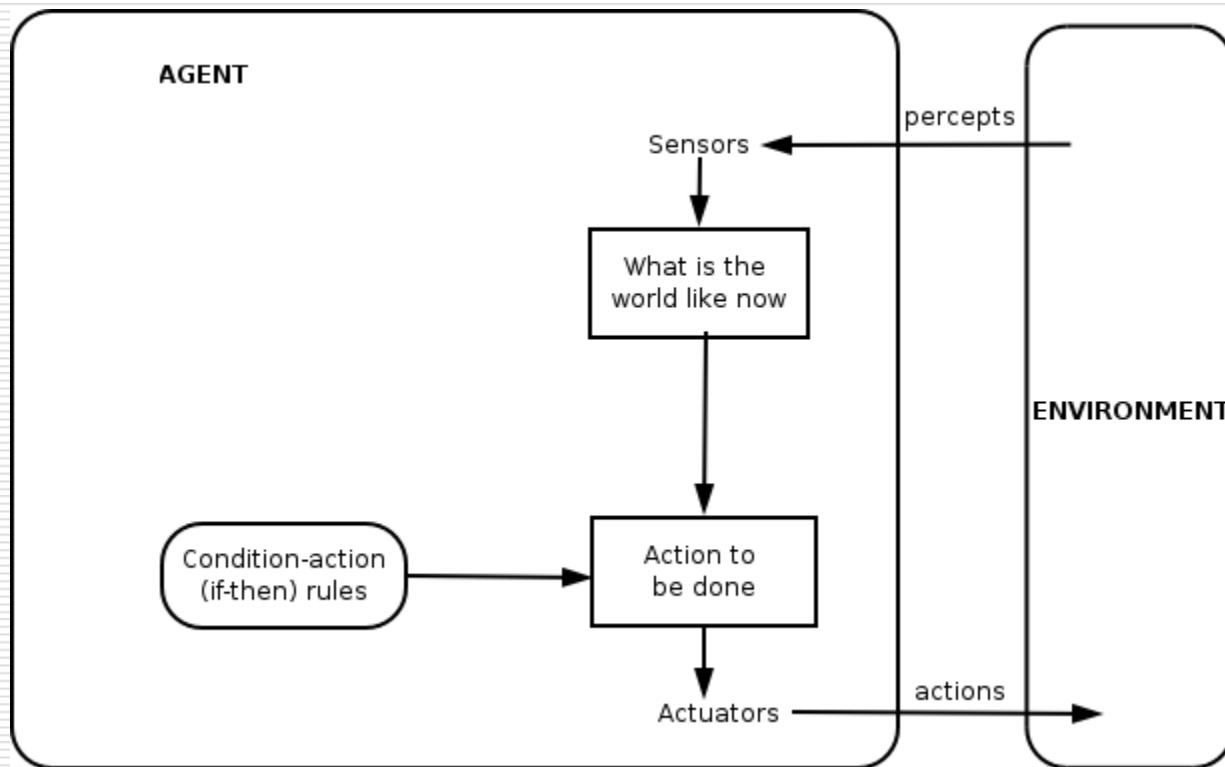
- Agent je računalni sustav koji se nalazi u nekom okolišu i koji je sposoban samostalno djelovati na taj okoliš.
  - Najvažnija karakteristika intelligentih agenata je autonomija (sposobnost samostalnog djelovanja bez direktne intervencije čovjeka ili drugih agenata).
-

# Inteligentni agent

---



# Inteligentni agent - refleksni dijagram



# Inteligentni agent

---

- Inteligentni agent je računalni sustav koji je *autonoman* i *fleksibilan*, te kao takav sposoban izvršavati zadane ciljeve.
- Pod *fleksibilnosti* misli se na slijedeće:
  - *reaktivnost*: agenti moraju percipirati svoju okolinu (koja može biti fizički svjet, korisnik, skupina agenata, mreža itd.) i pravovremeno odgovoriti na promjene.
  - *proaktivnost*: agenti moraju djelovati oportunistički, te preuzeti incijativu i djelovati s obzirom na postavljeni cilj tamo gdje je to potrebno.
  - *socijalna sposobnost*: agenti moraju imati sposobnost komuniciranja sa ljudima ili drugim agentima kako bi postigli rješenje svojeg problema i pomogli drugima u njihovim aktivnostima.

# Inteligentni agent

---

- Istraživači s polja umjetne inteligencije smatraju da je agent računalni sustav koji je uz to što posjeduje navedene osobine smisljen i implementiran na konceptima koji su srodniji ljudima nego računalnim sustavima.
  - Agenti se često karakteriziraju koristeći mentalističke pojmove kao što su znanje, vjerovanja, namjere i obveze tako da ih neki istraživači nazivaju i *emocionalni agenti*.
-

# Inteligentni agent

---

- Takvi agenti imaju neka od slijedećih svojstava:
  - *mobilnost*: sposobnost agenta da se kreće kroz računalnu mrežu.
  - *benevolentnost*: prepostavlja da ciljevi agenata nisu u konfliktu te da će svaki agent uvjek pokušati ostvariti ono što se od njega zahtjeva.

# Inteligentni agent

---

- Takvi agenti imaju neka od slijedećih svojstava:
  - *racionalnost*: prepostavlja da će se agent ponašati na način kojim će ostvariti cilj tj. da se neće ponašati na način koji kompromitira ostvarivanje ciljeva.
  - *adaptivnost*: agent mora biti sposoban prilagoditi se navikama, radnim metodama i preferencijama korisnika.
  - *kolaboracija*: agent ne bi trebao direktno prihvaćati i izvršavati instrukcije već uzeti u obzir da ljudski korisnik može raditi pogreške (npr. dati naredbu koja sadrži konfliktne ciljeve) i izostaviti važne informacije.

# Inteligencija agenta

---

- Stupanj inteligencije agenta ovisi o stupnju autonomije agenta kod reprezentacije korisnika ili drugih agenata, aplikacija ili sistema.
  - Stupanj inteligencije agenta se naziva stupanj *agencije* (eng. *agency*).
  - Može se mjeriti jedino kvalitativno i ovisi o vrsti interakcije između agenata i drugih entiteta u sustavu.
-

# Inteligencija agenta

---

- Što točno čini agenta inteligentnim ?
- Pojam *inteligencije* obuhvaća sposobnost agenta da odlučuje o mogućim akcijama te da planira, komunicira, uči i prilagođava se potrebama okoline.
- Inteligentni agent ima slijedeće elemente:
  - *element percipiranja (senzor)* - uočava promjene u okolini,
  - *prepoznavač ili klasifikator* - prepoznaće tip promjene,
  - *sustav logike* – mehanizam zaključivanja koji može biti direktno kodirani program, sustav zaključivanja na temelju pravila ili neki drugi koncept AI
  - *efektor* - mehanizam za poduzimanje akcije.

# Inteligencija agenta

---

- Za inteligentne agenate važna je paradigma *događaj-stanje-akcija*.
  - Događaj* se definira kao bilo kakva promjena u okolišu ili bilo što čega bi agent treba biti ‘svjestan’. (npr. dolazak nove pošte ili promjena web stranice.)
  - Kada dođe do promjene, agent mora biti u stanju to prepoznati i procjeniti što ta promjena znači.
-

# Inteligencija agenta

---

- Određivanje *stanja* okoliša može biti jednostavno ili ekstremno kompleksno.
- Npr. dolazak nove pošte lako se detektira. Nakon toga agent će možda trebati postaviti upit na sustav pošte da sazna tko je poslao poštu, koji je naslov poruke ili pak skenirati sadržaj poruke i pronaći ključne riječi. Ovo je dio ciklusa *prepoznavanja/klasificiranja* događaja. Inicijalni događaj može aktivirati agenta nakon čega on mora odrediti koja je važnost dotičnog događaja u kontekstu njegovih dužnosti/zadaća. Npr. ako je poruka stigla od korisnikovog nadređenog, onda se ona može klasificirati kao hitna. Nakon toga se pokreće akcija obavještavanja korisnika o pristigloj pošti.

# Okolina agenata

---

- Tipovi agentske okoline :
    - *Dostupne i nedostupne*
    - *Deterministički i nedeterministički*
    - *Epizodni i neepizodni*
    - *Statički i dinamički*
    - *Diskretni i kontinuirani*
-

# Okolina agenata

---

- *Dostupne i nedostupne* – dostupna okolina je ona u kojoj agent ima pristup potpunim, pravovremenim i točnim informacijama o stanju okoline. Što je okolina dostupnija lakše je napraviti agente koji će u njoj djelovati. Kompleksne okoline su u pravilu nedostupne ili djelomično dostupne.
- *Deterministički i nedeterministički* – kompleksni sustavi su najčešće nedeterministički. Stanje koje će prouzročiti neka akcija agenta nije predvidljivo unatoč sličnosti prijašnjim stanjima prije provođenja akcije. Ova nesigurnost predstavlja veći izazov za agenta nego kod determinističkih sustava.

