Uvod u softversko inženjerstvo

SOLID principi

Nikola Luburić
nikola.luburic@uns.ac.rs

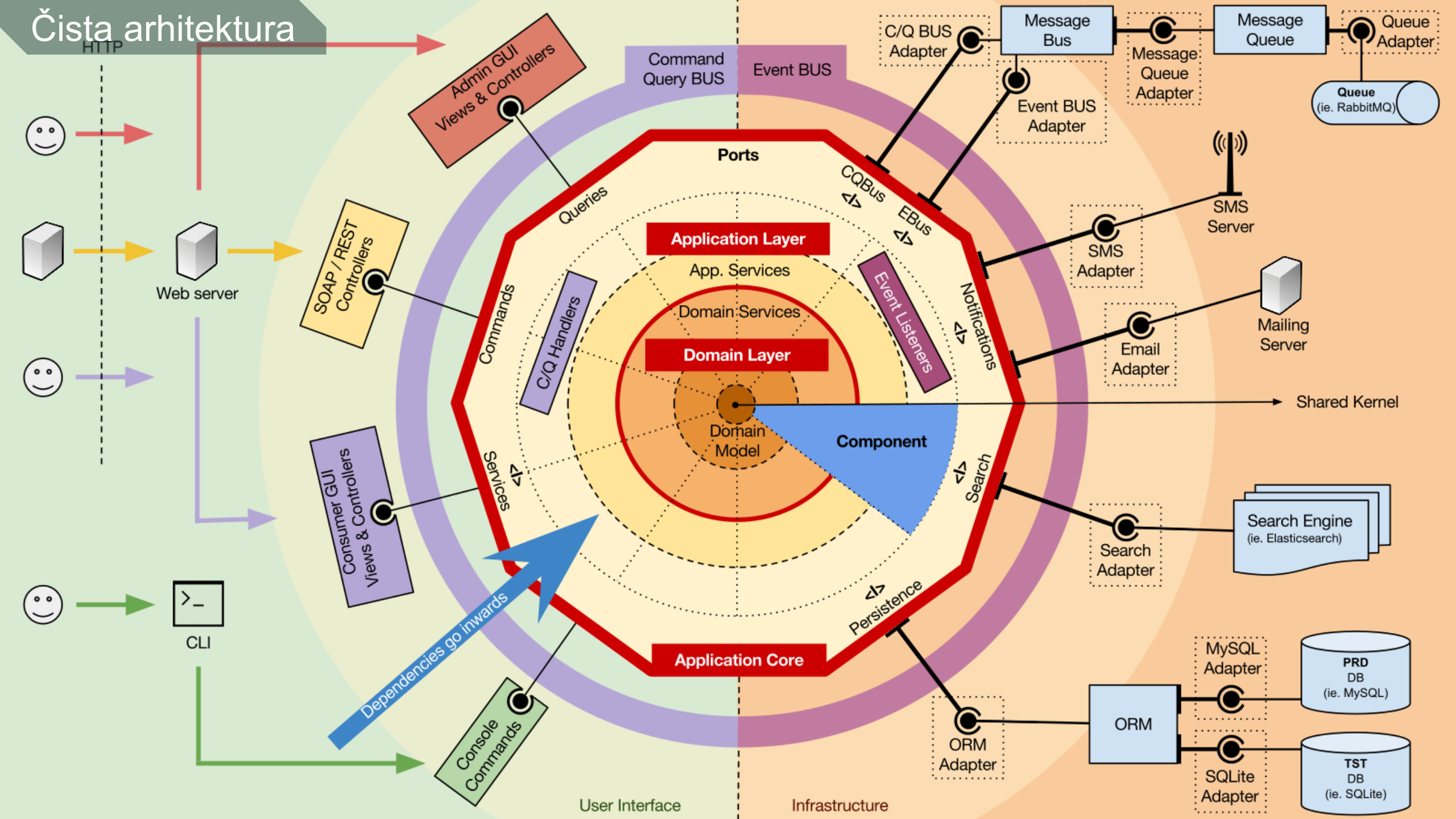
Šta je SOLID?

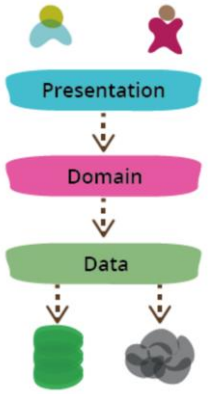
- **S**ingle Responsibility Principle
- **O**pen-Closed Principle
- **L**iskov Substitution Principle
- **I**nterface Segregation Principle
- **D**ependency Inversion Principle

