

# Priprema za Završni ispit

Oblikovanje programske potpore  
5. siječnja 2011.

# Završni ispit

- Sastoji se od cca. 20 zadataka vrijednosti 1 ili 2 boda.
- Cijelo gradivo - ravnomjerno zastupljeno.
- Iz 3. ciklusa:
  - Testiranje,
  - Logika,
  - CTL,
  - Protočne arhitekture.

# Zadaci 1 - 4

1. Temeljni nedostatak evolucijskog modela procesa programskog inženjerstva je:  
**loša struktura, arhitektura**
2. Model procesa programskog inženjerstva koji je najprikladniji kada zahtjevi u početku nisu potpuno (do kraja) definirani je:  
**evolucijski**
3. Obrasci uporabe koriste se pri modeliranju:  
**funkcionalnih / nefunkcionalnih zahtjeva.**
4. Za razliku od sekvencijskih dijagrama kojima je u fokusu vremenska uređenost između događaja, u fokusu UML komunikacijskih (kolaboracijskih) dijagrama je:  
**identifikacija sudionika komunikacije, tko s kime, sučelje ...**

# Zadaci 5 - 7

5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi Kocka, Kvadar, Valjak, Kugla i Stožac su „spojeni“ u razred Oblik3D (sve navedene razrede predstavlja razred Oblik3D uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:

# Zadaci 5 - 7

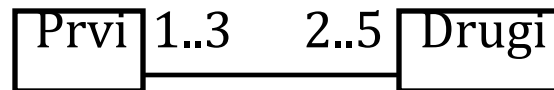
5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi Kocka, Kvadar, Valjak, Kugla i Stožac su „spojeni“ u razred Oblik3D (sve navedene razrede predstavlja razred Oblik3D uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:  
povećanje kohezije

# Zadaci 5 - 7

5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi `Kocka`, `Kvadar`, `Valjak`, `Kugla` i `Stožac` su „spojeni“ u razred `Oblik3D` (sve navedene razrede predstavlja razred `Oblik3D` uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:

**povećanje kohezije**

6. Ako u dijagramu razreda između razreda `Prvi` i `Drugi` postoji asocijacija (veza):



te se u sustavu (pri izvođenju) nađu tri objekta nastala iz razreda `Prvi`, koliko se najmanje i najviše objekata nastalih iz `Drugi` može naći u sustavu (uz pretpostavku poštivanja višestrukosti označenim na vezama)?

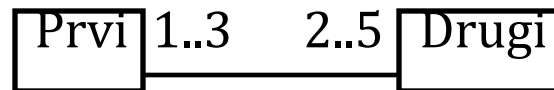
min = ?                      max = ?

# Zadaci 5 - 7

5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi Kocka, Kvadar, Valjak, Kugla i Stožac su „spojeni“ u razred Oblik3D (sve navedene razrede predstavlja razred Oblik3D uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:

**povećanje kohezije**

6. Ako u dijagramu razreda između razreda Prvi i Drugi postoji asocijacija (veza):



te se u sustavu (pri izvođenju) nađu tri objekta nastala iz razreda Prvi, koliko se najmanje i najviše objekata nastalih iz Drugi može naći u sustavu (uz pretpostavku poštivanja višestrukosti označenim na vezama)?

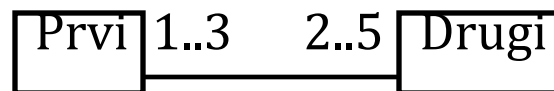
min = 2                      max = 15

# Zadaci 5 - 7

5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi Kocka, Kvadar, Valjak, Kugla i Stožac su „spojeni“ u razred Oblik3D (sve navedene razrede predstavlja razred Oblik3D uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:

**povećanje kohezije**

6. Ako u dijagramu razreda između razreda Prvi i Drugi postoji asocijacija (veza):



te se u sustavu (pri izvođenju) nađu tri objekta nastala iz razreda Prvi, koliko se najmanje i najviše objekata nastalih iz Drugi može naći u sustavu (uz pretpostavku poštivanja višestrukosti označenim na vezama)?