# Okolina agenata

---

- *Epizodni i neepizodni* – U epizodnom okolišu djelovanje je podjeljeno na jodnokratne, neovisne epizode. U svakoj epizodi agent percipira i provodi jednu akciju. Ovakvi sustavi su lakši za implementaciju jer agent ne mora voditi računa o interakciji prošlih i budućih epizoda
- *Staticki i dinamički* – Statički okoliši ostaju nepromjenjeni osim u slučaju akcija agenata. U dinamičkom okolišu djeluju drugi procesi te mjenjaju njegovo stanje neovisno o agentu. Din. sustavi zahtjevaju kompleksniji dizajn agenta.
- *Diskretni i kontinuirani* – ako je broj akcija i izhoda fiksan i konačan onda je okolina diskretna.

# Okolina agenata

	Križaljka	Šah	Backgammon	Internet kupovina	Taksi-vožnja	Robot za kupljenje dijelova
Potpuno dostupna	da	da	da	ne	ne	ne
Deterministička	da	djel.	ne	da	ne	ne
Epizodna	ne	ne	ne	ne	ne	da
Statička	da	djel.	da	djel.	ne	ne
Diskretna	da	da	da	da	ne	ne
Jednoagentska	da	ne	ne	ne	ne	da

Tipovi agentske okoline kod različitih primjena agenata

# Struktura agenta

---

- Agent sastoji se od *programa* i *arhitekture*.
  - Arhitektura** - hardverska podloga na kojoj se agentski program izvodi (obično računalo ili hardver specijaliziran za određene zadaće, a može uključivati softver koji predstavlja sučelje između hardvera i agentskog programa, npr. operacijski sustav )
-

# Struktura agenta

---

- Arhitektura kod *multiagentskih sustava* (*MAS*) se odnosi na određeni model uređenja multiagentskog sustava te koordinacije i komunikacije unutar njega.
  - Ovisno o okolini u kojoj agent djeluje senzori i efektori mogu biti posebni uređaji - kamere, sonari, robotske ruke i sl. a mogu biti ulazi i izlazi iz programa u nekom softverskom okolišu.
  - Arhitektura omogućuje primanje podataka sa senzora programu agenta, pokretanje agenta i izvršenje akcije na efektorima.
-

# Struktura agenta

---

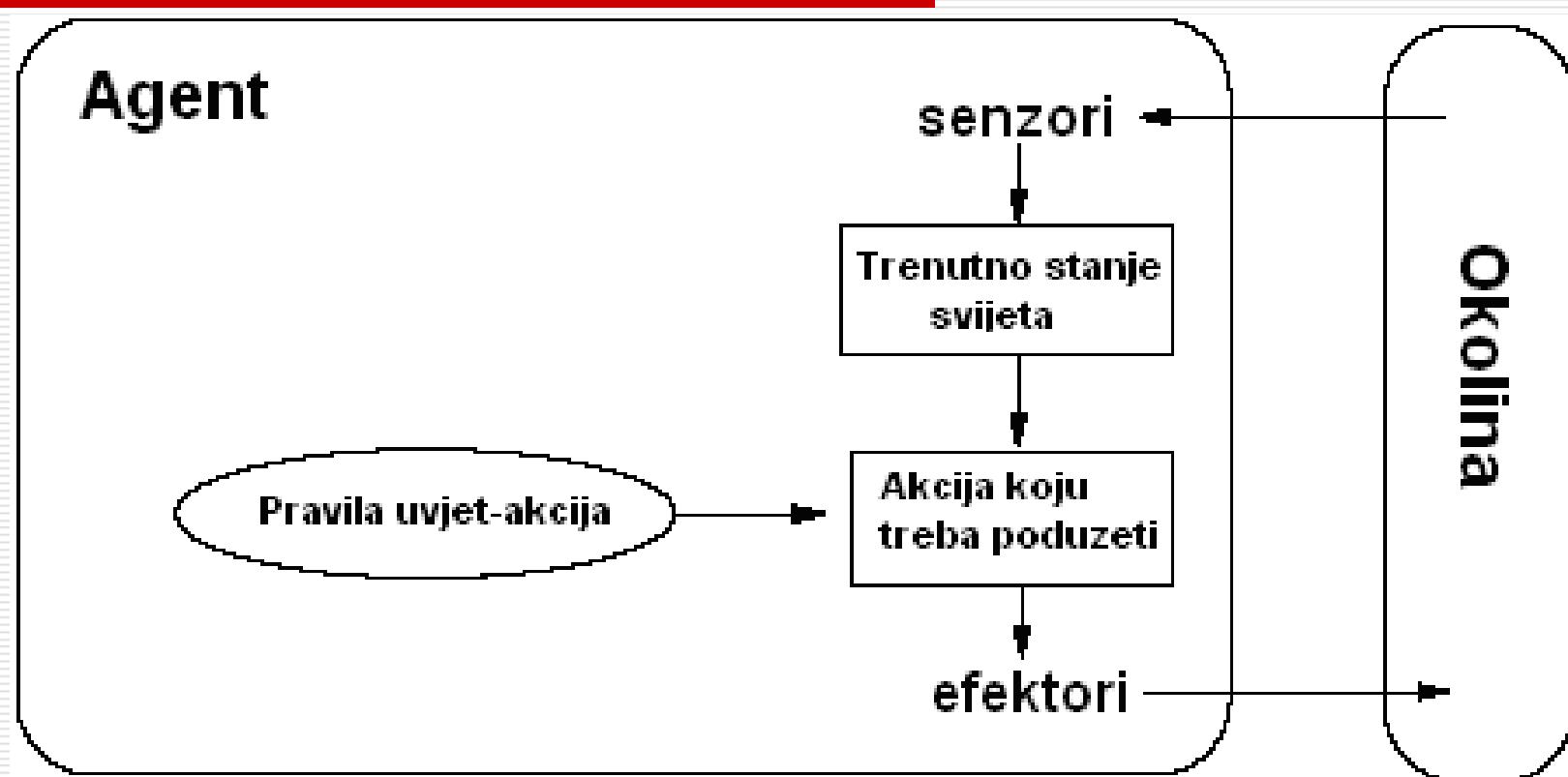
- Četiri osnovne strukture agenata:
    - Jednostavni refleksni agenti
    - Agenti koji pohranjuju stanje okoline
    - Agenti pogonjeni ciljem
    - Utility agenti ili agenti pogonjeni stupnjem korisnosti.
-

# Jednostavni refleksni agenti

---

- Većina agenata sadrži sustav zaključivanja baziran na pravilima *dogadjaj-akcija* (produkcijska pravila ili if-then pravila).
- Ona određuju pokretanje neke akcije ako je ostvaren određen uvjet.
  - *ako <UVJET ispunjen> onda <pokreni AKCIJU>* .
- Jednostavni refleksni agenti rade na način da pronađu pravilo čiji uvjet odgovara trenutnoj situaciji i pokrenu odgovarajuću akciju.

