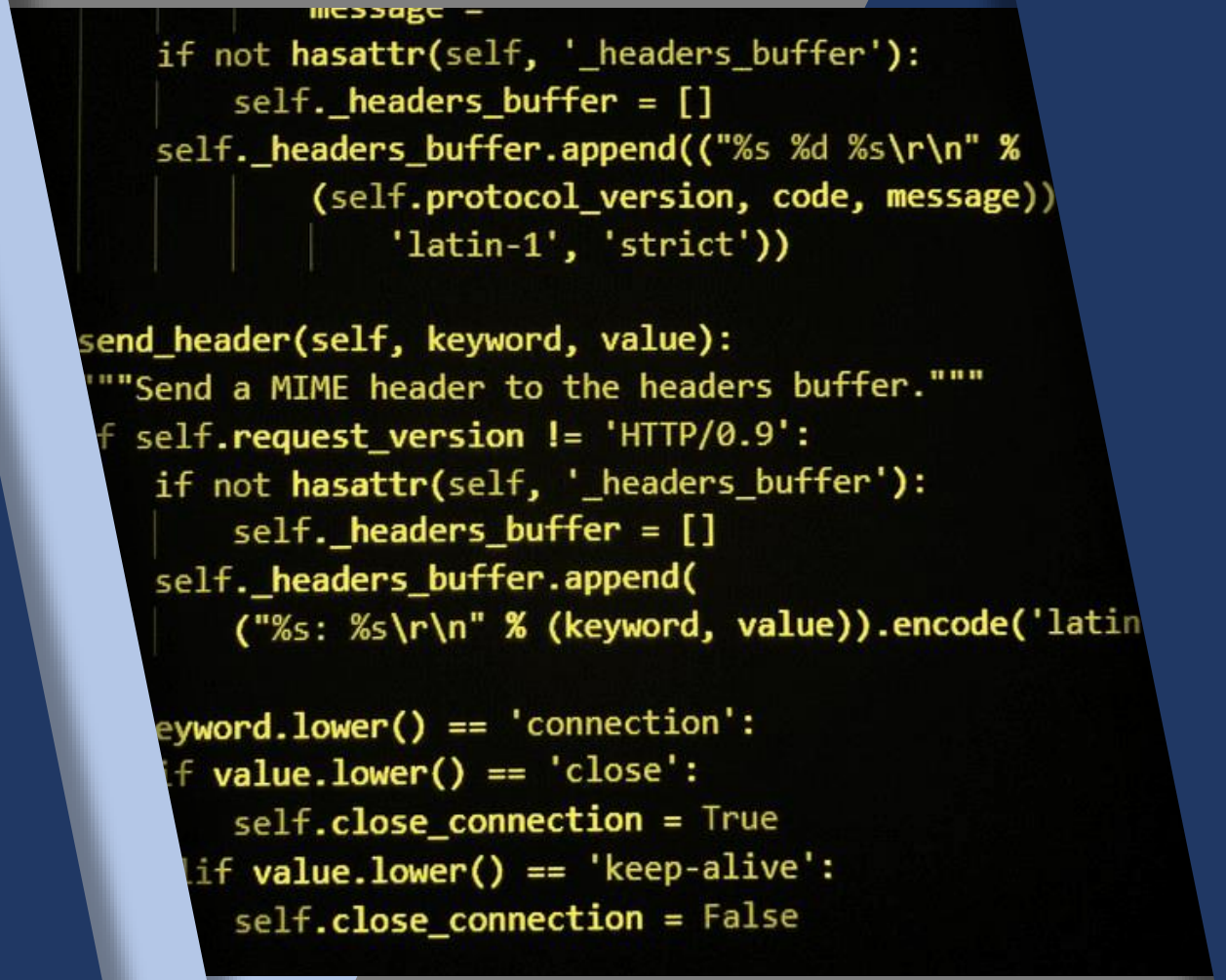
Specifikacija i modeliranje softvera

Čist dizajn koda V

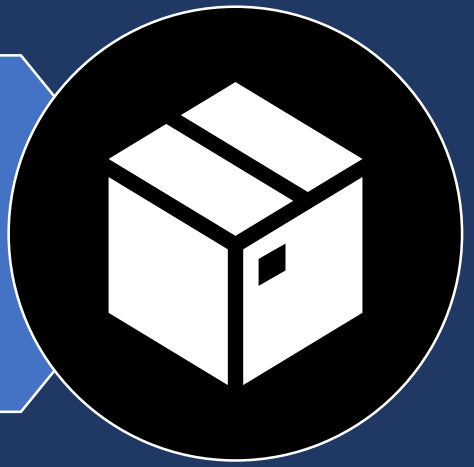
Nikola Luburić
nikola.luburic@uns.ac.rs



Čist dizajn koda V

Šta govore principi čistog dizajna?

Formatiranje i paketiranje koda u datotekama?



Značaj čistog koda i primena u praksi?

Čist dizajn koda V

SOLID Principi čistog dizajna?

Šta je SOLID?

- **S**ingle Responsibility Principle
- **O**pen-Closed Principle
- **L**iskov Substitution Principle
- **I**nterface Segregation Principle
- **D**ependency Inversion Principle

Šta nije SOLID?

- Gospodar
- *Framework*, biblioteka, šablon
- *Technology-specific*

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Skupi stvari koje se menjaju iz istih razloga
Razdvoj one koje se menjaju iz različitih

```
Employee
+ calculatePay()    : double
+ save()            : void
+ reportEmployee() : String
+ findById(int id) : Employee
```

broj odgovornosti?

što je problem?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Skupi stvari koje se menjaju iz istih razloga
Razdvoj one koje se menjaju iz različitih

```
Employee
+ calculatePay()    : double
+ save()            : void
+ reportEmployee()  : String
+ findById(int id)  : Employee
```

kako raspakovati da prati SRP?

Šta bi uzrokovalo promenu ove klase?

- Izmena skladišta podataka
- Izmena izveštavanja
- Izmena računice plate

ko naručuje ove promene?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Skupi stvari koje se menjaju iz istih razloga

Razdvoj one koje se menjaju iz različitih

šta menja nova baza?

```
Employee
```

```
EmployeeRepository
+ save(Employee e) : void
+ findById(int id) : Employee
```

```
PaycheckCalculator
+ calculatePay(Employee e) : double
```

šta menja nov izveštaj?

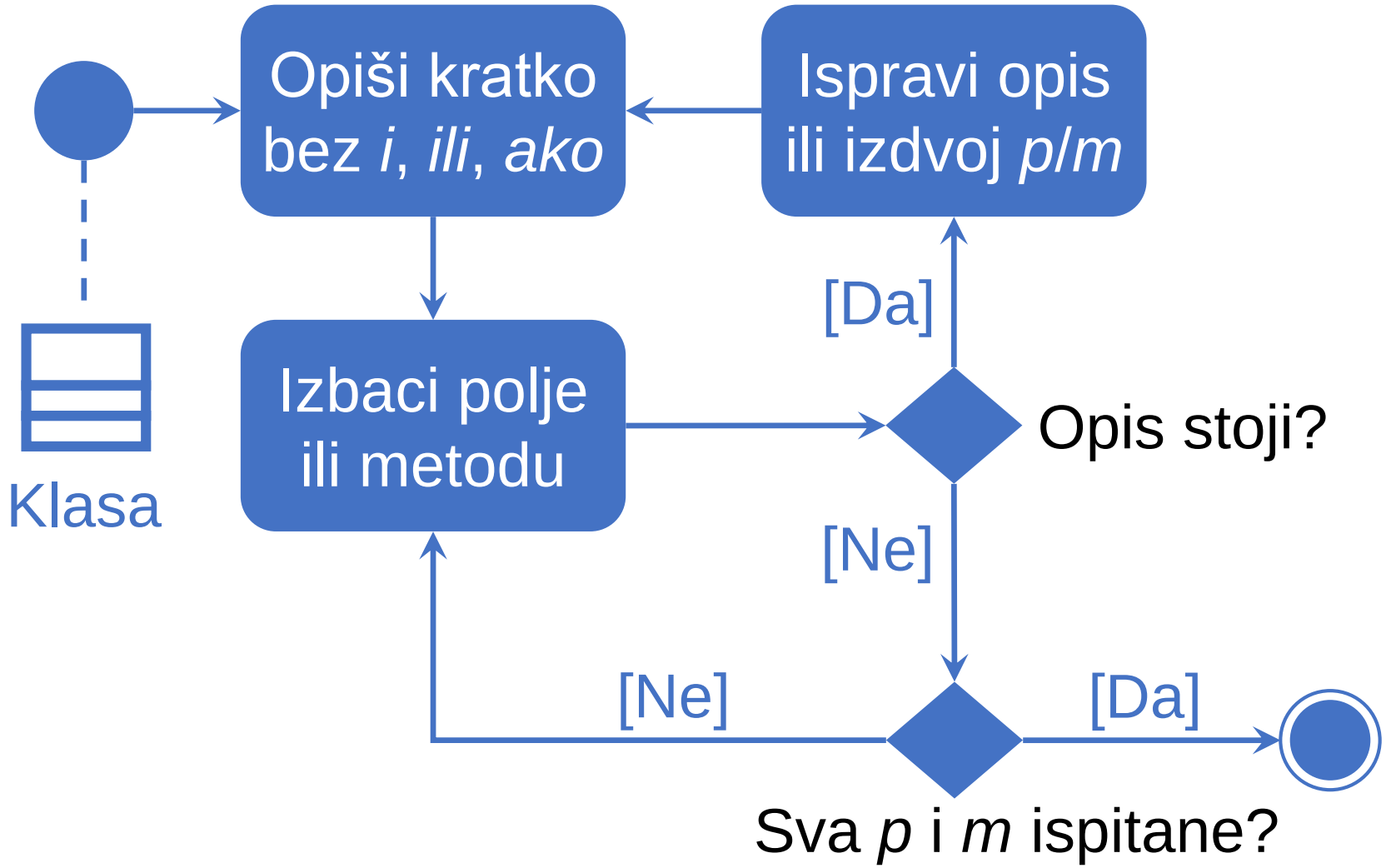
```
EmployeeReporter
+ report(Employee e) : String
```

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Skupi stvari koje se menjaju iz istih razloga
Razdvoj one koje se menjaju iz različitih



Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Skupi stvari koje se menjaju iz istih razloga
Razdvoj one koje se menjaju iz različitih

SRP je primenljiv na razne apstrakcije

- Klase
- Paketi
- Mikroservisi

SRP donosi više prednosti

- Povećana čitljivost
- Olakšane izmene
- Manja šansa za runtime greške

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

```
class EmployeeReporter {
 void reportEmployees() {
  List<Employee> ems = new ArrayList<>();
  String query = "select * from Employees";
  try {
   Statement stmt = con.createStatement();
   ResultSet rs = stmt.executeQuery(query);
   while (rs.next())
    ems.push(compileEmployee(rs));
   printEmployees(ems);
  } catch(Exception e) {...}
 }}
```

DIP

odakle da počnemo?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

```
class EmployeeReporter {
  void reportEmployees() {
    EmployeeDB edb = new EmployeeDB();
    printEmployees(edb.getEmployees());
  }
}
```

Znak da se DIP ne prati?

da li Reporter zavisi od DB?

DIP

```
class EmployeeReporter {
  void reportEmployees(IEmployeeRepo er) {
    printEmployees(er.getEmployees());
  }
}
```

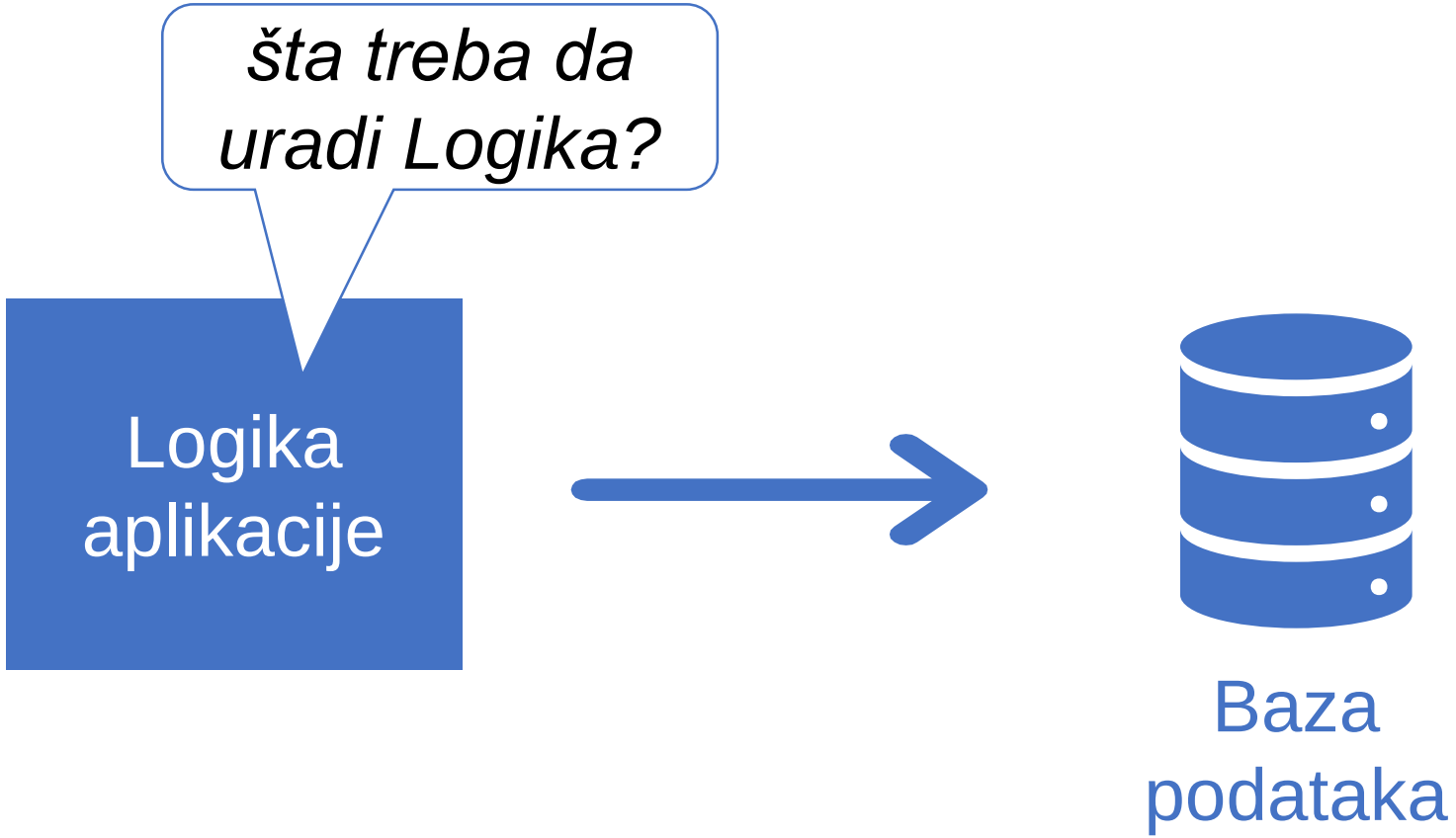
šta su prednosti?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