**min = 2                      max = 15**

7. Program izgrađen iz objektne arhitekture će biti brži ako ima:  
više / manje dinamičkih povezivanja.

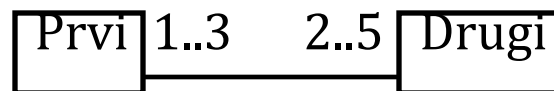


# Zadaci 5 - 7

5. U iterativnom postupku razvoja određene arhitekture programske potpore, razredi Kocka, Kvadar, Valjak, Kugla i Stožac su „spojeni“ u razred Oblik3D (sve navedene razrede predstavlja razred Oblik3D uz dodavanje atributa i adaptacijom metoda). Princip oblikovanja (jedan od 12) kojim se opisuju ovakvi postupci naziva se:

**povećanje kohezije**

6. Ako u dijagramu razreda između razreda Prvi i Drugi postoji asocijacija (veza):



te se u sustavu (pri izvođenju) nađu tri objekta nastala iz razreda Prvi, koliko se najmanje i najviše objekata nastalih iz Drugi može naći u sustavu (uz pretpostavku poštivanja višestrukosti označenim na vezama)?

**min = 2                      max = 15**

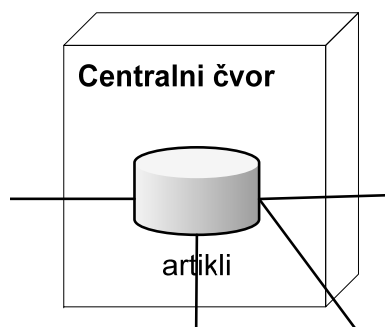
7. Program izgrađen iz objektne arhitekture će biti brži ako ima:  
više / manje dinamičkih povezivanja.

# Zadatak 8

- “Baza artikli se nalazi u centralnom računalu, a aplikacija unos koja koristi tu bazu nalazi se na računalima A, B, C i D. Program obračun nalazi se na računalu D i koristi zasebnu bazu zaposlenici koja se tamo nalazi.”  
Prikazati rečeno odgovarajućim UML dijagramom.

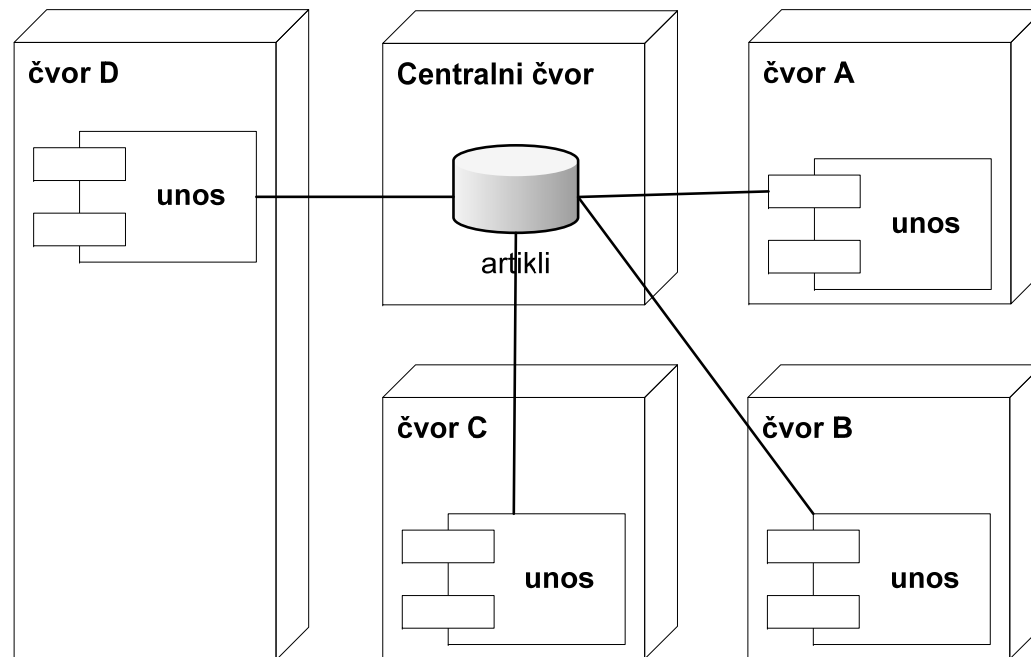
# Zadatak 8

- “Baza artikli se nalazi u centralnom računalu, a aplikacija unos koja koristi tu bazu nalazi se na računalima A, B, C i D. Program obračun nalazi se na računalu D i koristi zasebnu bazu zaposlenici koja se tamo nalazi.”  
Prikazati rečeno odgovarajućim UML dijagramom.