# Jednostavni refleksni agenti



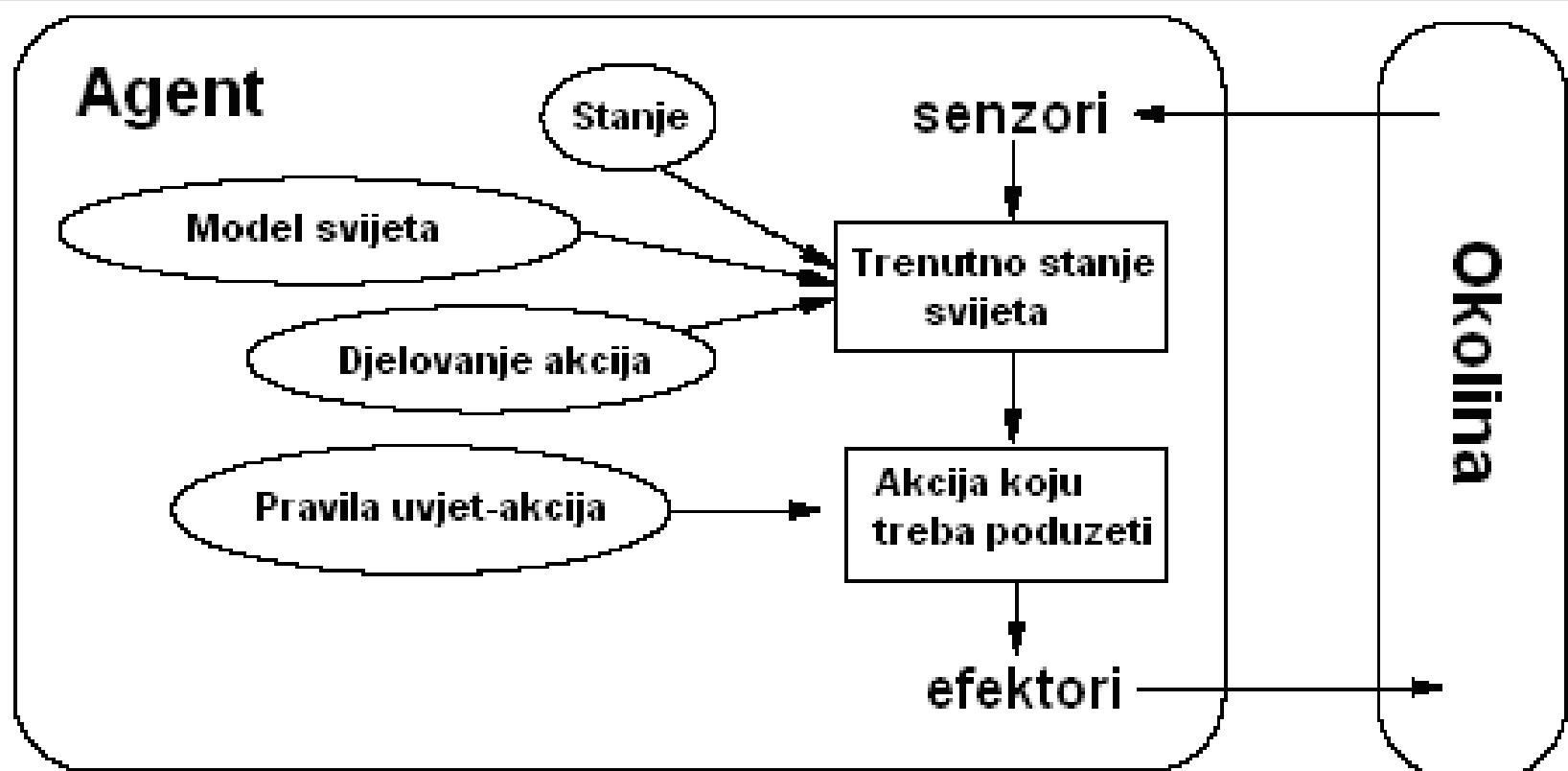
Jednostavni refleksni agent

# Agenti koji pohranjuju stanje svijeta

---

- Trenutno stanje svijeta u većini slučajeva nije dovoljno da bi se mogla korektno procjeniti određena situacija.
  - Ovi agenti mogu pohranjivati prijašnje percepse te ih uspoređivati sa trenutnim kako bi dobili precizniju sliku.
  - Npr. ako agent ima podatak od prije 0.2 sekunde da je vozilo ispred bilo udaljeno 17m može zaključiti da vozilo ispred njega usporava i pokrenuti pravilo: ako <vozilo ispred usporava> onda <prilagoditi brzinu> .
-

# Agenti koji pohranjuju stanje svijeta



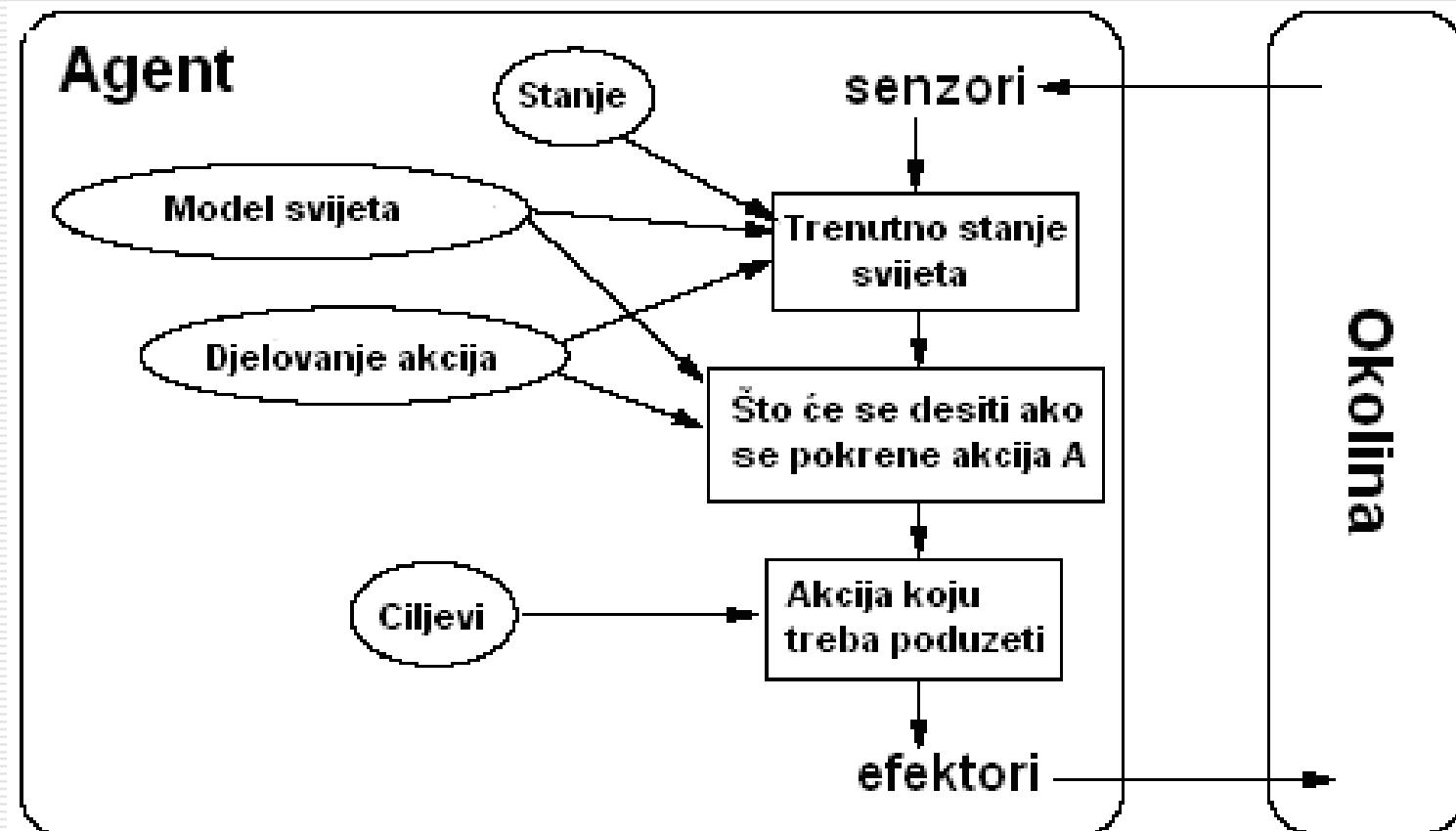
Agenti koji pohranjuju stanje svijeta

# Agenti orjentirani na ciljeve

---

- Kako bi se omogućila veća fleksibilnost agenata, ovi agenti imati ciljeve koji opisuju situacije koje su poželjne.
  - Agent može kombinirati opis ciljeva sa rezultatima mogućih akcija kako bi odabroao one koje ostvaruju cilj.
  - Da bi bio u stanju to postići mora sadržavati mehanizme umjetne inteligencije za pretraživanje stanja i planiranje.
-