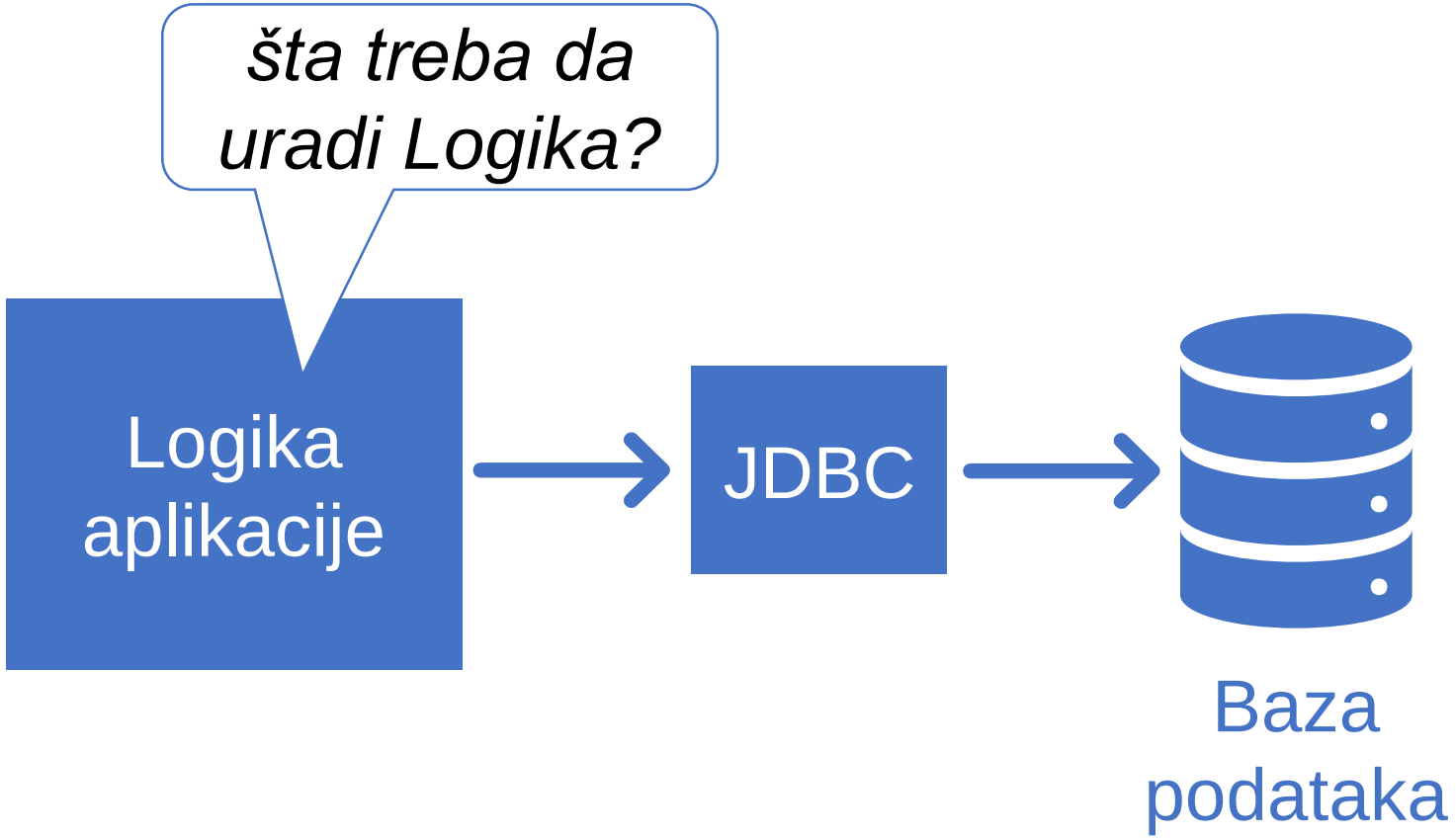
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



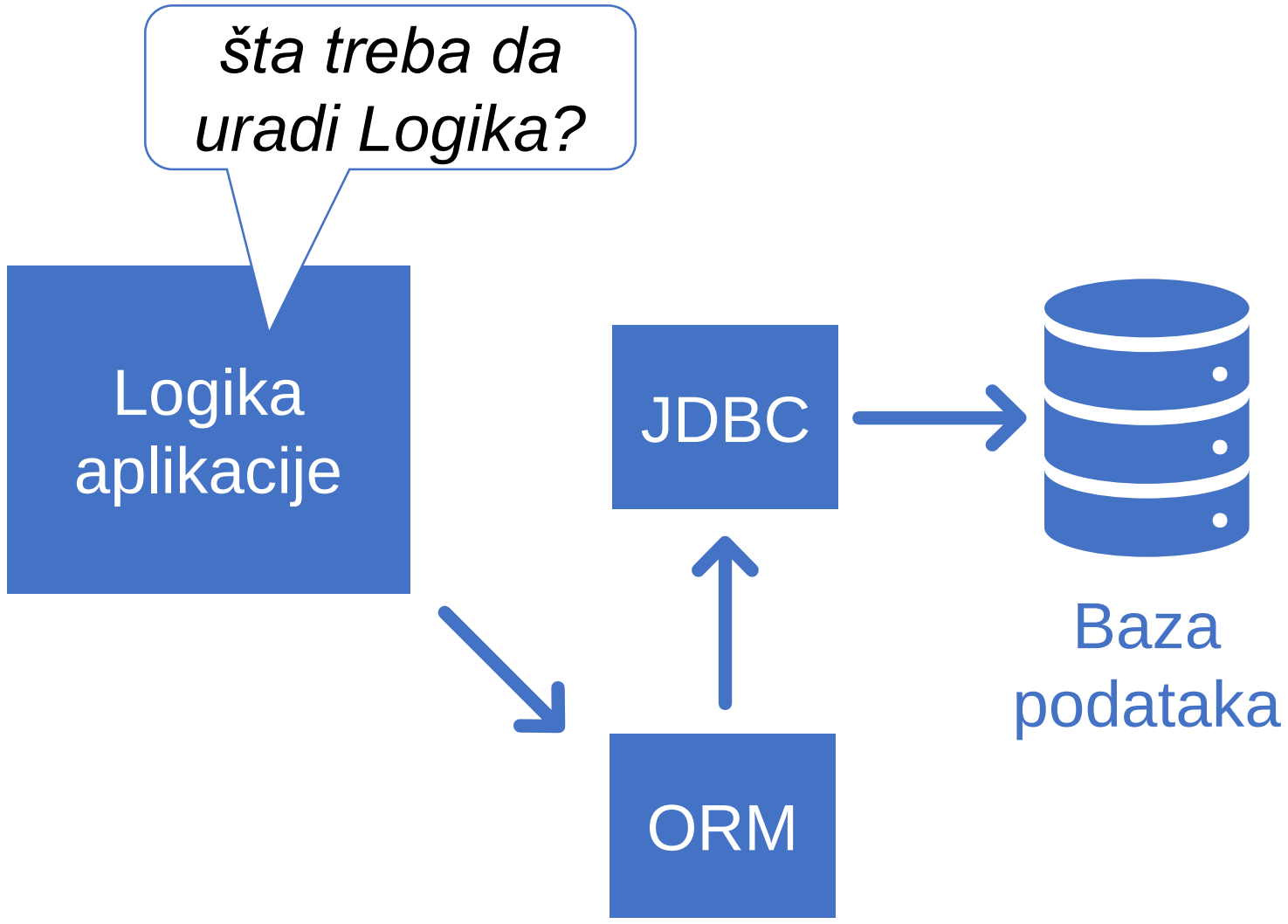
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



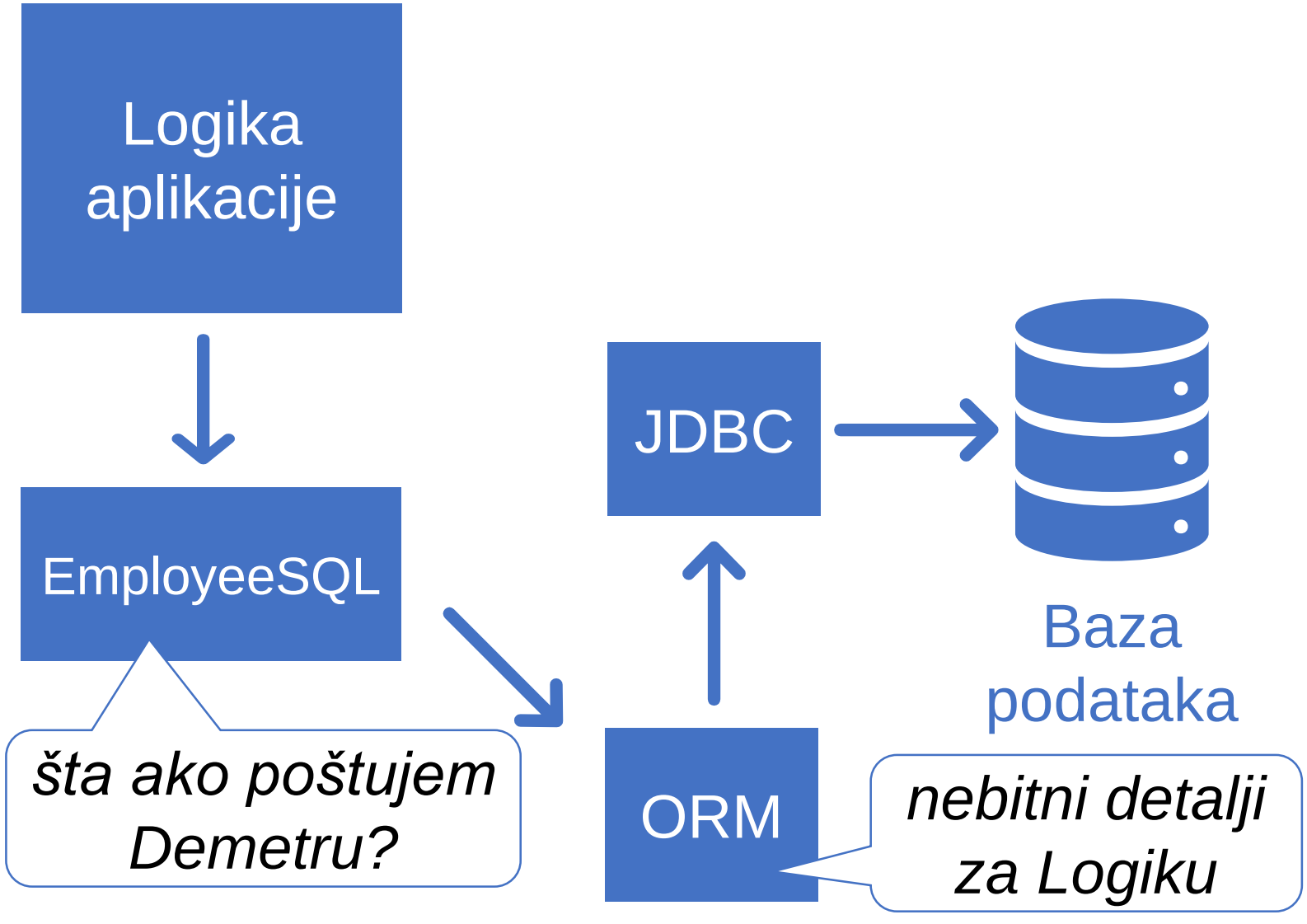
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