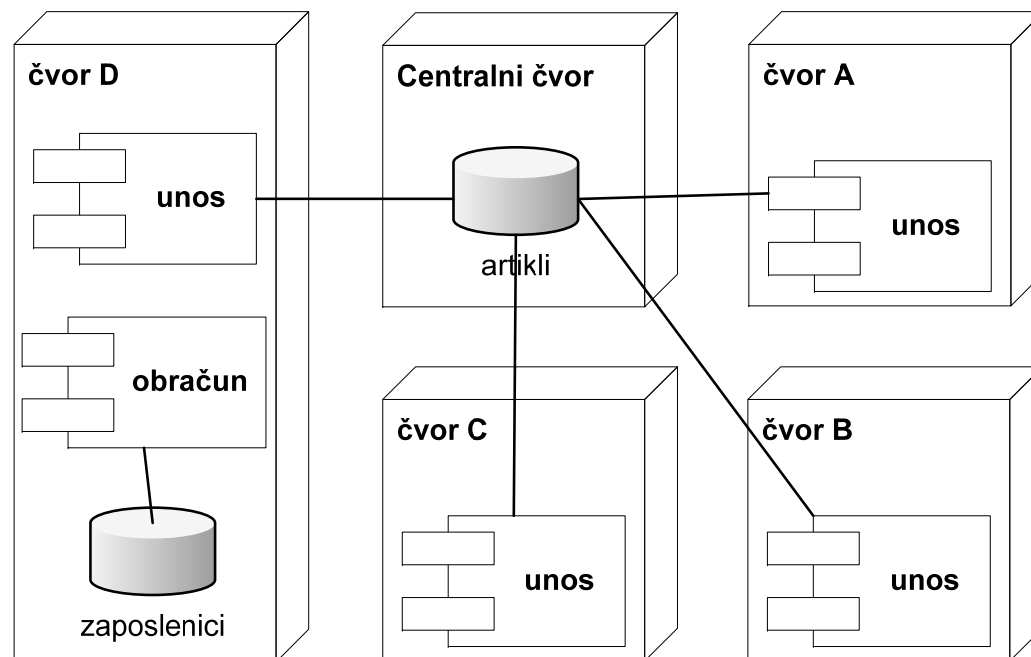
# Zadatak 8

- “Baza artikli se nalazi u centralnom računalu, a aplikacija unos koja koristi tu bazu nalazi se na računalima A, B, C i D. Program obračun nalazi se na računalu D i koristi zasebnu bazu zaposlenici koja se tamo nalazi.”  
Prikazati rečeno odgovarajućim UML dijagramom.



# Zadatak 8

- “Baza artikli se nalazi u centralnom računalu, a aplikacija unos koja koristi tu bazu nalazi se na računalima A, B, C i D. Program obračun nalazi se na računalu D i koristi zasebnu bazu zaposlenici koja se tamo nalazi.”  
Prikazati rečeno odgovarajućim UML dijagramom.



# Zadatak 9

- Dio nekog problema riješen je na slijedeći način:

```
cat popis.txt | grep ZI | sort | analiza > rez.txt
```

Koji standardni tip arhitekture je korišten?

**protok podataka, cjevovodi i filtri**

# Zadatak 10

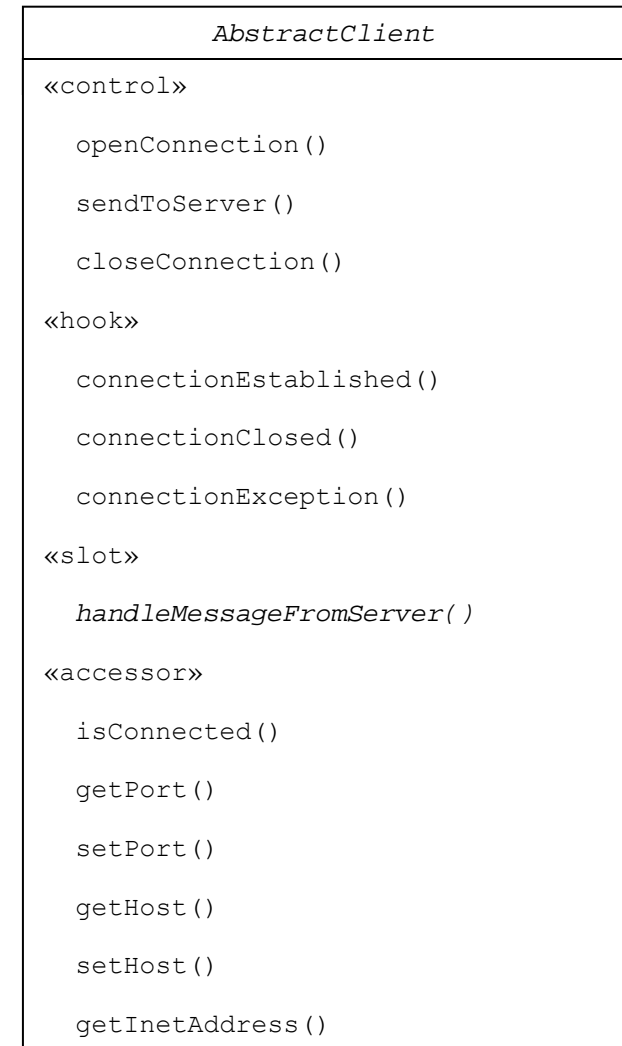
- Korištenjem *OCSF* (*Open Client Server Framework*) izgrađen je dio klijentske aplikacije, tj. razred `TClient` koji je nastao tako da je:

---

te je implementirana metoda:

---

Pokažite fragmentom kôda kako se razred koristi da bi se poslužitelju koji sluša na adresi 10.0.0.33:12345 poslala poruka sadržaja „test“.



# Zadatak 10

- Korištenjem *OCSF* (*Open Client Server Framework*) izgrađen je dio klijentske aplikacije, tj. razred `TClient` koji je nastao tako da je:  
\_\_ naslijeđen razred *AbstractClient* \_\_  
te je implementirana metoda:  
\_\_\_\_\_.

Pokažite fragmentom kôda kako se razred koristi da bi se poslužitelju koji sluša na adresi 10.0.0.33:12345 poslala poruka sadržaja „test“.

<i>AbstractClient</i>
«control»
openConnection()
sendToServer()
closeConnection()
«hook»
connectionEstablished()
connectionClosed()
connectionException()
«slot»
handleMessageFromServer()
«accessor»
isConnected()
getPort()
setPort()
getHost()
setHost()
getInetAddress()



# Zadatak 10

- Korištenjem *OCSF* (*Open Client Server Framework*) izgrađen je dio klijentske aplikacije, tj. razred `TClient` koji je nastao tako da je:  
\_\_ naslijeđen razred *AbstractClient* \_\_  
te je implementirana metoda:  
\_\_ *handleMessageFromServer* \_\_.

Pokažite fragmentom kôda kako se razred koristi da bi se poslužitelju koji sluša na adresi 10.0.0.33:12345 poslala poruka sadržaja „test“.

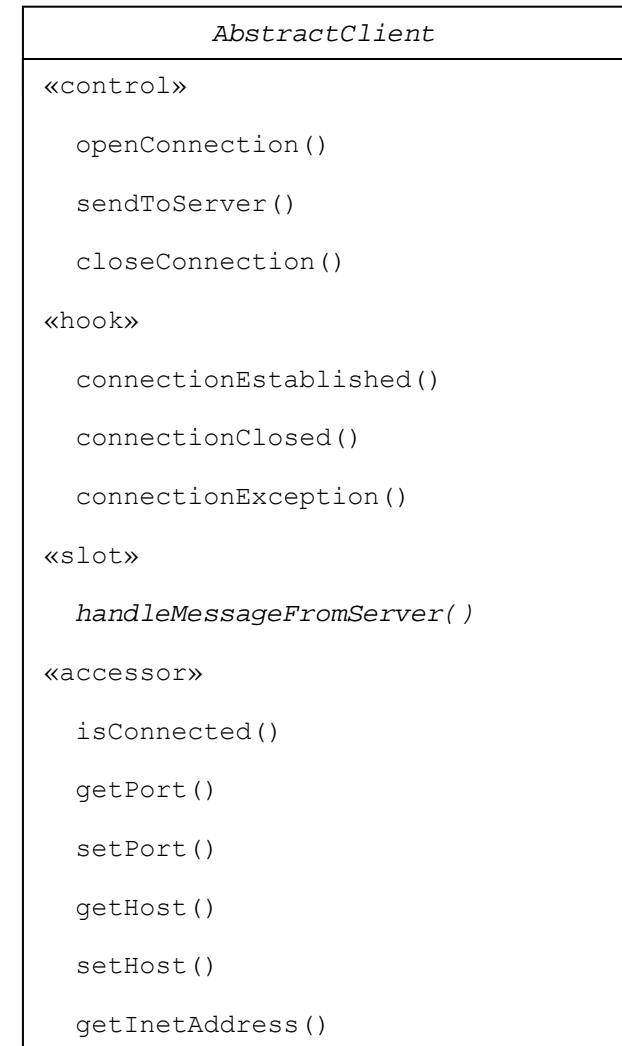
<i>AbstractClient</i>
«control»
openConnection()
sendToServer()
closeConnection()
«hook»
connectionEstablished()
connectionClosed()
connectionException()
«slot»
<i>handleMessageFromServer()</i>
«accessor»
isConnected()
getPort()
setPort()
getHost()
setHost()
getInetAddress()

# Zadatak 10

- Korištenjem *OCSF* (*Open Client Server Framework*) izgrađen je dio klijentske aplikacije, tj. razred `TClient` koji je nastao tako da je:  
\_\_ naslijeđen razred *AbstractClient* \_\_  
te je implementirana metoda:  
\_\_ *handleMessageFromServer* \_\_.