# Agenti orijentirani na ciljeve



Agenti orijentirani na ciljeve

# Agenti pogonjeni stupnjem korisnosti

---

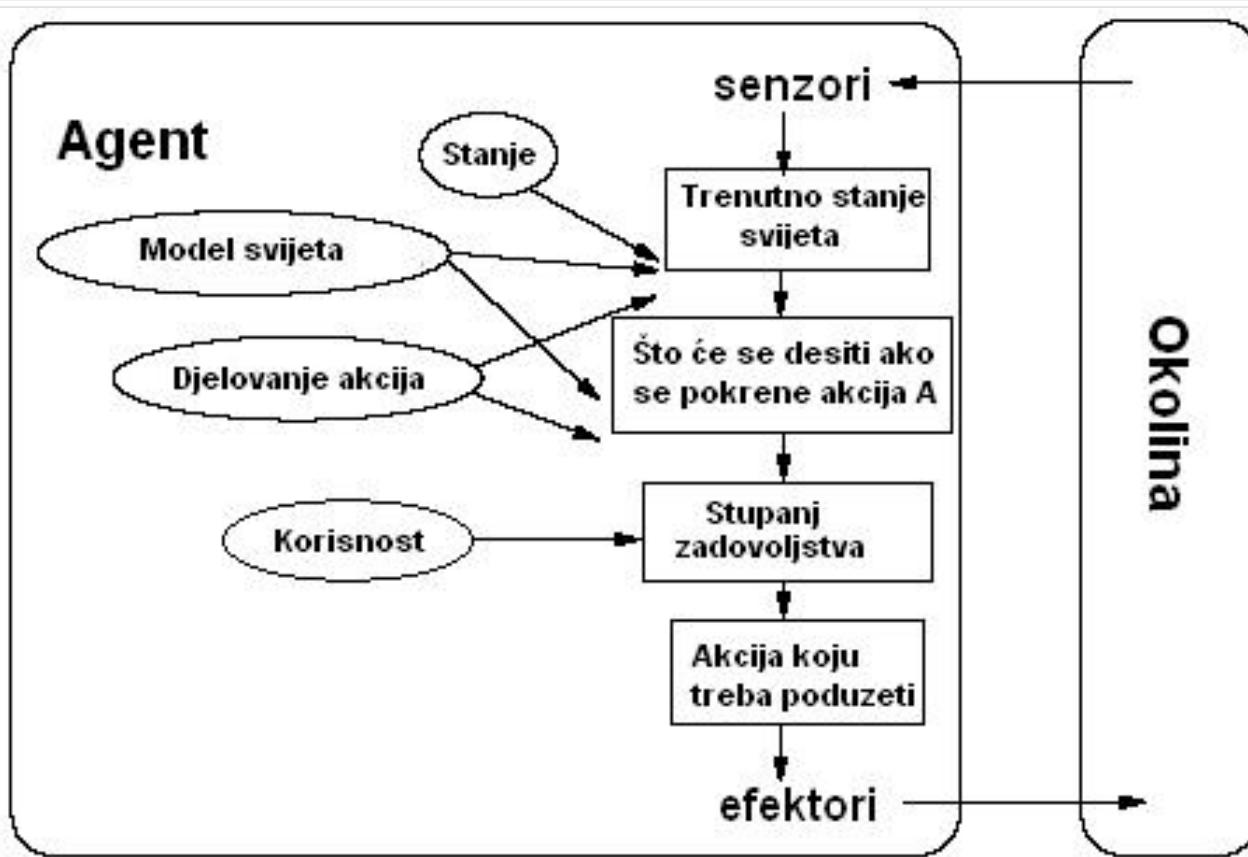
- Kod agenta pogonjenih ciljevima postoji samo distinkcija između prihvatljivih i neprihvatljivih stanja kod odlučivanja.
  - Utility agenti* ovaj proces poboljšavaju uvođenjem određenog stupnja 'zadovoljstva' agenta napravljenom akcijom.
  - Ako neko stanje svijeta omogućuje veće 'zadovoljstvo' agenta ima veći *stupanj korisnosti-utility*.
  - Agent posjeduje funkciju koja opisuje stupanj korisnosti neke akcije realnim brojem.
  - Također ovaj način omogućuje odlučivanje u slučaju nesigurnosti i konflikata ciljeva.
-

# Agenti pogonjeni stupnjem korisnosti

---

- Agent sa funkcijom korisnosti može donositi racionalne odluke ali za to mora provesti kompleksan proces usporedbe korisnosti pojedinih smjerova djelovanja.
  - Ciljno orijentirani ili refleksni agenti s druge strane, mogu odrediti slijedeću akciju mnogo brže.
-

# Agenti pogonjeni stupnjem korisnosti



# BDI model

---

- Najpoznatiji model agentske strukture koja predstavlja agenta kao skup *vjerovanja, želja(ili ciljeva) i namjera* (eng. Belief, Desire, Intention).
- BDI arhitektura se sastoji od sljedećih elemenata:
  - *Vjerovanja*
  - *Želje*
  - *Namjere*
  - *Zbirka planova*
  - *Interpreter*

# BDI model

---

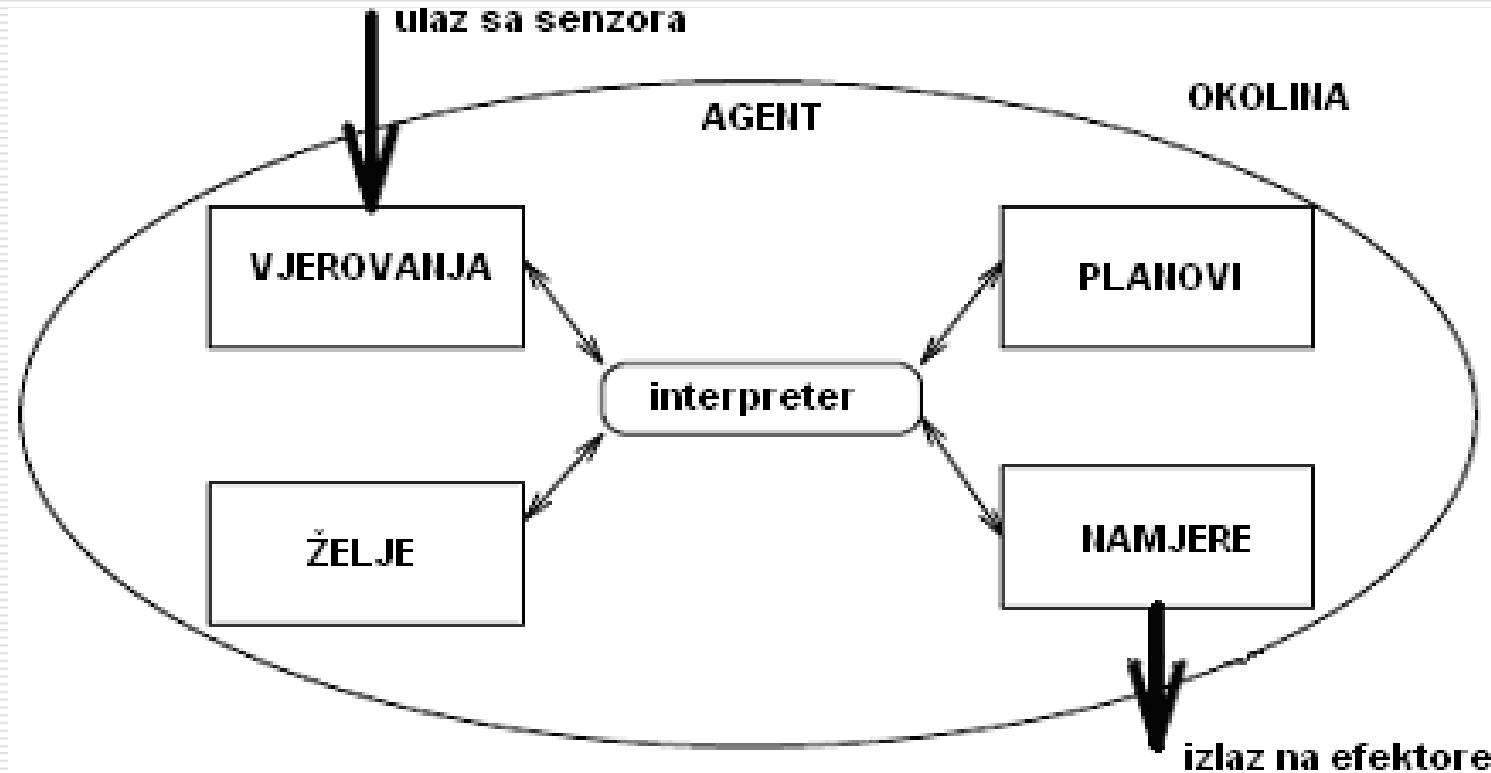
- *Vjerovanja* – informacije koje agent ima o svijetu, a mogu biti nepotpune ili pogrešne; mogu biti jednostavne varijable ili izražene u nekom simboličkom jeziku (npr. PROLOG).
  - *Želje* – predstavljaju intuitivan način prikaza alociranih zadataka - odgovaraju mogućim akcijama i formiraju se na temelju ulaza i vjerovanja. Moraju biti logički konzistentne.
  - *Namjere* – predstavljaju želje agenta koje namjerava ostvariti. Agent neće moći ostvariti sve svoje želje, čak i ako su logički konzistentne, stoga mora odrediti neki podskup želja i upotrijebiti resurse kako bi ih ostvarili. Želje koje je agent odlučio ostvariti su *namjere* i pokušavat će ih ostvariti dok ne bude vjerovao da je namjera ostvarena ili da se više ne može ostvariti.
-

# BDI model

---

- *Zbirka planova* – skup planova koji specificiraju više mogućih tijekova akcija za ostvarivanje želja agenta.
- Plan agenta predstavlja proceduralno znanje, a sastoji se od dva dijela:
  - *tijela* ili programa koji definira tijek akcije i
  - *deskriptora* koji određuje uvjete pod koji okolnostima plan može biti korišten i koje namjere može ostvariti.
- *Interpreter* – dio odgovoran za osvježavanje vjerovanja putem promatranja promjena u stanju svijeta.
- Generira nove želje(zadatke) na temelju vjerovanja i određuje skup trenutno aktivnih želja koje postaju namjere.
- Na kraju interpreter mora odrediti akciju koju želi provesti na temelju agentovih trenutnih namjera i proceduralnog znanja.