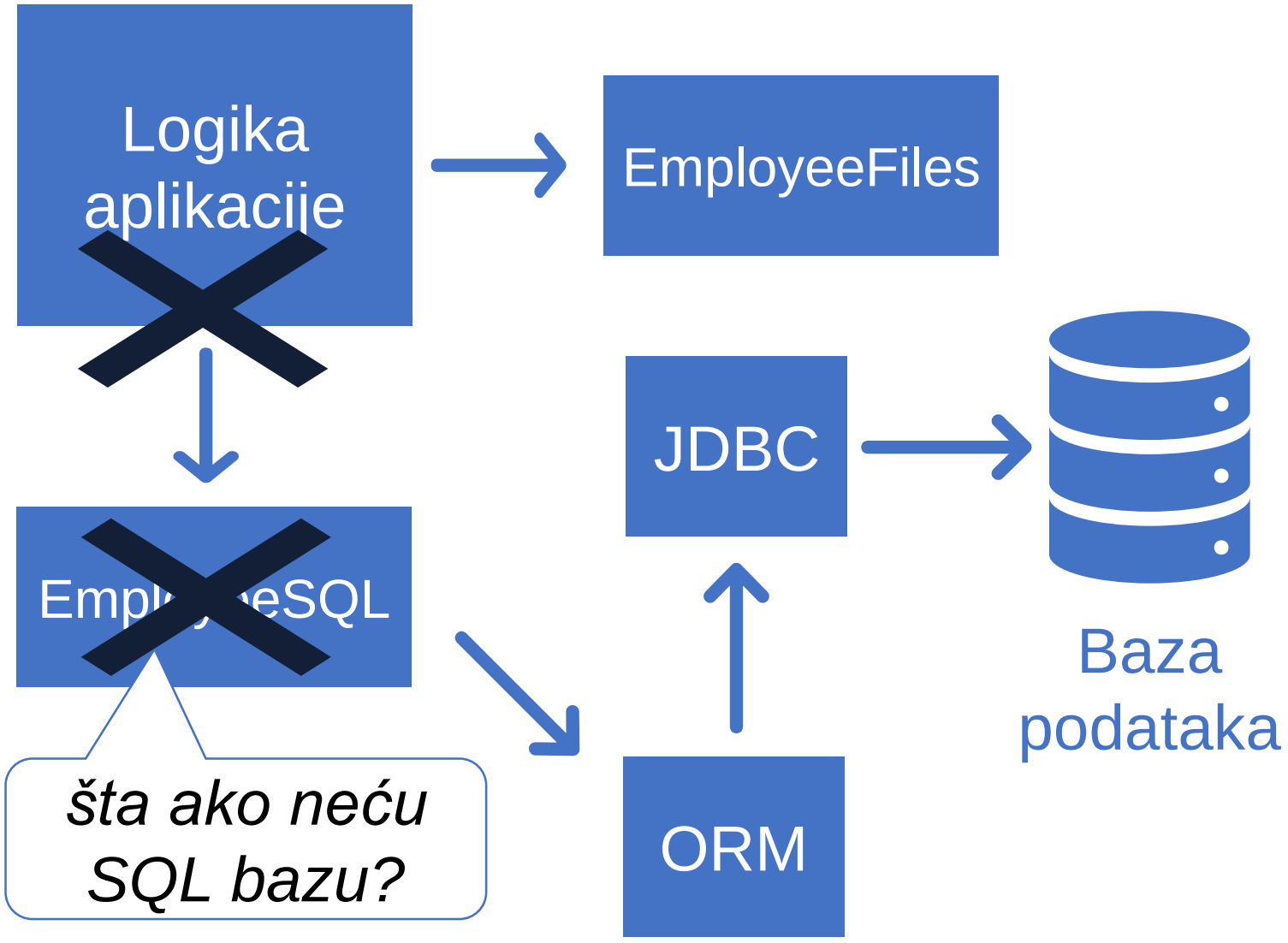
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



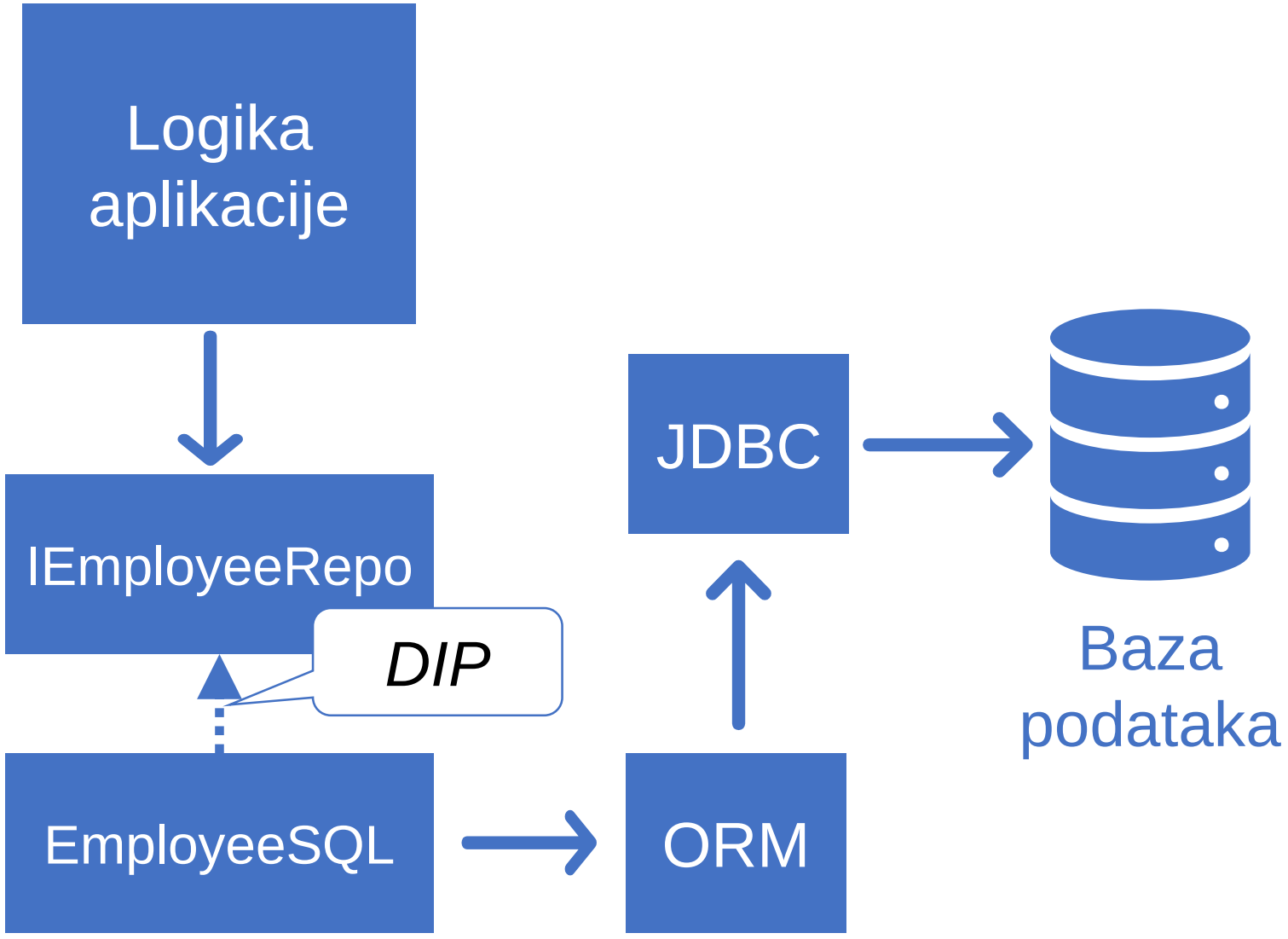
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

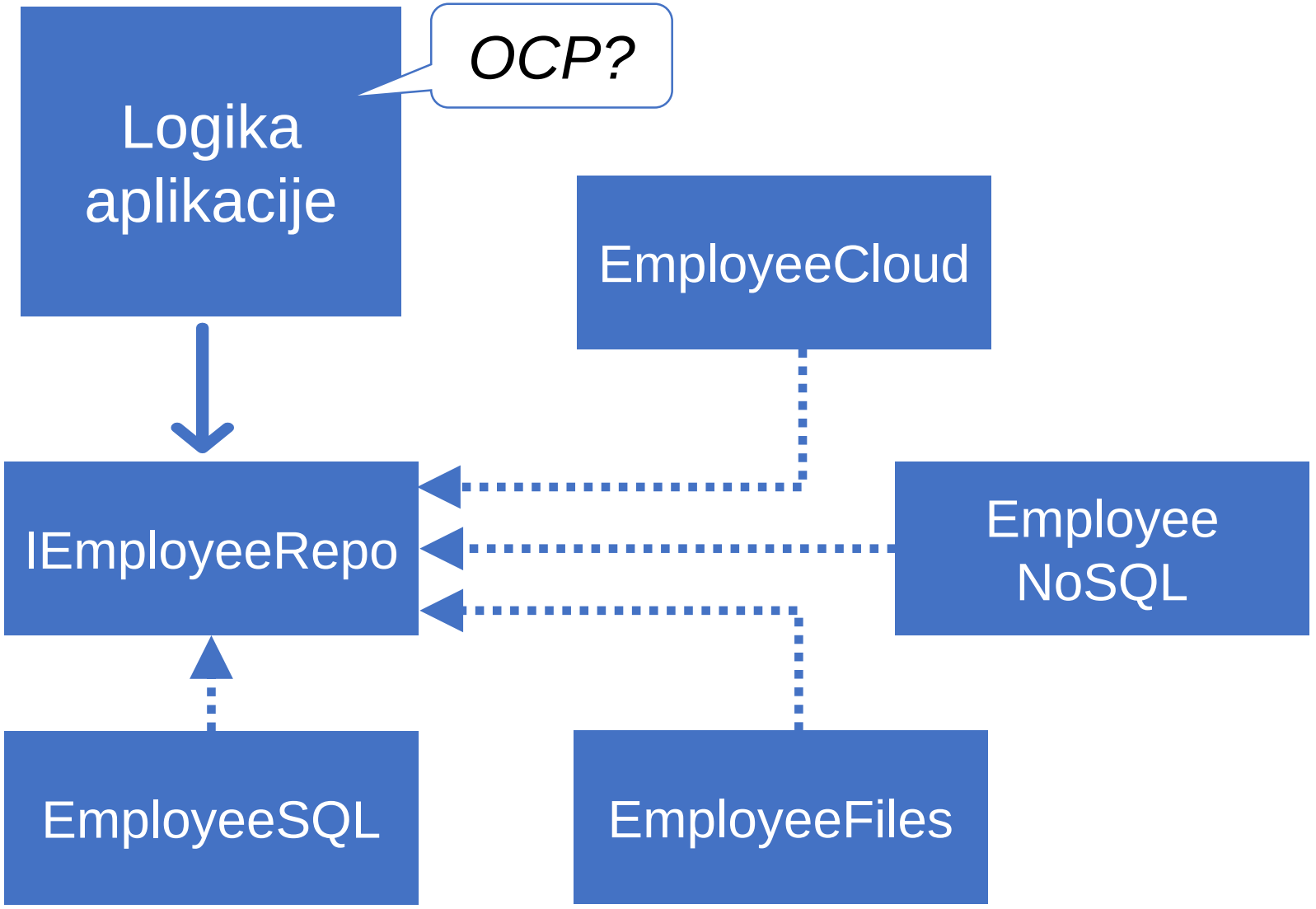


Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



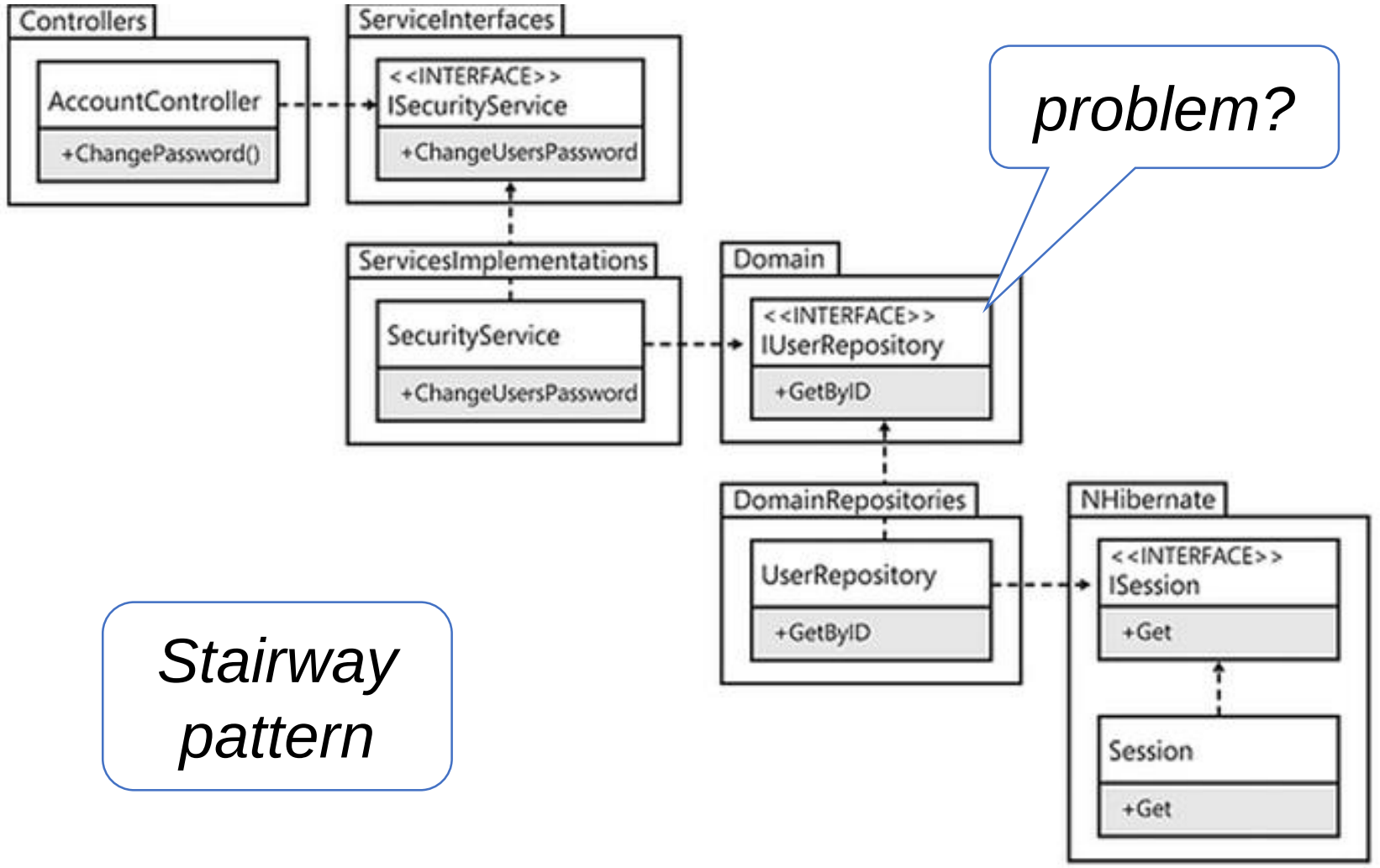
DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