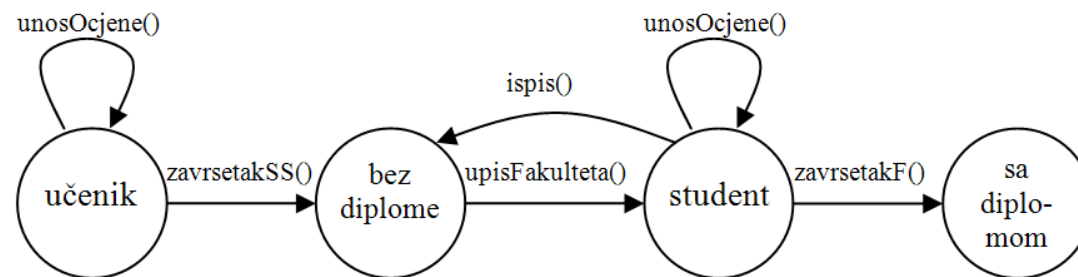
Pokažite fragmentom kôda kako se razred koristi da bi se poslužitelju koji sluša na adresi 10.0.0.33:12345 poslala poruka sadržaja „test“.

```
TClient a („10.0.0.33“, 12345);  
a.openConnection();  
a.sendToServer („test“);
```



# Zadatak 11

- Za razred `StatusOsobe` zadan je dijagram stanja prema slici, gdje su prijelazi uzrokovani pozivima metoda razreda.



- Kako oblikovati testni slučaj?
- Navedite ga.

a) Tako da se prođu sva stanja i svi prijelazi.

b) `unosOcjene` → `zavrsetakSS` → `upisFakulteta` → `ispis` → `upisFakulteta` → `unosOcjene` → `zavrsetakF`

ili

`unosOcjene` → `zavrsetakSS` → `upisFakulteta` → `unosOcjene` → `ispis` → `upisFakulteta` → `zavrsetakF`

# Zadatak 12

- Rečenicu „Najveći miš se ne boji najmanje mačke“ preslikati u predikatnu logiku koristeći predikate:

*miš*(*X*),

*mačka*(*X*),

*boji*(*X*, *Y*) (*X* se boji *Y*),

*najmanji*(*X*),

*najveći*(*X*).

$$\forall X \forall Y ( (\text{miš}(X) \wedge \text{najveći}(X) \wedge \text{mačka}(Y) \wedge \text{najmanji}(Y)) \Rightarrow \neg \text{boji}(X, Y) )$$

može i:

$$\exists X \exists Y (\text{miš}(X) \wedge \text{najveći}(X) \wedge \text{mačka}(Y) \wedge \text{najmanji}(Y) \wedge \neg \text{boji}(X, Y) )$$

$$\forall X \forall Y (\text{miš}(X) \wedge \text{najveći}(X) \Rightarrow (\text{mačka}(Y) \wedge \text{najmanji}(Y) \wedge \neg \text{boji}(X, Y) )$$

# Zadatak 13

- U CTL-u prikazati rečenicu „Svjetlo može biti ili upaljeno ili ugašeno“ (ex-ili) korištenjem (oba) atoma: *upaljeno* i *ugašeno*.

$AG((upaljeno \wedge \neg ugašeno) \vee (\neg upaljeno \wedge ugašeno)),$

može i:

$AG (A(upaljeno \ U \ ugašeno) \vee A(ugašeno \ U \ upaljeno))$

ako je izraz polovični, recimo:

$AG(upaljeno \vee ugašeno),$

ili ako nedostaje AG, pa ima samo:

$(upaljeno \wedge \neg ugašeno) \vee (\neg upaljeno \wedge ugašeno)$

ili :

$EG((upaljeno \wedge \neg ugašeno) \vee (\neg upaljeno \wedge ugašeno)),$

$AG (AX(upaljeno \ U \ ugašeno) \vee AX(ugašeno \ U \ upaljeno))$

POLOVIČNO!

# Zadatak 14

- Kripke struktura zadana je grafom.  
Za koja stanja grafa vrijede formule:  
a)  $AG(uči \Rightarrow AF(položio))$   
b)  $E(uči \text{ U } položio)$

a)  $S_2$

b)  $S_2$