# BDI model



# Multiagentski sustavi (MAS)

---

- Multiagentski sustavi (**MAS**) se sastoje od 2 ili više agenta koji zajednički rješavaju probleme čije rješenje nije u dosegu samostalnog agenta.
  - Proučava ih polje umjetne inteligencije distribuiranog rješavanja problema (*Distributed Problem Solving* – DPS).
-

# Multiagentski sustavi (MAS)

---

- MAS sustavi imaju slijedeća svojstva:
    - otvoreni sustavi decentraliziranog dizajna
    - sadrže autonomne, heterogene i distribuirane agente sa različitim ‘osobnostima’ i sposobnostima
    - imaju infrastrukturu koja određuje komunikaciju i protokole za agentsko međudjelovanje
    - postoje višestruki pogledi na sustav
    - postoji više metoda rješavanja problema.
-

# Multiagentski sustavi (MAS)

---

- Dobiti MAS sustava:
    - povećana robusnost, efikasnost, fleksibilnost, adaptivnost i skalabilnost sustava
    - međusobno povezivanje postojećih legacy sustava
    - mogućnost rješavanja problema prevelikih i prekompleksnih za jednog centraliziranog agenta
    - mogćnost primjene u domenama gdje su podaci, ekspertno znanje i kontrola distribuirani
    - moguće povećanje brzine, pouzdanosti i proširivosti sustava
    - MAS sustavi nude jednostavan i konceptualno ‘čist’ dizajn.
-

# Multiagentski sustavi (MAS)

---

- Problemi MAS sustava:
    - Formulacija, opis, dekompozicija i alokacija problema i sinteza rezultata u grupi inteligenčnih agenata.
    - Komunikacija i interakcija među agentima: koje komunikacijske jezike i protokole koristiti kako bi se ostvarila smislena interakcija među agentima tj. kada i kako komunicirati?
    - Koordinacija među agentima: kako uskladiti proces planiranja, akcija i vjerovanja agenata sa strategijskim planovima za ostvarenje cilja sustava.
-

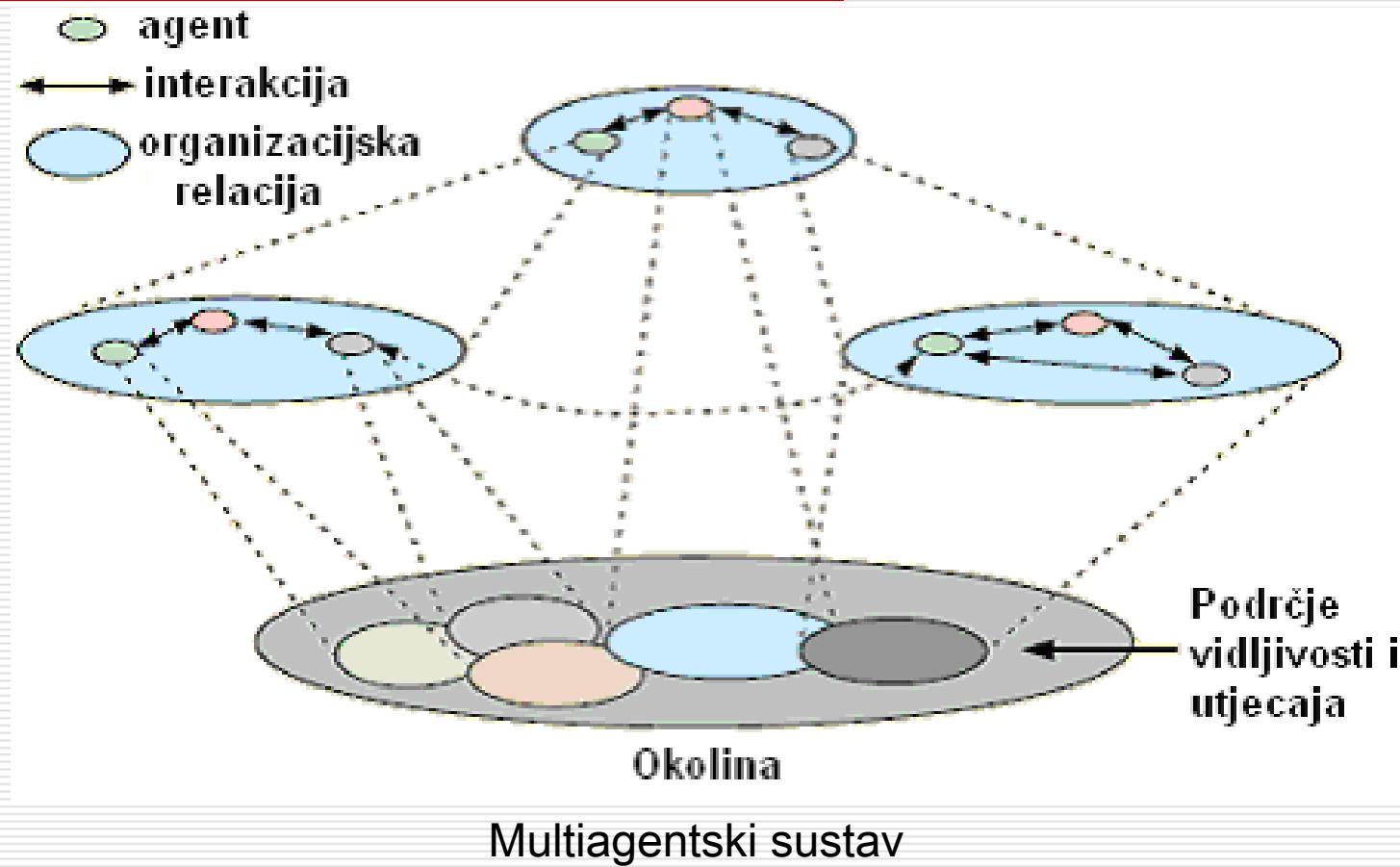
# Multiagentski sustavi (MAS)

---

## □ Problemi MAS sustava:

- Identifikacija i usklađivanje različitih pogleda i konflikata među agentima koji pokušavaju koordinirati svoje akcije.
  - Ujednačavanje lokalnog procesiranja i komunikacije: kako izbjegći preoptereženje resursa pomoću strategija raspodjele opterećenja?
  - Implementacija MAS sustava: kako konstruirati praktične MAS sisteme i koje tehnološke platforme i metodologije razvoja koristiti?
  - Verifikacija i korekcija MAS aplikacija koristeći formalne i praktične pristupe.
-

# Multiagentski sustavi (MAS)



# Vrste MAS sustava

---

- Ovisno o stupnju kooperacije koje pokazuju pojedini agenti razlikujemo:
  - *Kooperativne multiagentske sustave* (eng. *Cooperative MAS - CMAS*) – kod ovih sustava važne su sveukupne performanse sustava. Stoga svi agenti u sustavu moraju funkcionirati kooperativno. Dizjnera takvog sustava ne zanimaju performanse pojedinog agenta.
  - *Sustave agenta sa individualnim interesom* (eng. *Self-Interested MAS- SMAS*) – agenti ovdje su individualno motivirani.
-

# Međuagentska komunikacija

---

- Agenti međusobno komuniciraju na različite načine, bilo direktno ili indirektno.
  - Agenti na strani korisnika komuniciraju sa serverskim agentima.
  - Serverski agenti opet registriraju svoje usluge sa agentima brokerima.
  - Korisnički agenti pronađuju server pomoću brokera.
  - Serverski agenti mogu slati korisničkim agentima obavijesti o događajima određenog tipa putem pretplate.
-

# Međuagentska komunikacija

---

- Vrste agentske komunikacije u distribuiranim sustavima :
    - Sinkrona komunikacija putem poruka
    - Asinhrona komunikacija putem poruka
    - Komunikacija putem pretplate/notifikacija
    - Transakcijska podrška
    - Brokerska komunikacija
    - Pregovaranje.
-

# Međuagentska komunikacija

---

- *Sinkrona komunikacija putem poruka* (*Remote Procedure Call*) – klijent šalje poruku serveru i čeka na odgovor. Na serveru je moguć nastanak redova čekanja. U klijent/server arhitekturi moguće je da server delegira obradu zahtjeva klijenta na neki drugi server koji onda komunicira sa klijentom. Moguća je višestruka client/server komunikacija.
  - *Asinhrona komunikacija putem poruka* – koristi FIFO ili rangiranje po prioritetu unutar redova poruka. Server ne mora čekati potvrdu o primitku poruke kako bi poslao sljedeću poruku. U distribuiranim okolišima, asinhrona komunikacija porukama se koristi kada god je to moguće zbog veće fleksibilnosti jer pošiljaoc ne treba čekati potvrdu od primaoca.
-

# Međuagentska komunikacija

---

- Komunikacija putem pretplate/notifikacija* – u ovoj vrsti komunikacije više primaoca prima jednu poruku. Grupna komunikacija može biti opća (*broadcast*) ili pretplaćena(*multicast*). U slučaju opće komunikacije nesolicitirane poruke se prenose svim klijentima nekog sustava, npr. informacija o servisnom gašenju servera. Kod pretplaćene komunikacije server šalje notifikaciju samo onim klijentima sa liste pretplatnika.
  - Transakcijska podrška* – za transakcije koje trebaju biti atomske (nedjeljive) moraju postojati servisi koji će otpočeti transakciju, provesti je i završiti. Ovaj način se koristi za osvježavanje distribuiranih baza podataka koje moraju biti nedjeljive.
-