Stairway pattern

DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

Dependency injection

- Šablon koji pomaže da se ostvari *DIP*
- Opskrbi modul sa onim što mu je potrebno
- Nivo konstruktora, polja (*setter*) i funkcije

DIP

```
class EmployeeReporter {
  void reportEmployees(IEmployeeRepo er) {
    printEmployees(er.getEmployees());
  }
}
```

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

Dependency injection

- Šablon koji pomaže da se ostvari *DIP*
- Opskrbi modul sa onim što mu je potrebno
- Nivo konstruktora, polja (*setter*) i funkcije

```
class EmployeeReporter {
  IEmployeeRepo er;
  public EmployeeReporter(IEmployeeRepo e) {
    this.er = e;
  }
}
```

DIP

Čist dizajn koda V

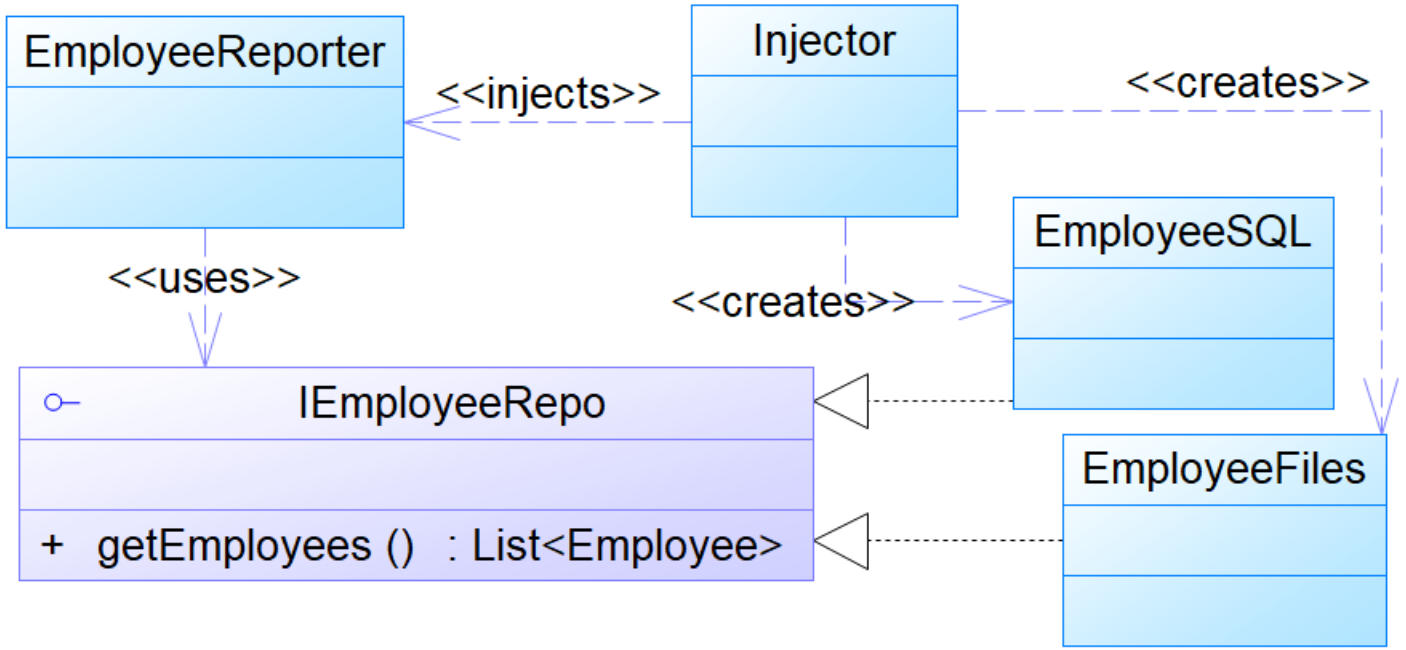
SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

Dependency injection

- Šablon koji pomaže da se ostvari *DIP*
- Opskrbi modul sa onim što mu je potrebno
- Nivo konstruktora, polja (*setter*) i funkcije



DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

DIP omogućuje *loose coupling*

- Među klasama
- Među komponentama

DIP i DI su osnovni mehanizmi za ono što zovemo proširivost u OO jezicima

DIP

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

DIP

Ponašanje modula je moguće menjati bez da se menja njegov izvorni kod

```
public class Logger {
  void Log(String message, String logType) {
    switch(logType) {
      case “Console”:
        System.out.println(message);
        break;
      case “File”:
        // Code to write message to file
        break;
    }
  }
}
```

nov zahtev traži zapis poruke u bazu

Čist dizajn koda V

SOLID Principi čistog dizajna?

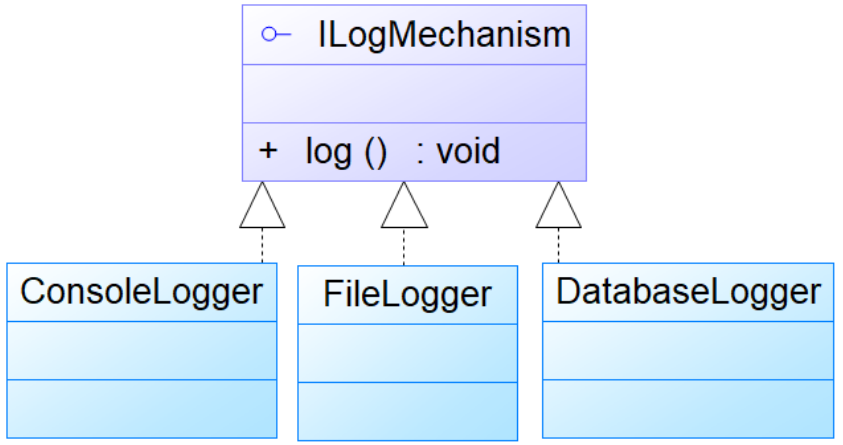
SRP

OCP

DIP

Ponašanje modula je moguće menjati bez da se menja njegov izvorni kod

```
public class Logger {
  private ILogMechanism logMechanism;
  public Logger(ILogMechanism lm) {
    this.logMechanism = lm;
  }
  public void Log(String message) {
    this.logMechanism.log(message);
  }
}
```



Čist dizajn koda V

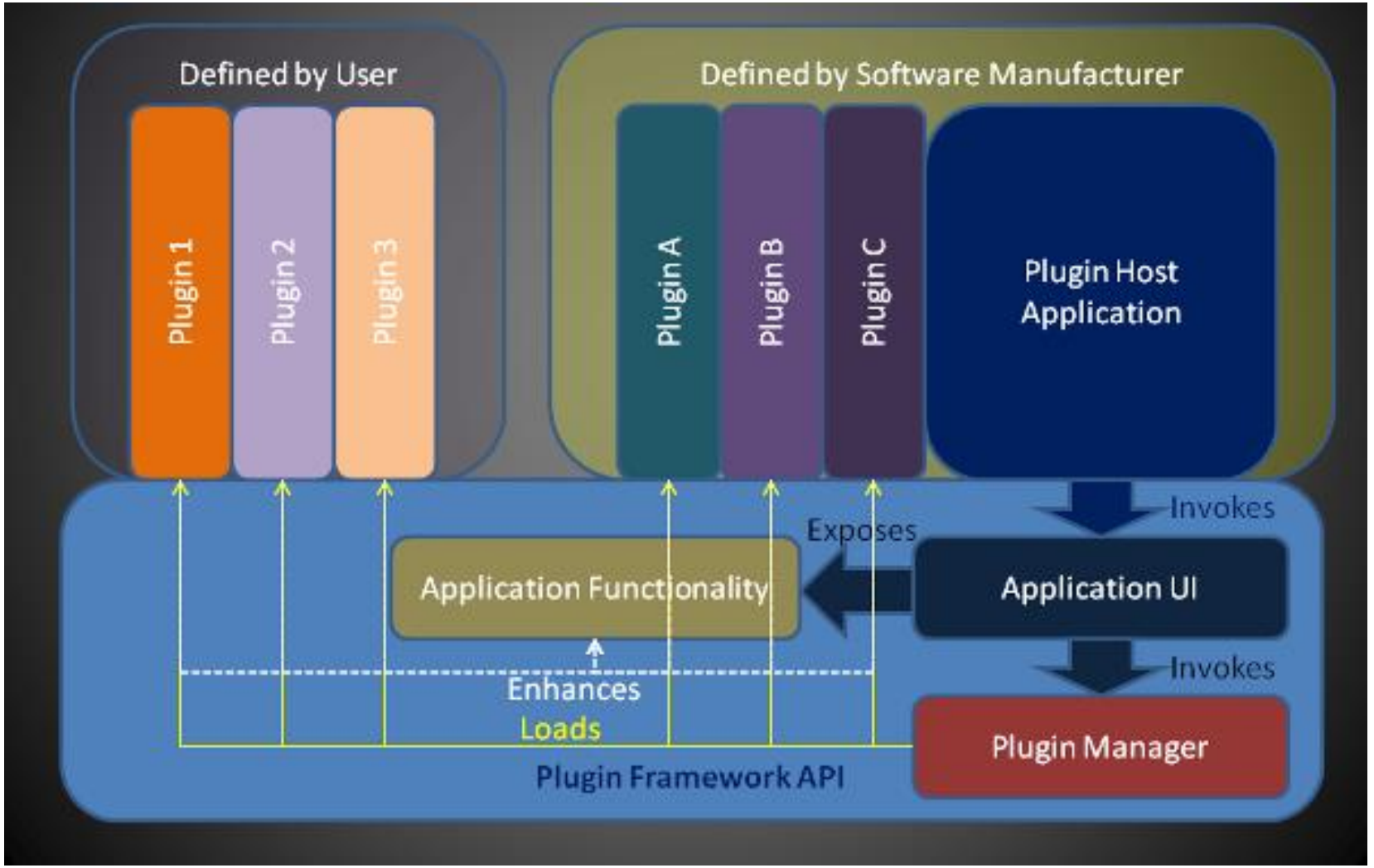
SOLID Principi čistog dizajna?

SRP

OCP

DIP

Ponašanje modula je moguće menjati bez da se menja njegov izvorni kod



Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

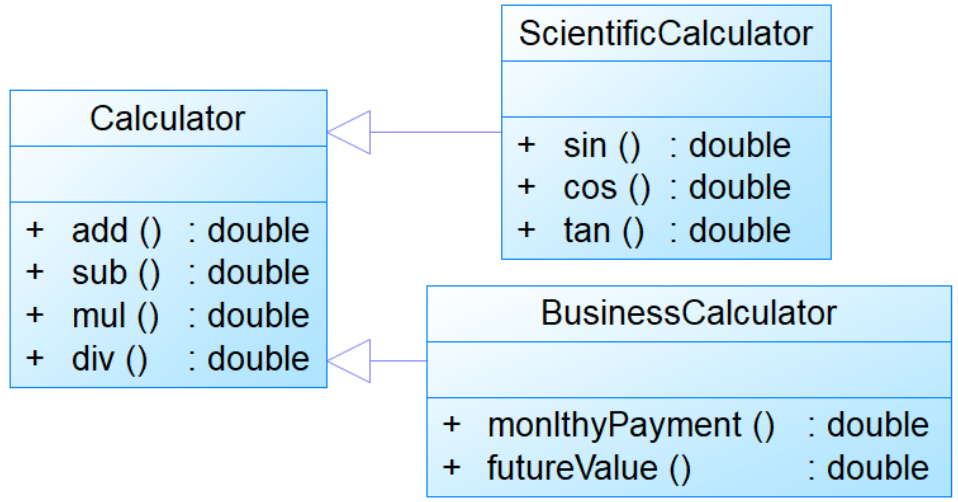
OCP

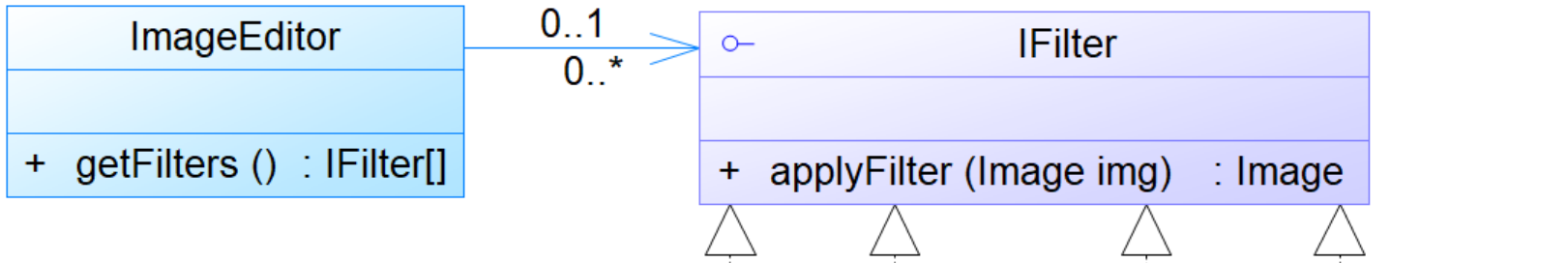
DIP

Ponašanje modula je moguće menjati bez da se menja njegov izvorni kod

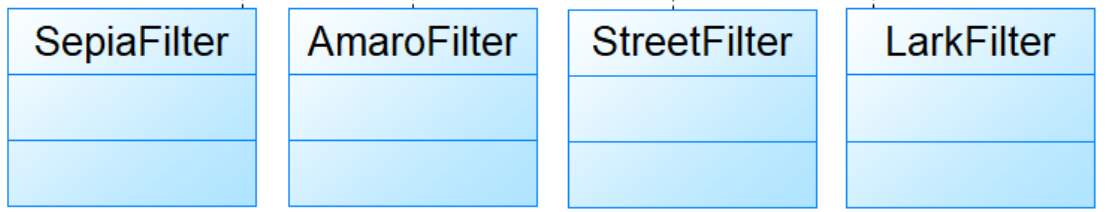
seal by default

OCP putem nasleđivanja





OCP putem kompozicije



Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

DIP

Ponašanje modula je moguće menjati bez da se menja njegov izvorni kod

OCP zahteva procenu

- Šta konkretizovati
- Šta apstrahovati

OCP donosi više prednosti

- Fleksibilnost
- Ponovnu iskoristivost
- Olakšano održavanje

Čist dizajn koda V

SOLID Principi čistog dizajna?

