

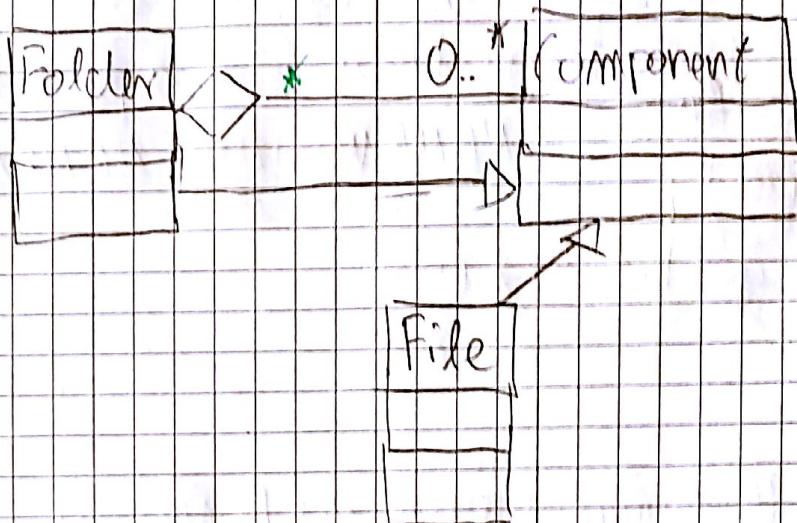
AUDITI OPNE VJEŽBE

50%

60%

AI → 3-4 teorijska (predavanje) + 2-3 zadatka

① Dati je sljedeći dizajn kojim su modelirani pojmovi i
faktora u nekom OS.



Želimo u klasi Folder funkcionalnost za izračunavanje uk.
veličine svih fajlova i foldera unutar tog foldera.

- * folder sadrži listu komponenti / listu referenci na komponente jer se mogu nalaziti i drugi folderi, a ne samo fajlovi
- u folder moramo dodati neku funkciju getSize za računanje veličine svih fajlova i foldera unutar tog foldera

```
class Folder {
    List<Component> comp; // popis fajlova i foldera
    int getSize() { ukVel = 0;
        foreach (Component c in comp) ukVel += c.getSize();
    }
```

→ switch-case problem koji eliminišemo
POLIMORFIZMOM

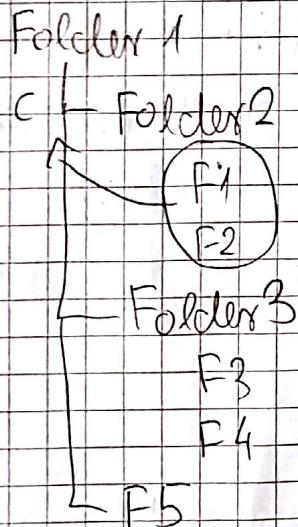
```

interface Component {
    long getSize(); → nasljeduje Folder
}

class File { → dodali smo abstractnu funkciju
    FILEHANDLE fh; koju klasa definira, u Component
    long getSize() { NEMAMO IMPLEMENTACIJU
        return getFileSize(fh);
    }
}

```

- unutar komponente stavimo VIRTUALNU FUNKCIJU
getFileSize



- (2) - funkcionalnost za kriptiranje datoteke, opšt., XYZ¹¹
 - implementirano ver.

```
class EncryptFile {
    void encryptFile(string inFile Name, string
out File Name), → učitamo dat. u mem., kriptiramo i snimimo
    na disk
```

- aplikacija se počela rušiti zbog nedostatka memorije
 => developeri uočili da treba implementirati umanjiju varijantu kriptiranja kojo sa primjenjuju za velike datoteke

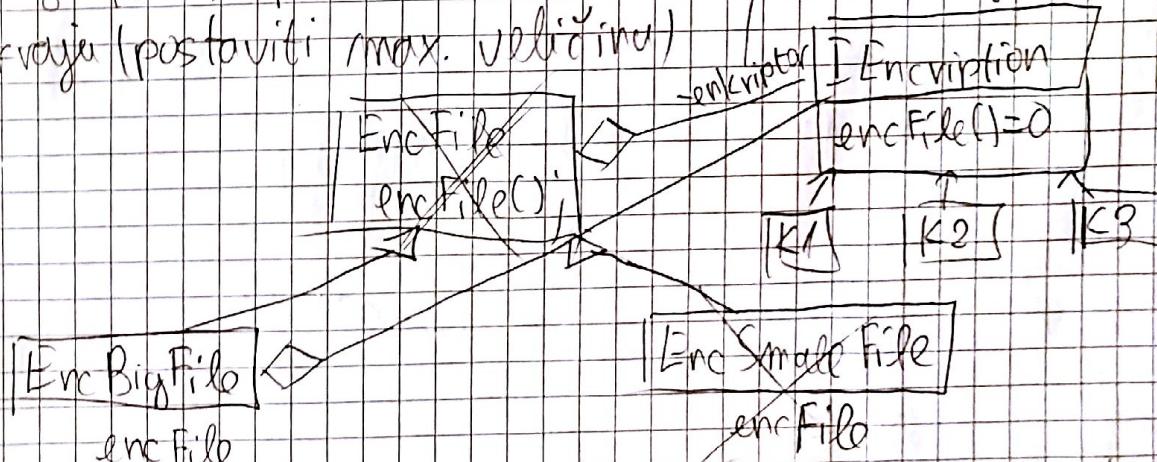
- PITANJE: Kako modificirati dizajn „XYZ“ aplikacije, a da se navedena promjena ugradbi uz min. problem i promjene u klijenu (zbog performansi k. vješ. je i dalje prisutno, a nije nemoguće doći do promjene s dod. načinima kriptiranja)

članstva vam.