# Međuagentska komunikacija

---

- *Brokerska komunikacija* – agent broker služi kao posrednik između klijenta i servera. On oslobađa klijenta obveze čuvanja informacija o tome kako i gdje pronaći određenu uslugu. Serveri registriraju svoje lokacije i usluge koje pružaju kod brokera. Klijentski agenti onda šalju upite o dostupnim uslugama.
  - *Pregovaranje* – agenti razmjenjuju prijedloge i odgovaraju sa ponudama koje mogu biti prihvaćene, odbačene ili izmjenjene.
-

# Međuagentska komunikacija

---

- Za sustave inteligentnih agenta najfleksibilnije su indirektne forme komunikacije, točnije komunikacija putem pretplate/notifikacie i brokerske komunikacije.
  - Kod ovih pristupa agenti mogu pronalaziti usluge koje pružaju drugi agenti nakon što se one uključe. Mogu pratiti usluge koje oni pružaju i pretpisati se kod drugih agenata.
-

# Brokerska komunikacija

---

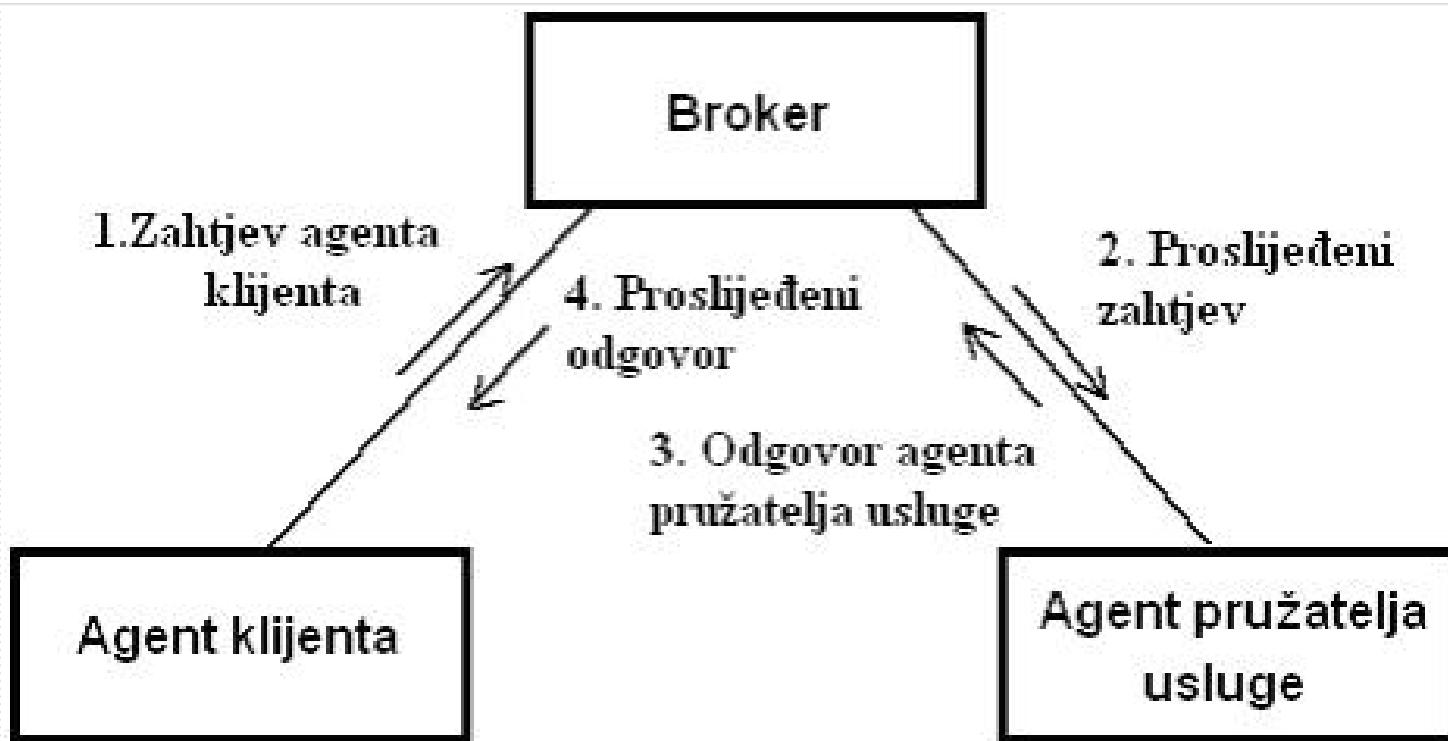
- Agent broker služi kao posrednik između agenata klijenta i agenata servera i interakcija između klijenata i objekata na serveru. Omogućava *transparentost lokacije* (u slučaju promjene adrese servera treba obavjestiti samo agenta brokera. Agent klijenta šalje poruku kojom zahtjeva određenu uslugu, npr. videokonferencijsku uslugu određenog ponuđača).
-

# Brokerska komunikacija

---

- Broker prima taj zahtjev, određuje lokaciju agenta pružatelja usluge(tj. čvor na kojem se dotični nalazi) i proslijeđuje mu poruku klijenta. Nakon toga pružatelj usluge šalje odgovor brokeru, koji ga dalje proslijeđuje klijentu.
  - Ovakva vrsta komunikacije u kojoj agent klijenta zna naziv agenta servisa kojeg traži naziva se '*white pages*' brokering
-

# Brokerska komunikacija



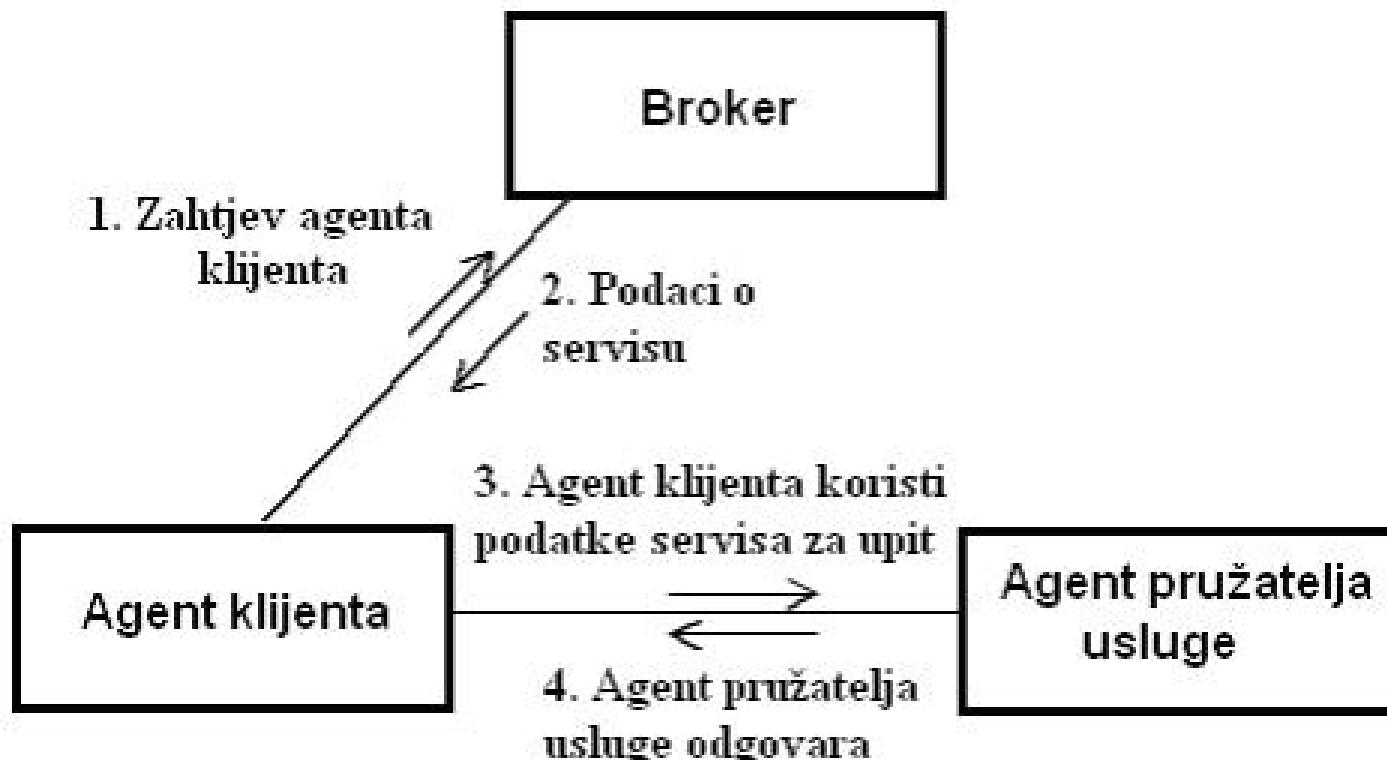
*White pages brokering*

# Brokerska komunikacija

---

- Druga vrsta komunikacije '*bijelih stranica*' putem brokera je razmjena podataka o nekom servisu. Broker na zahtjev klijenta pronalazi podatke o servisu (service-handle) i šalje klijentu lokaciju servisa . Nakon toga klijent komunicira direktno sa serverom.
- Ovaj pristup je prikladniji jer rasterećuje brokera u slučaju da klijent i server trebaju razmjeniti više poruka.