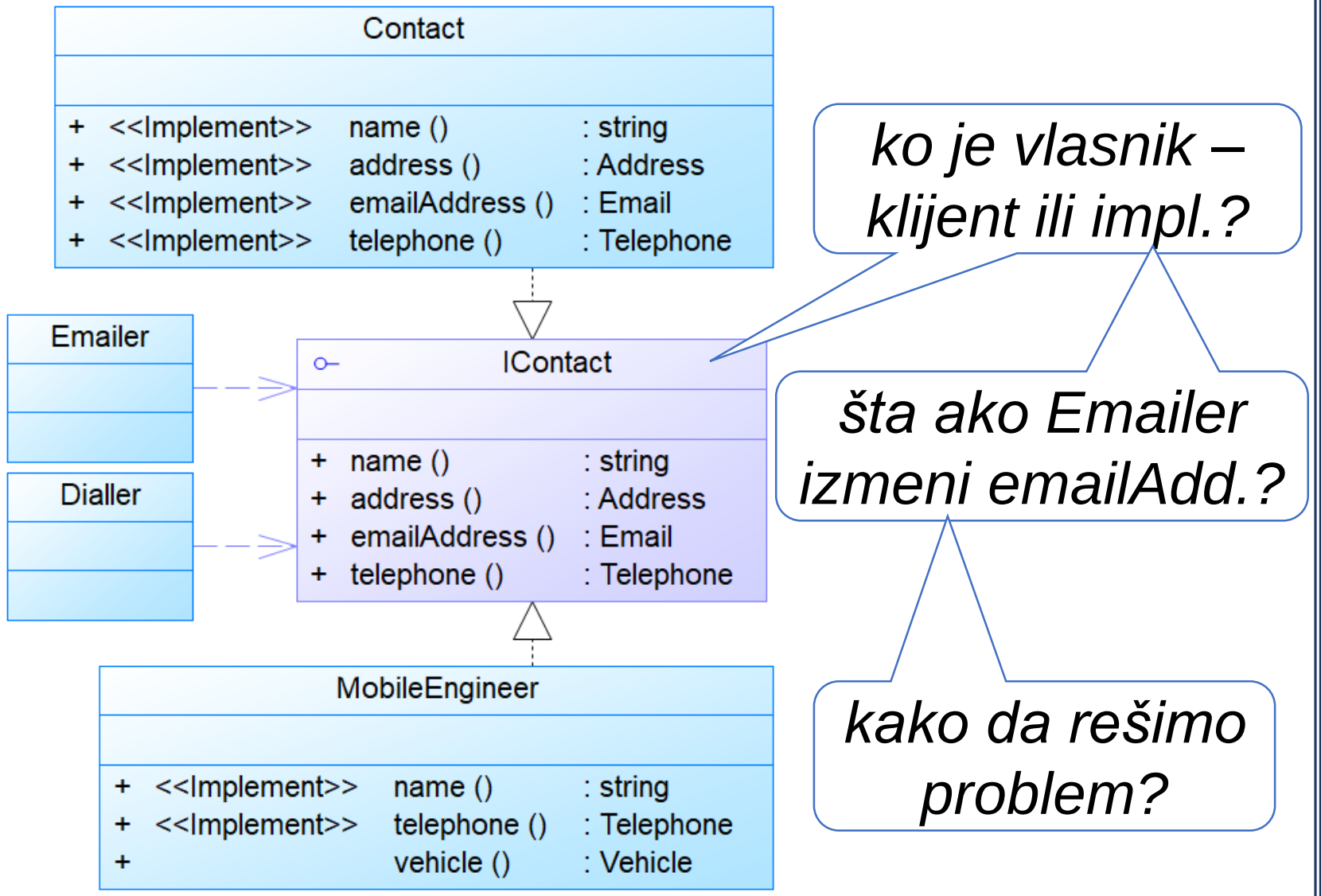
SRP

OCP

ISP

DIP

Interfejsi su tanki, da njihovi klijenti ne zavise od metoda koje ne koriste



Čist dizajn koda V

SOLID Principi čistog dizajna?

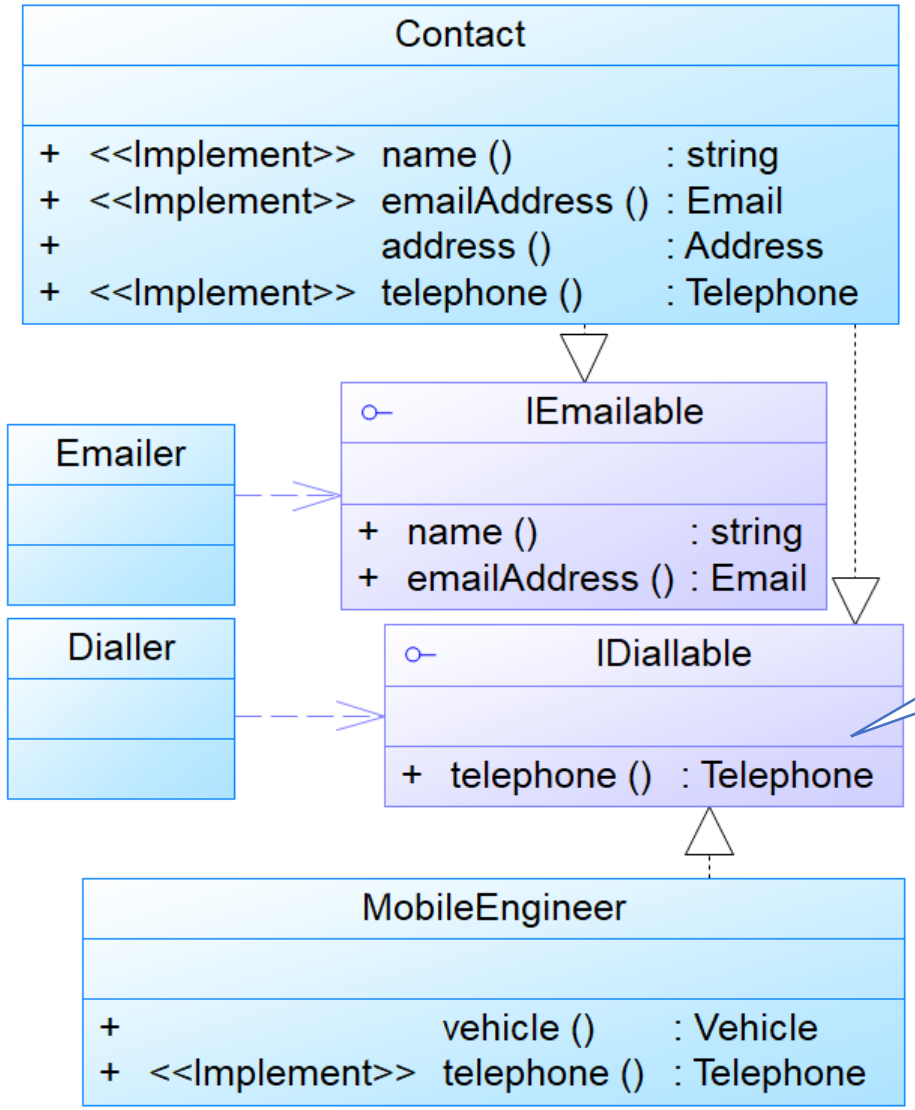
SRP

OCP

ISP

DIP

Interfejsi su tanki, da njihovi klijenti ne zavise od metoda koje ne koriste



header interface – dobijen iz klase

role interface – šta klijentu treba

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

ISP

DIP

Interfejsi su tanki, da njihovi klijenti ne zavise od metoda koje ne koriste

ISP podržava druge principe

- Implementacija „debelog“ interfejsa narušava SRP
- Implementacija dela „debelog“ interfejsa narušava LSP

ISP podstiče dobar dizajn

- Implementacije
- Klijenta

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

LSP

ISP

DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

```
class Rectangle {
  int height;
  int width;

  void setW(int w) {
    this.width = w;
  }
  void setH(int h) {
    this.height = h;
  }
  // Kod za getere
}
```

```
class Square
  extends Rectangle {
  @Override
  void setW(int w) {
    this.width = w;
    this.height = w;
  }
  @Override
  void setH(int h) {
    this.height = h;
    this.width = h;
  }}
```

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

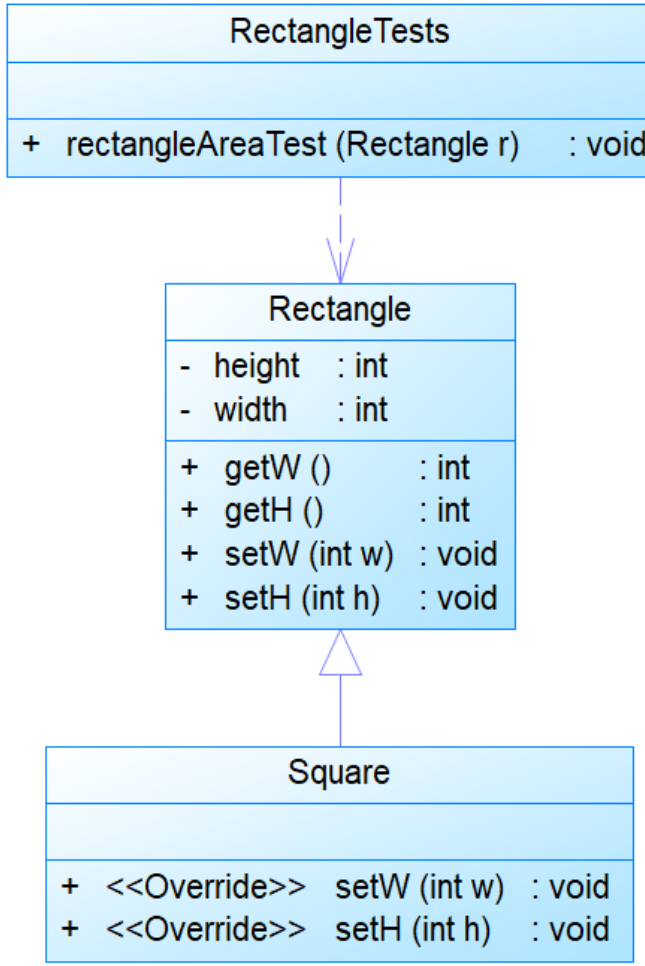
OCP

LSP

ISP

DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku




```
class RectangleTests {
 void rectangleAreaTest(
    Rectangle r) {
  r.setW(5);
  r.setH(2);
  assert
   r.getW()*r.getH() == 10;
 }
}
```