MOGUĆA RJEŠENJA:

I podlježe kod kriptiranoga dijela po dio datoteke i spajamo na kraju (postaviti max. veličinu)

II



U ovoj funkciji uzmemmo komad velikog fajla i dajemo encFile u EncSmallFile

→ funkciju za enkripciju koristimo i mijestu

→ na kraju spajamo

- zaukljucujemo da će stvar koči dok smo novi obj. za enkripciju

=> INTERFACE

encFile() { (Small)

 enkriptir. encryptFile();
}

~~referenca, tj. članska varijabla na algoritam
za kriptiranje~~

encFile() { (Big)

 koliki je fajl

 koliko ima memorije

- for petlja da polazimo fajl (ako jo proveliko) }

→ ispostavlja da nam treba samo 1 klasi koja
potraživa i male i velike datoteke pa nam nije
potrebno za mala dat.

⇒ dobivamo UNIVERZALNO RJEŠENJE

ali zmati ŠTO MANJE KLASA

③ class AnalysisData

{ // klasa koja modelima rezultate analize loga }

class LogFileProcessor {

public void processLog (string filePath);

private string getContents ();

private AnalysisData performAnalysis (string logContents);

private void saveToDatabase (AnalysisData infoData);

}

void LogFileProcessor::processLog (string filePath) {

string log = getContents (filePath);

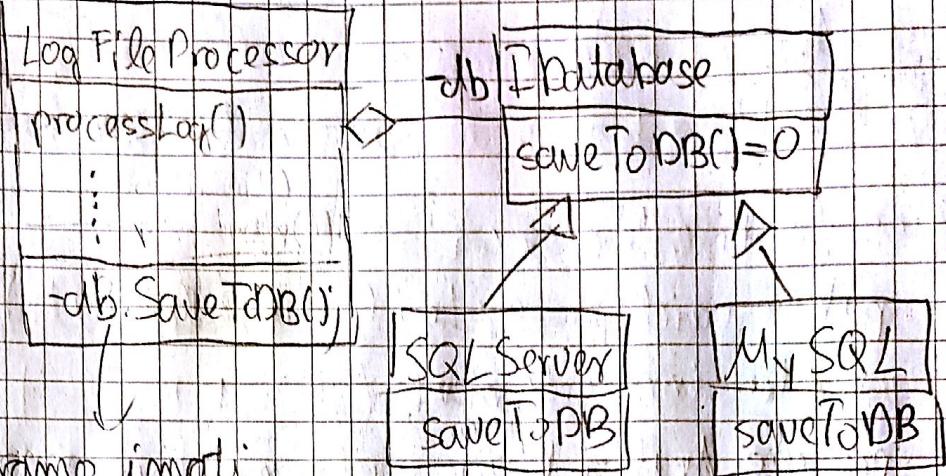
AnalysisData data = performAnalysis (log);

saveToDatabase (data);

}

Kako modifikaciji dizajn do modela raditi i s MySQL uz
već postojeci rad s SQL Serverom?

PJ ↗



monadno imati
referencu na objekt te samo pozovemo potrebnu
funkciju

Zamjenjujemo tehničke značajke, bitno je da znamo
ispuniti s vlastnicima funkcionalnost, definirati
zajedničko mjesto koje prosinjujemo traženim
implementacijama

Strategy Pattern:

MI 2013 / (8.) (zadatok sa sedm zornom) → Observer

- dobro naučiti Strategy Pattern, NAVČESE se pojavljuje

```
class Senzor {
    foreach(Criterium c in listOfUserCriterias)
        void OnReadData (SensorData inData) { }
```

```
if (c.mitemiaEvaluator(inData, criterium))
    SendNotification();
```

moguće postojati više

prodstavlja kriterij → uviđa da li smo se

```
class GreaterThan {
    long _val;
    float - val;
```

1 notifikacija

```
bool isSatisfied (SensorData inData)
```

- naci id u listi
- ako nema → return false

class LessThan, Equal, ...

ako ima => VR = Vrednost
return VR > val;

SensorData {

long id;

float moistVal;

}

interface Criteria {

bool isSatisfied(SensorData inData);

class GreaterThan {

long id;

float val;

}

- trebamo imati neko ugovoreno postaviti ih:

→ poslati mail:

Temp > 35 °C

Temp > 30 °C && Tlak > 1.5 atm

id(1)

, id(0)

→ is SensorData

- zanimanjivo da postoji više korisnika, većno je da postoji 1.

- možemo imati listu kriterijija i za svaki kriterij posebnu implementaciju

- mogemo kriterijevi da su kompleksne operacije

class ComplexCriteria AND {

(Criteria - c1, Criteria - c2,

bool isSatisfied (SensorData dt) {

bool b1 = c1.isSatisfied();

bool b2 = c2.isSatisfied();

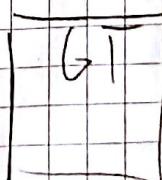
return b1 && b2;

}

Temp > 35 AND Tlak > 1.5



AND



$S_1 > 30 \text{ AND } S_2 \leq 1000 \text{ OR } S_3 = 25 \text{ AND } S_4 \leq 5$

Complex AND

GT
 $S_1 > 30$
AND
 $S_2 \leq 1000$

LT
OR

Complex AND

EQ
 $S_3 = 25$
AND
 $S_4 \leq 5$

Complex XOR

—dosta složen zad (vjerovatno neće na 11)