# Brokerska komunikacija



Razmjena podataka o nekom servisu

# Brokerska komunikacija

---

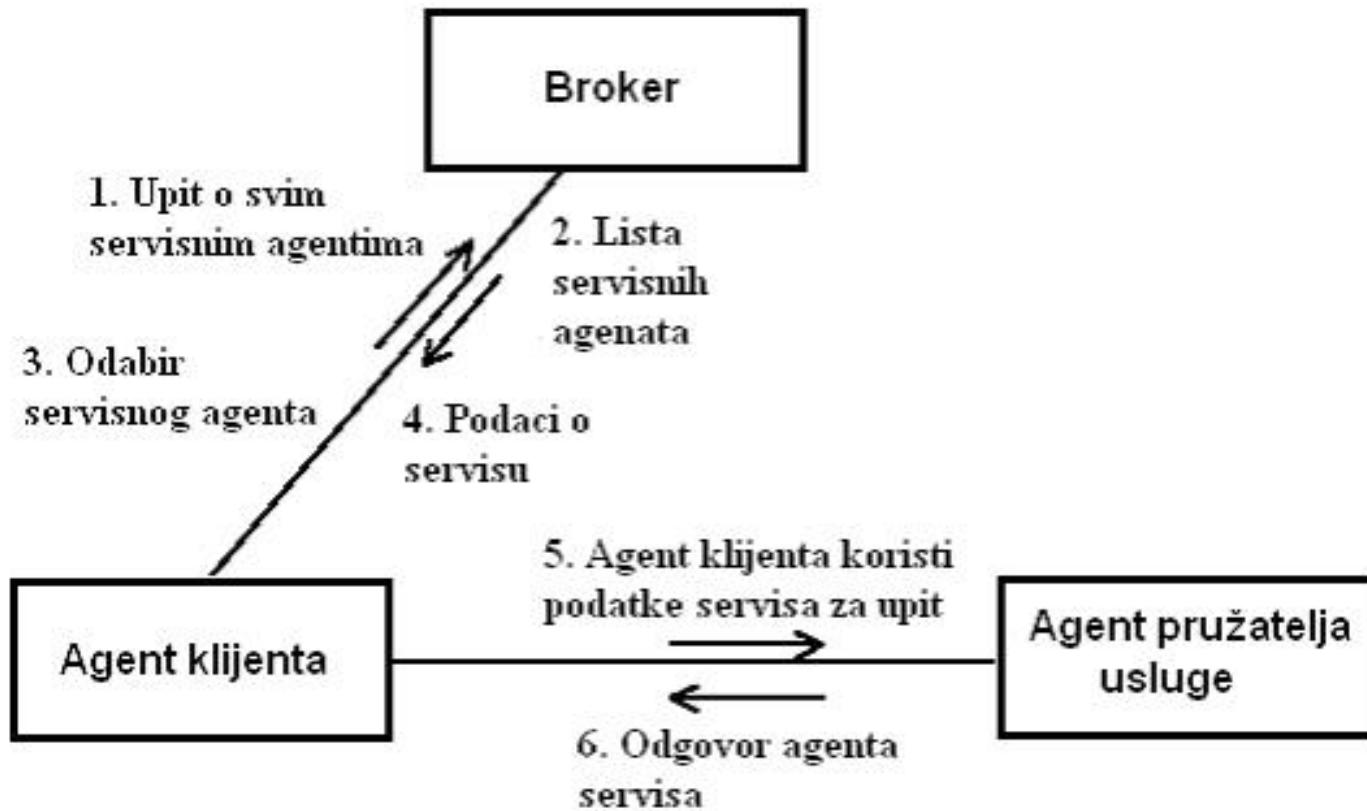
- Treća vrsta brokerske komunikacije je ona putem '**žutih stranica**' tj. imenika.
  - Agent klijenta zna tip usluge koju zahtjeva ali ne i naziv specifičnog agenta pružatelja te usluge.
-

# Brokerska komunikacija

---

- Klijent šalje upit brokeru u kojem zahtjeva podatke o svim agentima koji pružaju određenu vrstu usluge. Broker odgovara sa listom svih pružatelja usluga čiji agenti odgovaraju kriterijima koje je tražio klijent. Nakon toga klijent komunicira direktno sa servisom.
-

# Brokerska komunikacija



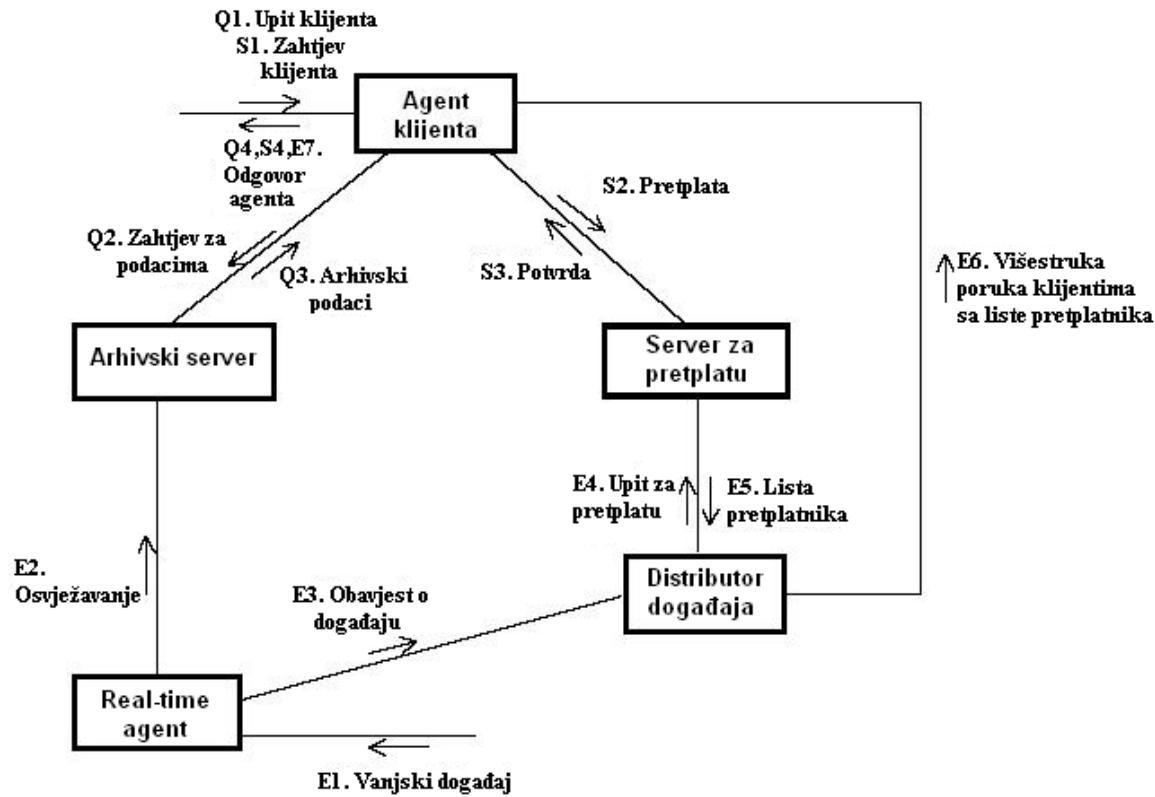
*Komunikacija putem imenika*

# Brokerska komunikacija

---

- Komunikacija putem pretplate i notifikacije***
- Koristi se kod real-time agenata koji prate promjene u okolini.
- RT agent održava pretplatničku listu klijenata koji žele primati obavijesti o određenom tipu događaja. Kod detekcije neke promjene agent utvrđuje njen prioritet i po potrebi obavještava agente sa liste pretplatnika.