šta dodati da radi za oba?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

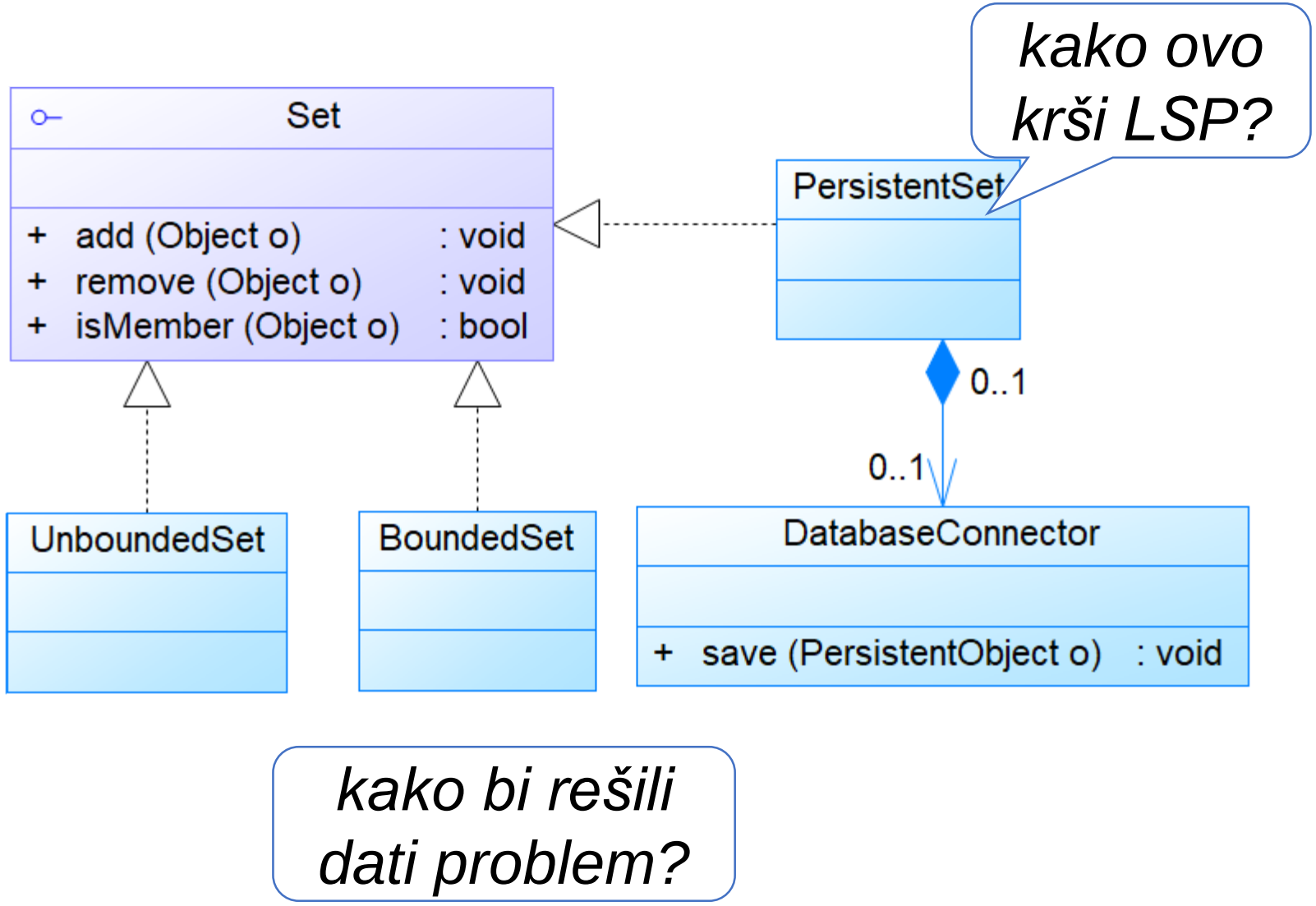
OCP

LSP

ISP

DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku



kako bi rešili dati problem?

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP
OCP
LSP
ISP
DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

Kršenje LSP-a

- (Uglavnom) *Cast* u kodu klijenta nadklase (eksplicitan ili *instanceof*)
- Kada podklasa ima zahtevnije preduslove za izvršavanje ponašanja koje nasleđuje
- Kada podklasa ima slabije postuslove kao rezultat ponašanja koje nasleđuje
- Kada podklasi nije potreban deo metoda i polja nadklase (*refused bequest*)

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

LSP

ISP

DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

LSP zahteva brigu o apstrakciji

Da li *override* menja pred i post uslove koje očekuju klijenti

Da li je data klasa potomak nadklase

LSP donosi više prednosti

Ispravnost polimorfizma

Smislene hijerarhije

Čist dizajn koda V

SOLID Principi čistog dizajna?

SRP

OCP

LSP

ISP

DIP

Šta je SOLID?

- **S**ingle Responsibility Principle
- **O**pen-Closed Principle
- **L**iskov Substitution Principle
- **I**nterface Segregation Principle
- **D**ependency Inversion Principle

Šta nije SOLID?

- Gospodar
- *Framework*, biblioteka, šablon
- *Technology-specific*

Čist dizajn koda V

SOLID Principi čistog dizajna?

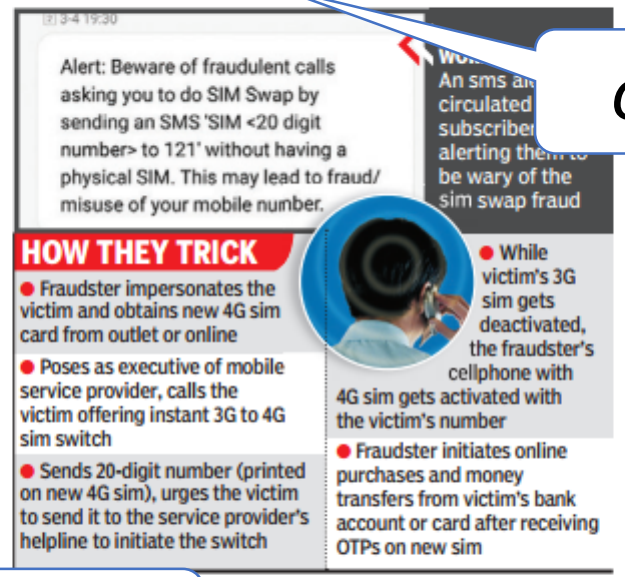
Formatiranje i paketiranje?

Many B'lureans lose cash to sim card swap fraud

Bank Insiders Part Of Ploy: Investigators

Petlee.Peter@timesgroup.com

Bengaluru: If you are using a cellphone number with a 3G sim card and your online banking account is linked to it, you could be the next victim of a thriving 'sim card swap fraud'. At least 30 Bengalureans have reportedly fallen prey to scamsters, losing huge sums of money since mid-2016.



…WARE THE TRAP

"The male caller claimed he was calling from a mobile service provider and confirmed with me if I was still using a 3G sim. He told me that there is an offer for easy swapping to 4G for better internet speed and sent me a 20-digit number by SMS after disconnecting the call," he said.

…osh took …e 20-digit …le service …and selected option 1 to confirm the 4G swap as advised by the conman. "Within a few seconds my sim card got deactiva… and it remained …

…tronics good worth over … lakh were purchased on … using his HDFC bank acco…

According to an inv… gating officer with the CID's cybercrime unit, the modus operandi is thus: The culprits obtain a new 4G sim for the victim's cellphone number by either impersonating him at an outlet of the service pro-

…tives from the service provid… …phone becomes inactive, the 4G one on the fraudsters' cellphone becomes active. The fraudsters then use it to receive OTPs," the officer added.

Investigators suspect the scamsters must be obtaining victims' confidential bank account or card details, including cellphone details, from bank insiders. "They try every number pertaining to the accounts and some 3G sim card users fall for it," the officer added.

Over 30 victims of the sim swap fraud have approached cybercrime police stations of state CID and Bengaluru city police (BCP) since mid-2016.

Some like Manish Raj, a city-based BPO employee, who are tech aware have also fallen prey to the fraud. "I didn't receive a call but only an internet-generated SMS …umber …hich I …d lost …ICI ac-

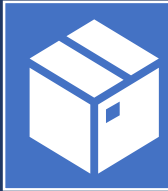
o čemu se radi?

pregled bez detalja

razrada sa sve više detalja

Čist dizajn koda V

SOLID Principi čistog dizajna?



Formatiranje i paketiranje?

Datoteku koda urediti kao novinski članak

`TragedyPerformanceCalculator.java`

o čemu se radi?

```
public class TragedyPerformanceCalculator {
  public double calculatePrice() {...}
  public double calculateVolCredits() {...}
  private int setBonus(int factor) {...}
  private double uti
}
```

pregled bez detalja

razrada sa sve više detalja

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Kod smisleno grupisati unutar datoteke

- Belina razdvaja misli

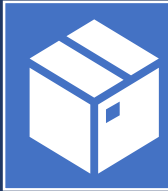
```
public class BoldWidget extends Widget {
  private Pattern pattern = Pattern.compile();
  public BoldWidget(Widget parent, String text) {
    Matcher match = pattern.matcher(text);
    addChildWidgets(match.group(1));
  }
  public String render() throws Exception {
    StringBuffer html = new StringBuffer("<b>");
    html.append(childHtml()).append("</b>");
    return html.toString();
  }
}
```

kako izgleda kada skrolamo kroz kod?

šta ukazuje belina unutar funkcije?

Čist dizajn koda V

SOLID Principi čistog dizajna?



Formatiranje i paketiranje?

Kod smisleno grupisati unutar datoteke

- Vezane koncepte staviti jedan uz drugi

```
public class XYZ {
  private Object A;
  private Object B;

  public Object XY() {...}
  private Object XY1() {...}

  public Object XZ() {
    A.CDX();
    Object T = …
    T.TGX();
  }}
```

zašto polja nisu uz funkcije koje ih koriste?

gde postaviti ekstrahovane funkcije?

gde deklarisati promenljivu unutar funkcije?

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Izbegavati ponavljanje istog koda

- *Don't Repeat Yourself* (*DRY*)

```
public User getUser(int id) {
  if(activeUser.getRoles().contains("ADMIN")) {
    User u = findUserById(id);
    u.setPrivateData(getUserPrivateData(id));
    return u;
  } else {
    User u = findUserById(id);
    return u;
  }
}
```

šta ne valja sem duplikata?

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Izbegavati ponavljanje istog koda

- *Don't Repeat Yourself* (*DRY*)

```
public User getUser(int id) {
  User u = findUserById(id);
  if(activeUser.hasPrivilege("ADMIN")) {
    u.setPrivateData(getUserPrivateData(id));
  }
  return u;
}
```

što je dupliran kod problematičan?

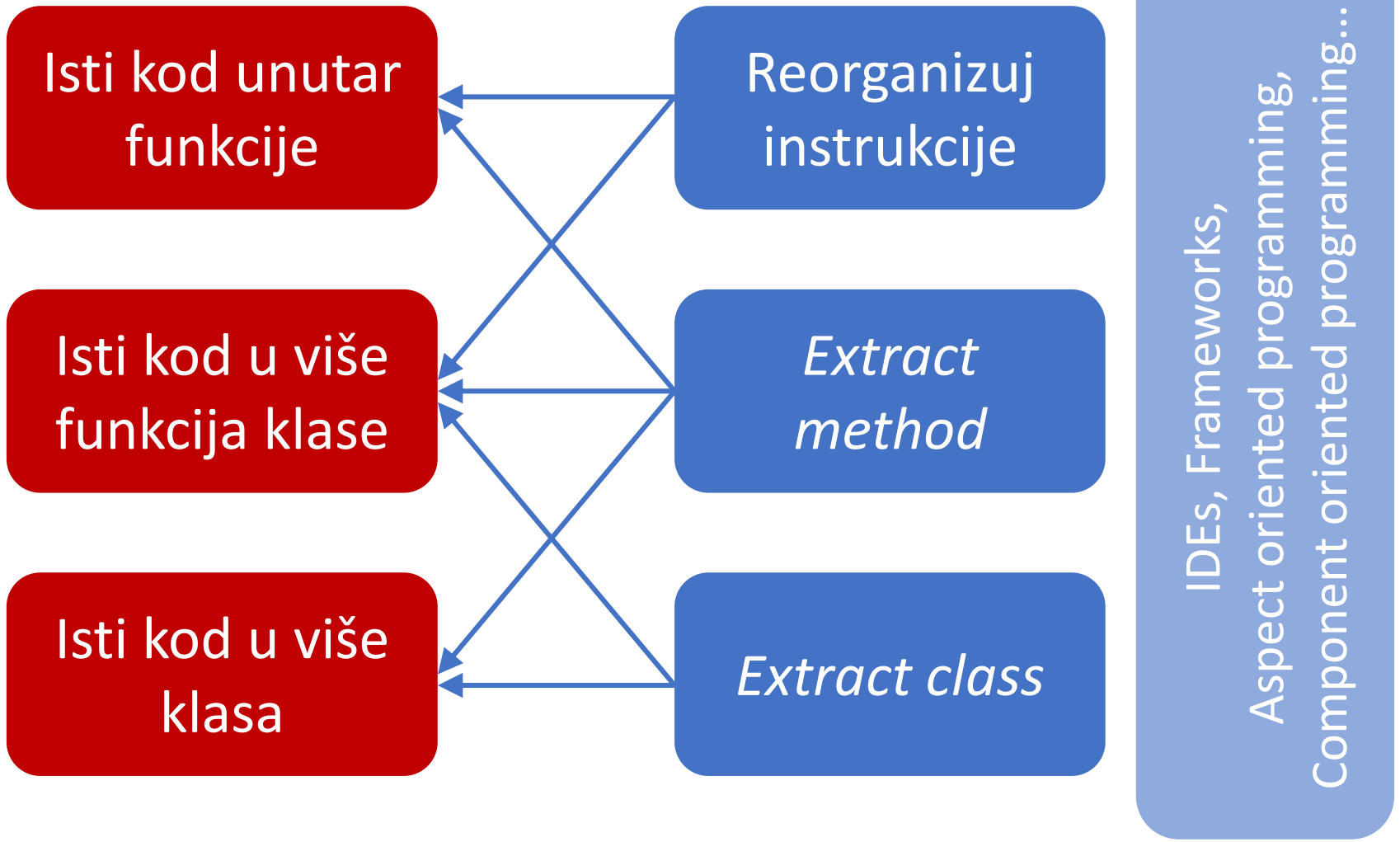
Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

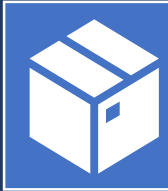
Izbegavati ponavljanje istog koda

- *Don't Repeat Yourself* (*DRY*)



Čist dizajn koda V

SOLID Principi čistog dizajna?



Formatiranje i paketiranje?

Z11.1: Refaktoriši da ostaneš suv

- https://pastebin.com/T0BQyAGx

Z11.2: Prateći DIP i DRY, definiši repozitorijume za sledeće entitete:

- https://pastebin.com/XvaEbeZy

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Paketiranje – značajno ime

Primeri

org.hibernate.sessions.exceptions

Zdravo.PatientSystem.HealthMonitor

Struktura

(<Company>|<Domain>).[(<Product>|<Technology>)]
[.<Feature>][.]

Resursi

https://docs.oracle.com/javase/tutorial/java/package/namingpkgs.html

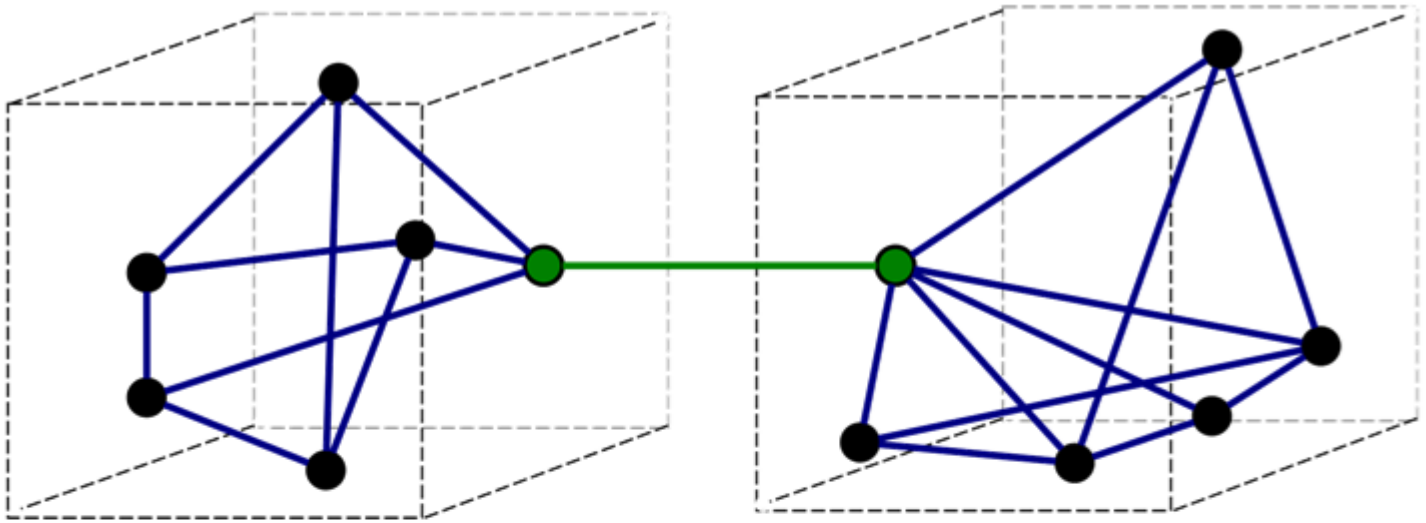
https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/

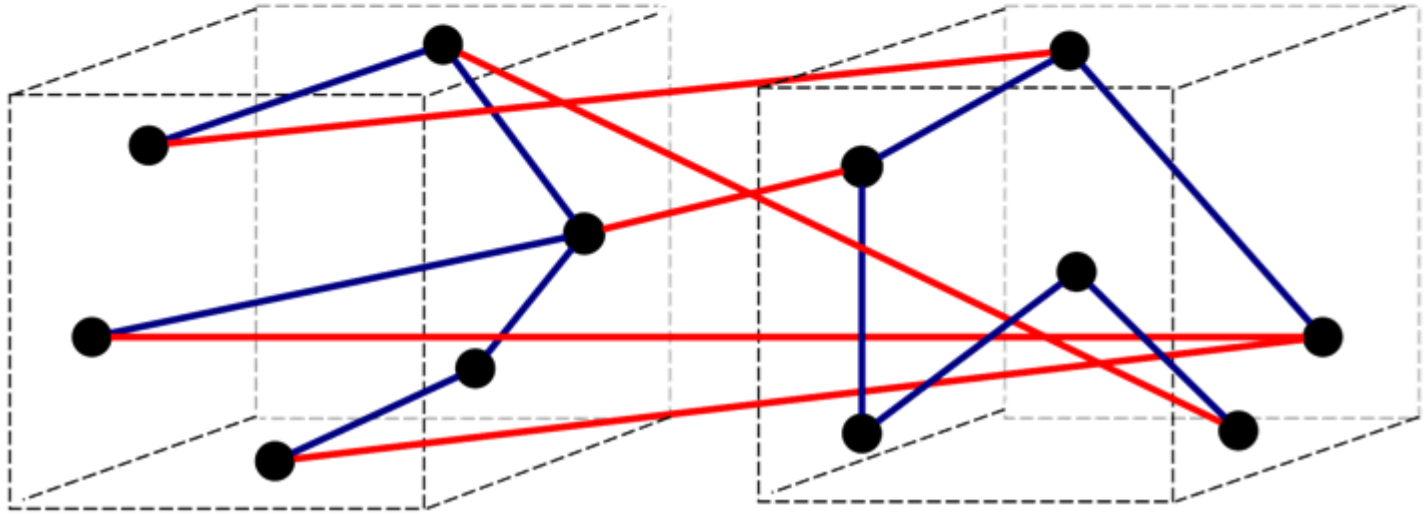
Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Paketiranje – visoka kohezija





Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Paketiranje – dobra organizacija

L ocate

Intuitivno, jednostavno, lako

I dentify

Ime kaže sve

F lat

Izbegavati duboko ugnježđavanje

T ry to be DRY

Izbegavati redundantnost u imenu

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Dogovorite timske konvencije

- Čist kod pospešuje komunikaciju
- Kod koji prati timska pravila pospešuje timsku komunikaciju
- Paketiranje, imenovanje, indentacija, formatiranje, širina koda, dužina datoteka…

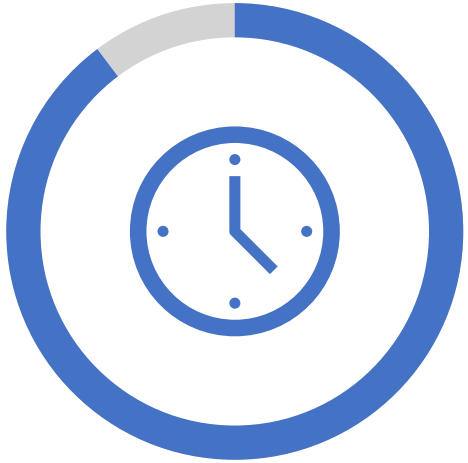
Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

×



Održavanje košta 40 do 80% ukupnog troška

Na zrelom projektu odnos čitanja i pisanja je 10:1

Većina troška razvoja je u čitanju koda

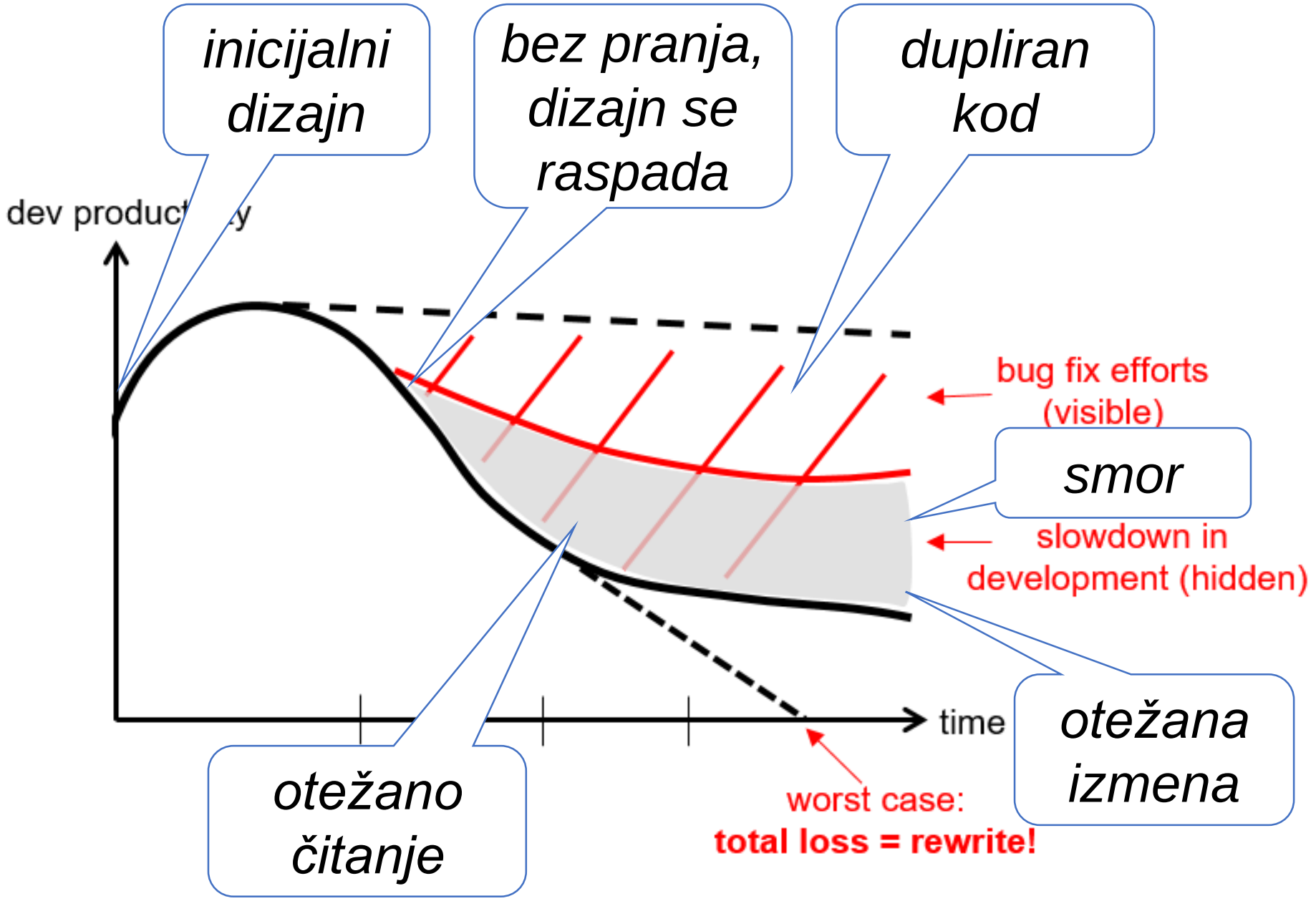
Čist dizajn koda **značajno** smanjuje troškove

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

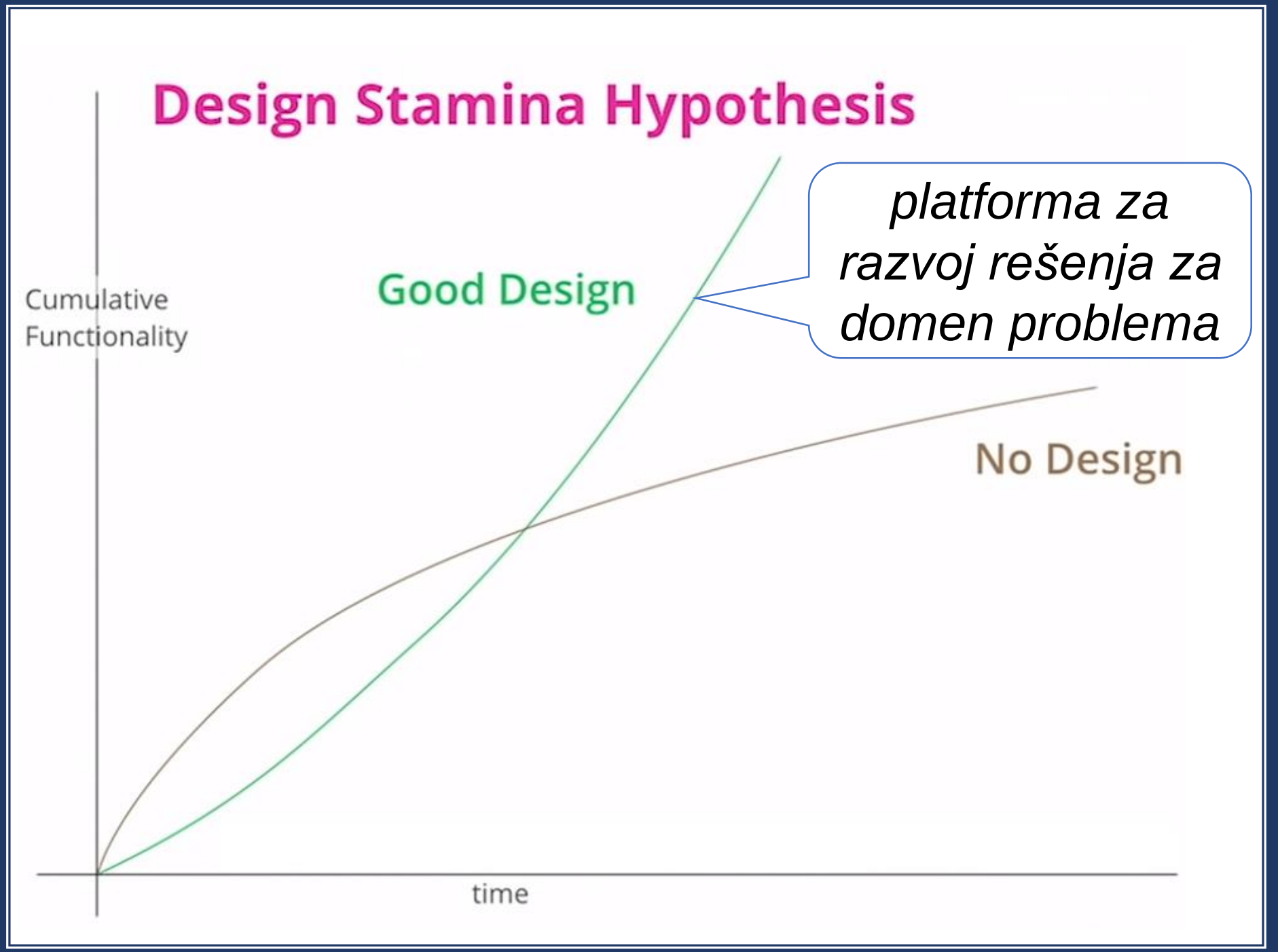


Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?



Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Zašto pišemo prljav kod?

Nedostatak podrške menadžmenta

Nedostatak veštine i brige inženjera

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Zašto pišemo prljav kod?

Nedostatak
veštine i **brige** inženjera

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Zašto pišemo prljav kod?

Kako rešiti nedostatak veštine i **brige** inženjera?

- ~5 predavanja
- ~5 vežbi
- ~5 knjiga
- Ozbiljan projekat (i svi budući…)

- Novac
  Ugled
- Zabava
  Izazovi
- Ispunjenost
  Smisao
- Sloboda

Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Kako da integrišemo čišćenje koda u posao?

Izdvoj vreme za refaktorisanje u procene

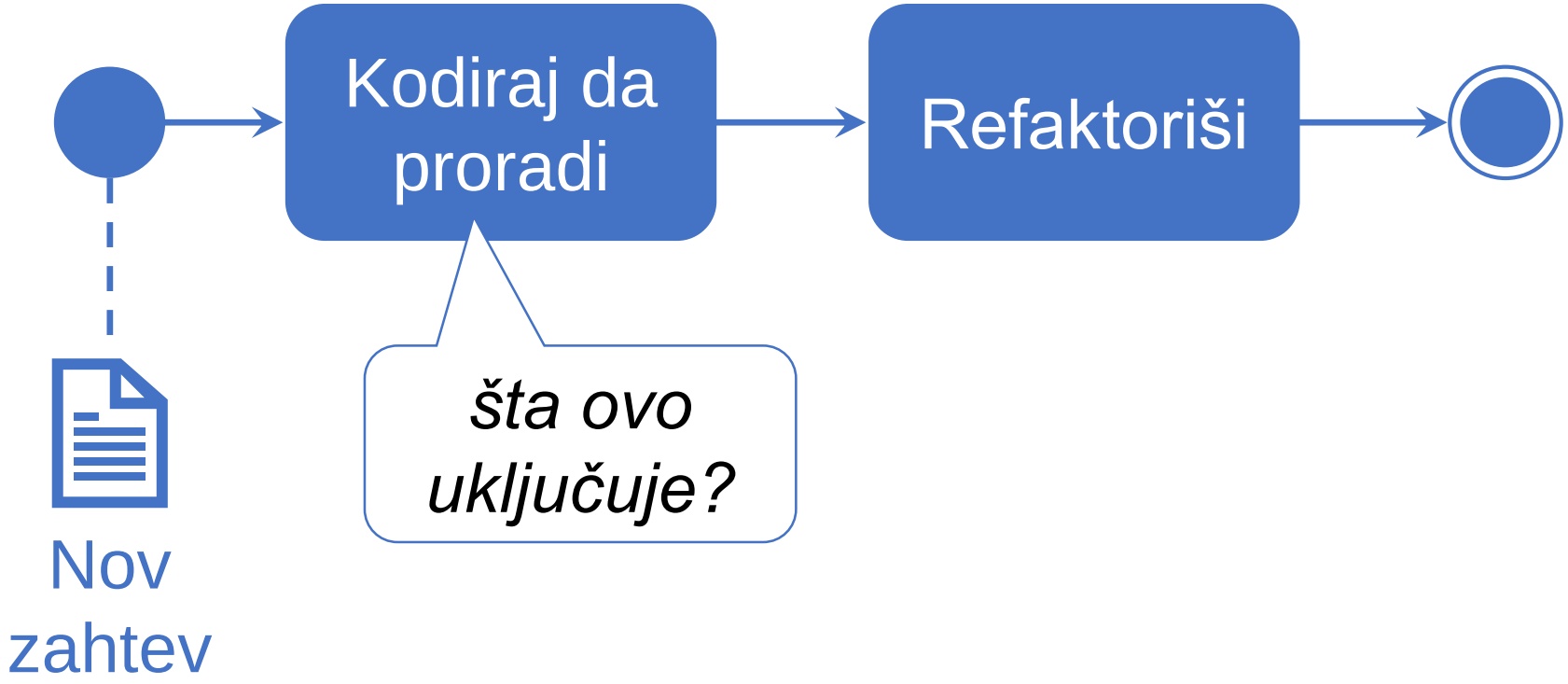
Šta je refaktorisanje?

Izmena unutrašnje strukture softvera bez izmene vidljivog ponašanja, radi boljeg razumevanja strukture i olakšavanja njene buduće izmene

Čist dizajn koda V

- SOLID Principi čistog dizajna?
- Formatiranje i paketiranje?
- Čist kod u praksi?

Kako da integrišemo čišćenje koda u posao?



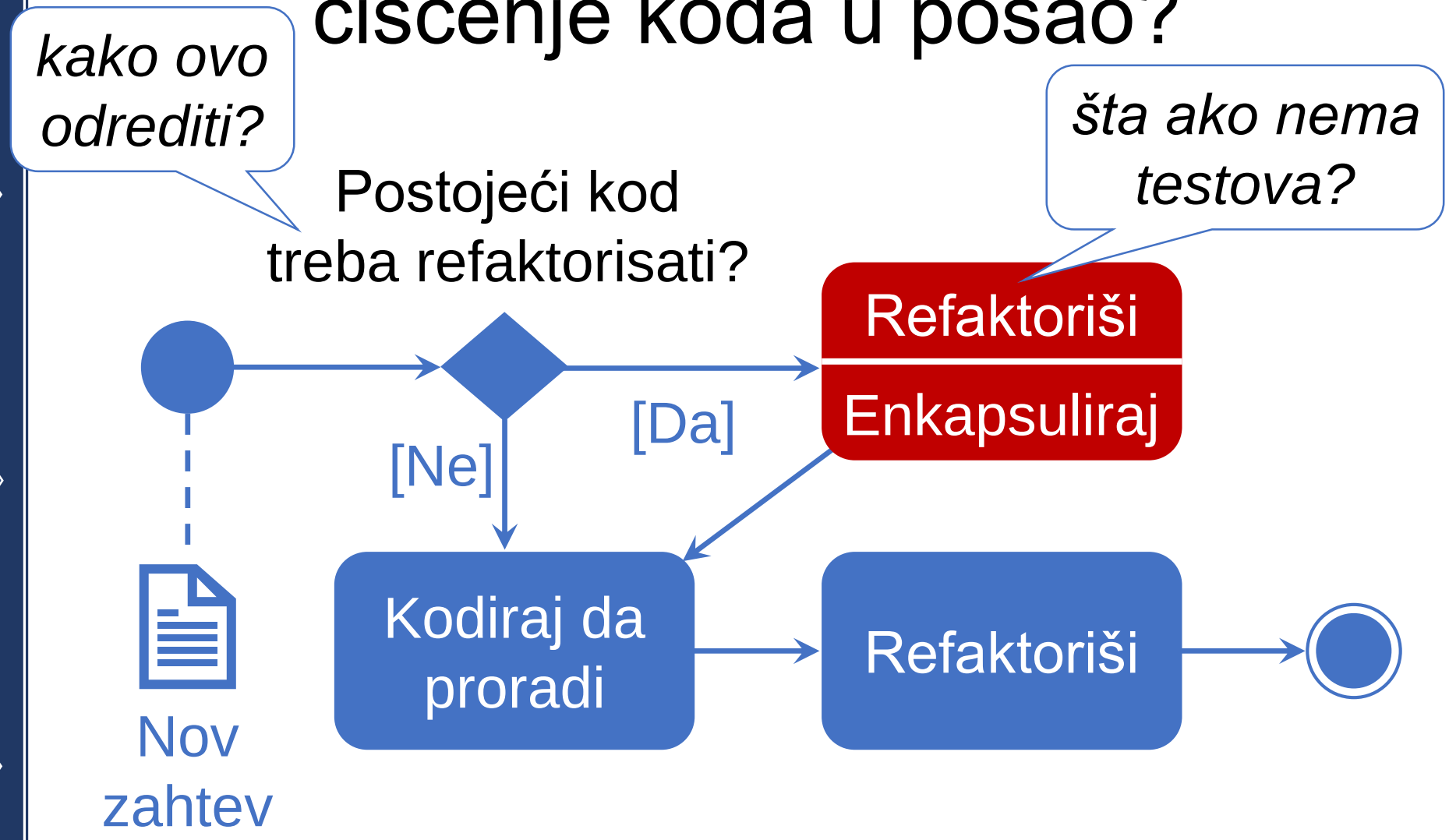
Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Kako da integrišemo čišćenje koda u posao?



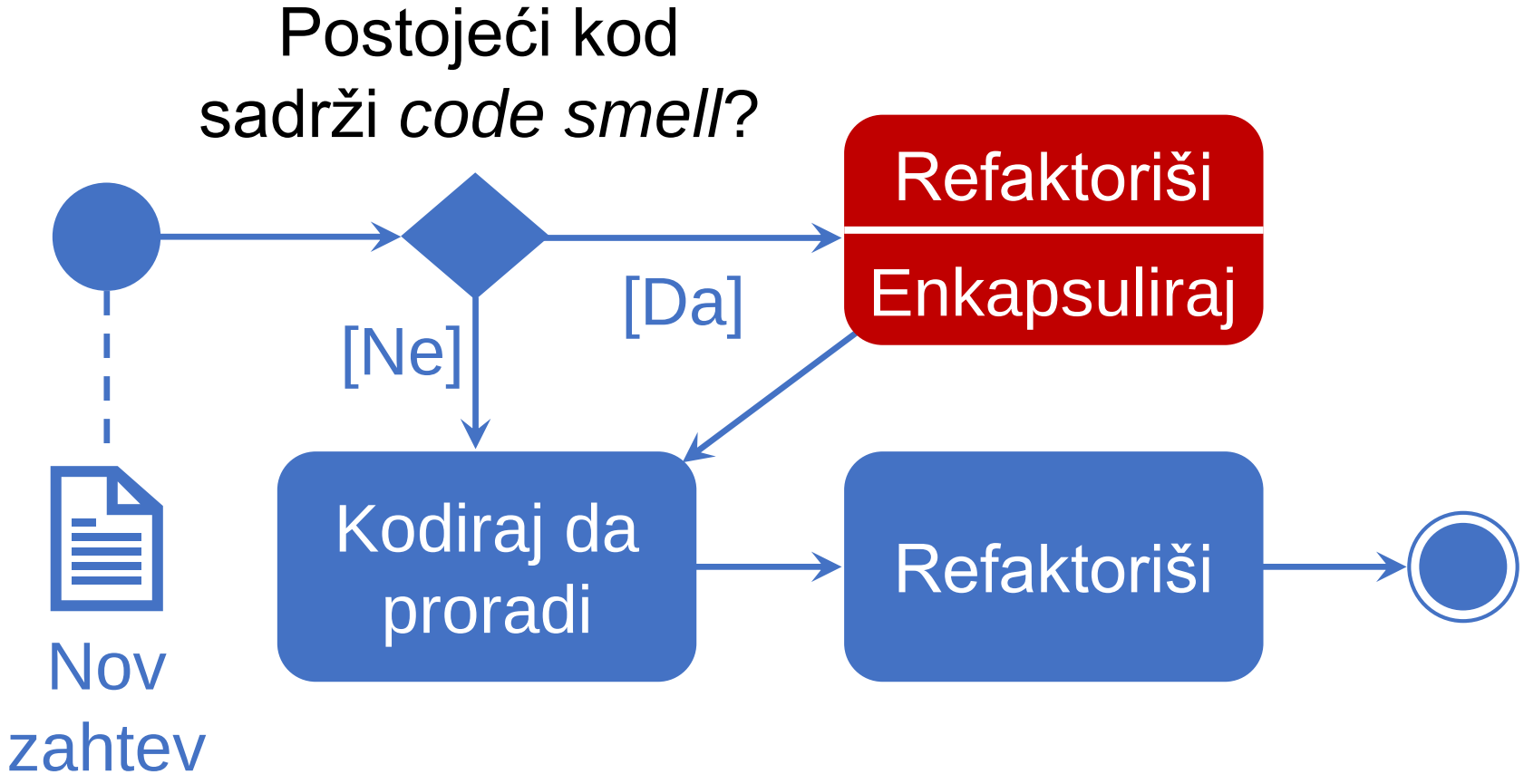
Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Kako da integrišemo čišćenje koda u posao?



Čist dizajn koda V

SOLID Principi čistog dizajna?

Formatiranje i paketiranje?

Čist kod u praksi?

Šta je *code smell*?

Struktura u kodu koja ukazuje na potrebu za refaktorisanjem

Čist dizajn koda V

SOLID Principi čistog dizajna?

Z11.3: Mapiraj na pređeno gradivo

Formatiranje i paketiranje?

Čist kod u praksi?

MYSTERIOUS NAME	COMMENTS	LONG FUNCTION	LONG PARAMETER LIST
MUTABLE DATA	DIVERGENT CHANGE	SHOTGUN SURGERY	FEATURE ENVY
PRIMITIVE OBSESSION	REPEATED SWITCHES	DUPLICATED CODE	LAZY ELEMENT
TEMPORARY FIELD	MESSAGE CHAINS	MIDDLE MAN	INSIDER TRADING
CLASSES WITH DIFFERENT INTERFACES	DATA CLASS	REFUSED BEQUEST	LOOPS
GLOBAL DATA	DATA CLUMPS	SPECULATIVE GENERALITY	LARGE CLASS

Čist dizajn koda V

SOLID Principi čistog dizajna?


Formatiranje i paketiranje?


Čist kod u praksi?

Šta ako menadžment ne podržava refaktorisanje?

Šta ako kolege ne podržavaju refaktorisanje?

Menjajte firmu ili menjajte firmu

Easier To Change