# Brokerska komunikacija



*Komunikacija putem preplate i notifikacije*

# Klasifikacija agenata

---

- Klasifikacija agenata po svojstvima:
    - kolaborativni,
    - osobni,
    - pametni.
  - Agenti mogu kombinirati svetri sposobnosti.
  - Osobni agenti su proaktivni i služe individualnim korisnicima - također mogu biti i adaptivni.
  - Adaptivni agenti mogu u pozadini pretraživati korisničke informacije i često služe pri pretraživanju Web-a.
  - Kolaborativni agenti su i proaktivni i kooperativni
-

# Klasifikacija agenata

---

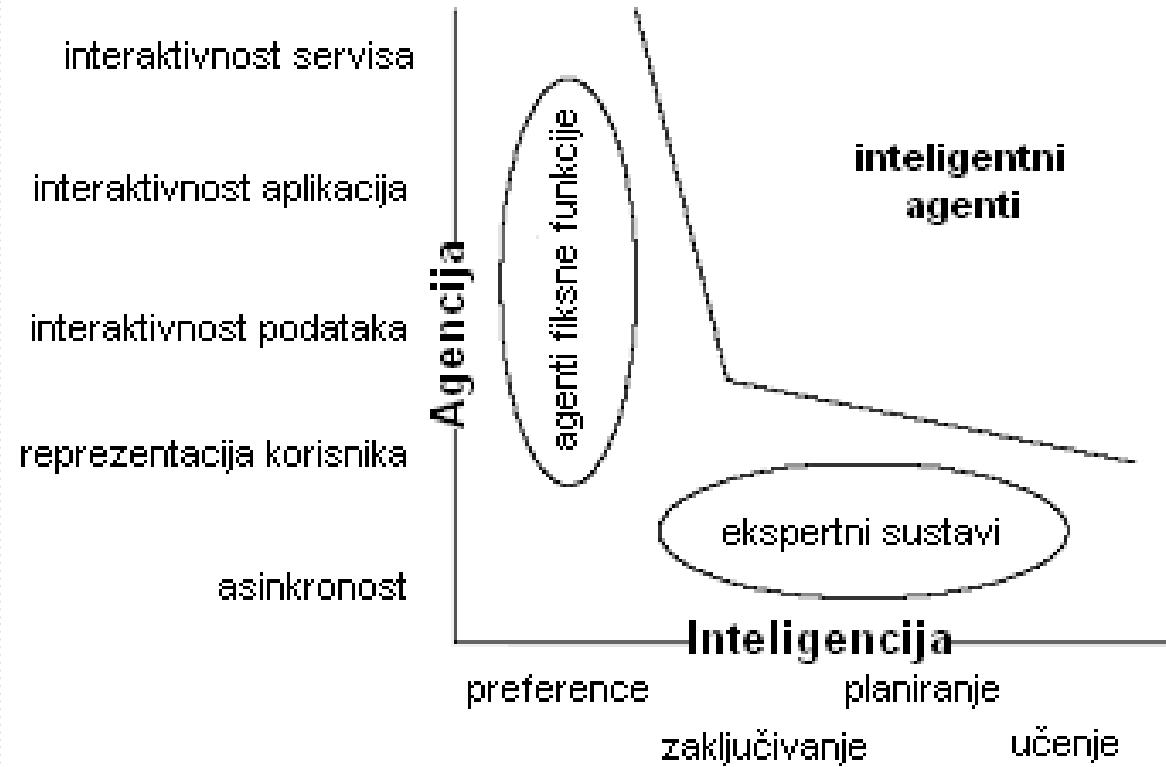


## IBM klasifikacija

---

- Uzima u obzir dva kriterija:
    - stupanj agencije i
    - inteligenciju agenta.
  
  - Ovisno o inteligenciji agenti se mogu kretati od jednostavnih specifikatora korisničkog sučelja do agenata koji su u stanju aktivno zaključivati i učiti.
-

# IBM klasifikacija



IBM klasifikacija

## IBM klasifikacija

---

- Razlikuje slijedeće skupine agenata:
    - agente za filtriranje,
    - informacijske agente,
    - agente korisničkog sučelja,
    - uredske ili agente za kontrolu toka, sistemske agente,
    - brokere ili komercijalne agente.
-

# IBM klasifikacija

---

- Agenti za filtriranje**
  - Služe kao filteri informacija u nekom mrežnom okruženju.
  - Ovisno o podešenju i preferencama korisnika, oni propuštaju samo relevante informacije dok u isto vrijeme blokiraju pristup beskorisnim podacima.
  - Filteri rade na nekoliko načina od koji je najčešće korišten način onaj u kojem korisnik definira neki predložak ili profil područja od interesa.
-

# IBM klasifikacija

---

- Informacijski agenti**
  - Rade na sličnom principu kao filtrirajući agenti s razlikom da umjesto da propuštaju relevantne informacije koje korisnik prima ovi agenti aktivno traže podatke za korisnika.
  - Prvenstveno se koriste za pretraživanje Interneta i World Wide Weba i mogu skenirati baze podataka, skupine dokumenata ili direktorije u potrazi za korisnim podacima.
-

## IBM klasifikacija

---

- Agenti korisničkog sučelja
  - Prate interakciju korisnika i mogu kontrolirati različite aspekte te interakcije, kao što je količina upozorenja i uputa koje program daje korisniku ili broj ponuđenih opcija.
  - Kreiraju korisničke profile ovisno o stupnju interakcije korisnika s agentom.
-

## IBM klasifikacija

---

- Uredski agenti i agenti kontrole toka**
  - Služe za automatiziranje nekih rutinskih, svakodnevnih zadaća koje zahtjevaju relativno mnogo vremena.
  - Ovi zadaci uključuju automatsko zakazivanje sastanaka, slanje faksova, održavanje sastanaka i obnavljanje dokumenata.
-

# IBM klasifikacija

---

- Sistemski agenti**
  - Glavni im je zadatak upravljanje operacijama računalnog sustava i komunikacijske mreže.
  - Nadgledaju stanje sustava kako bi otkrili kvarove uređaja ili preopterećenje sustava i preusmjeravaju opterećenje na druge resurse kako bi se osigurale određene performanse i pouzdanost sustava.
  - Njihova važnost najviše dolazi do izražaja kod distribuiranih sustava.
-

# IBM klasifikacija

---

- Brokerski agent**
  - Program koji preuzima zahtjeve kupca te pretražuje određeno područje po zadanim kriterijima kako bi pronašao moguće prodavače.
  - Kada pronađe prodavače koji bi možda mogli zadovoljiti korisnikove zahtjeve on vraća rezultate korisniku, koji odabire prodavača i ručno provodi transakciju.
  - Agent može također automatski provesti transakciju umjesto korisnika.
-

# DARPA klasifikacija

---

- Razlikuje slijedeće vrste agenata:
    - Izvršni agent
    - Korisnički agent
    - Real-time agent
    - Facilitacijski agenti
    - Medijatori
    - Agenti aktivnog pogleda(Active View)
    - Informacijski kuratori
    - Pretraživači znanja
    - Radni agenti.
-

# DARPA klasifikacija

---

- Izvršni agent (koordinator)* - usklađuje aktivnosti skupine agenata. Prati bitne događaje u sustavu i u skladu sa potrebama aktivira nove agente.
  
  - Korisnički agent* - djeluje umjesto korisnika, služi kao podrška u pregledavanju kataloga i informacijskih repozitorija, inteligentnoj formulaciji upita i planiranju zadaća unutar scenarija specifičnog za neku misiju
-

# DARPA klasifikacija

---

- *Real-time agent* - definirani i konfiguirirani za praćenje i procesiranje ulaznih podataka i osvježavanje pripadajućih baza podataka i notifikaciju korisnika. Autonomni agenti, komuniciraju međusobno koristeći predefinirane protokole. Oni su odgovorni za praćenje stanja sustava.
  
  - *Facilitacijski agenti* - pružaju usluge inteligentog riječnika i nalaženja objekata. Služe za pretraživanje vanjskih sustava.
-

# DARPA klasifikacija

---

- *Medijatori* - pomažu pri integraciji informacija iz različitih izvora, različitih značenja i vremenskih jedinica i osiguravaju određeni nivo kvalitete informacija. Primaju upite od koordinatora i prevode ih u jezik određene baze podataka, primaju rezultat tih upita te ga integriraju sa rezultatima iz ostalih izvora. Nakon toga šalju tu informaciju koordinatoru koji on proslijeđuje korisničkom agentu.
  
- *Agenti aktivnog pogleda (Active View)* - služe za podršku korisničkih pogleda na višestruke, autonomne i heterogene izvore podataka. Obnavljaju i sinhroniziraju aktivne poglede na baze na korisničkim radnim stanicama.

# DARPA klasifikacija

---

- Informacijski kuratori* - odgovorni su za kvalitetu informacije u informacijskom repozitoriju. Pomažu kod unaprijedivanja baze znanja u organizaciji. Rade sa pretraživačima znanja kako bi ugradili novootkrivene resurse u informacijse repozitorije.
  
  - Pretraživači znanja (eng. Knowledge Rovers)* - provode specifične zadaće za kooordinatora kao što su identificiranje skupina ponuđača određenih proizvoda ili usluga. Oni mogu poslati više *radnih agenata* (eng. Field Agent) na specifične lokacije kako bi pribavili relevantne informacije.
-

# DARPA klasifikacija

---

- Radni agenti* - specijalizirani pretraživači koji sadrže ekspertno znanje određene domene npr. farmaceutike i lokacije na kojima se to znanje može pronaći.
- Npr. radni agent može primiti zahtjev za nadgledanje jedne stavke kao što je antibiotik kojeg proizvodi nekoliko proizvođača i distribuira nekoliko prodavača. Radni agent pregovara sa lokalnim sistemima kroz njihov *wrapper*, pribavlju potrebne podatke i proslijeđuju ih agentu koji ih je zahtjevao.