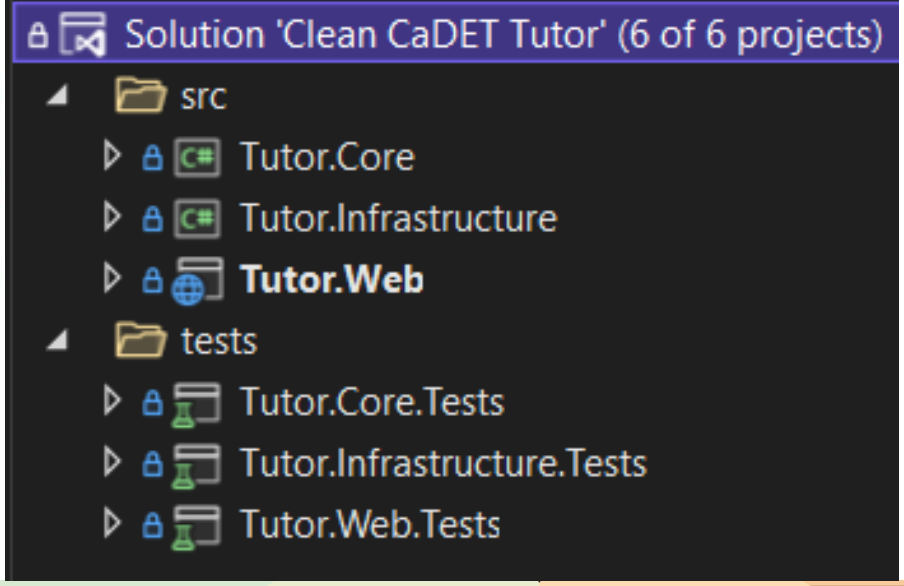
Šta nije SOLID?

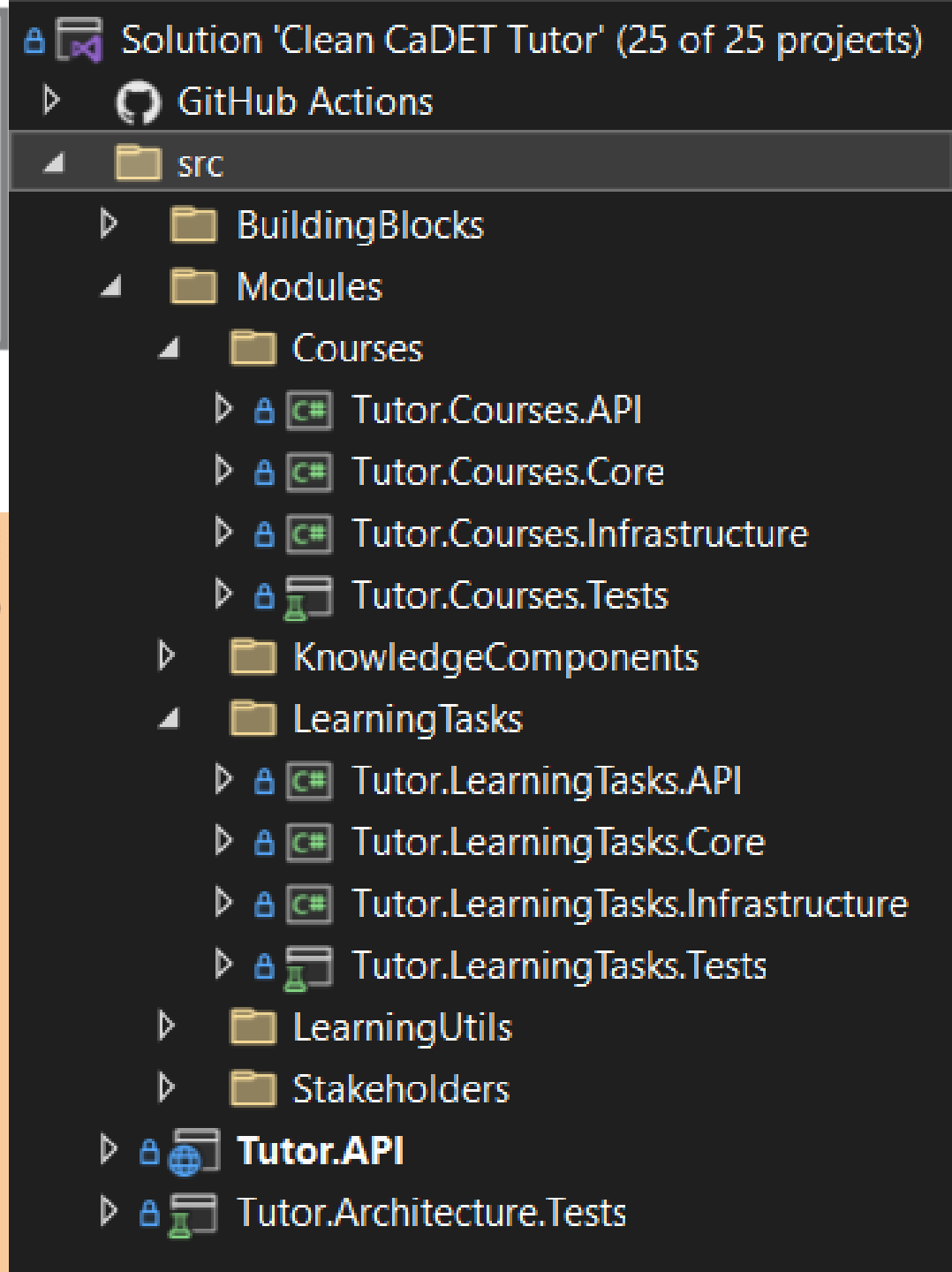
- Gospodar
- *Framework*, biblioteka, šablon
- *Technology-specific*

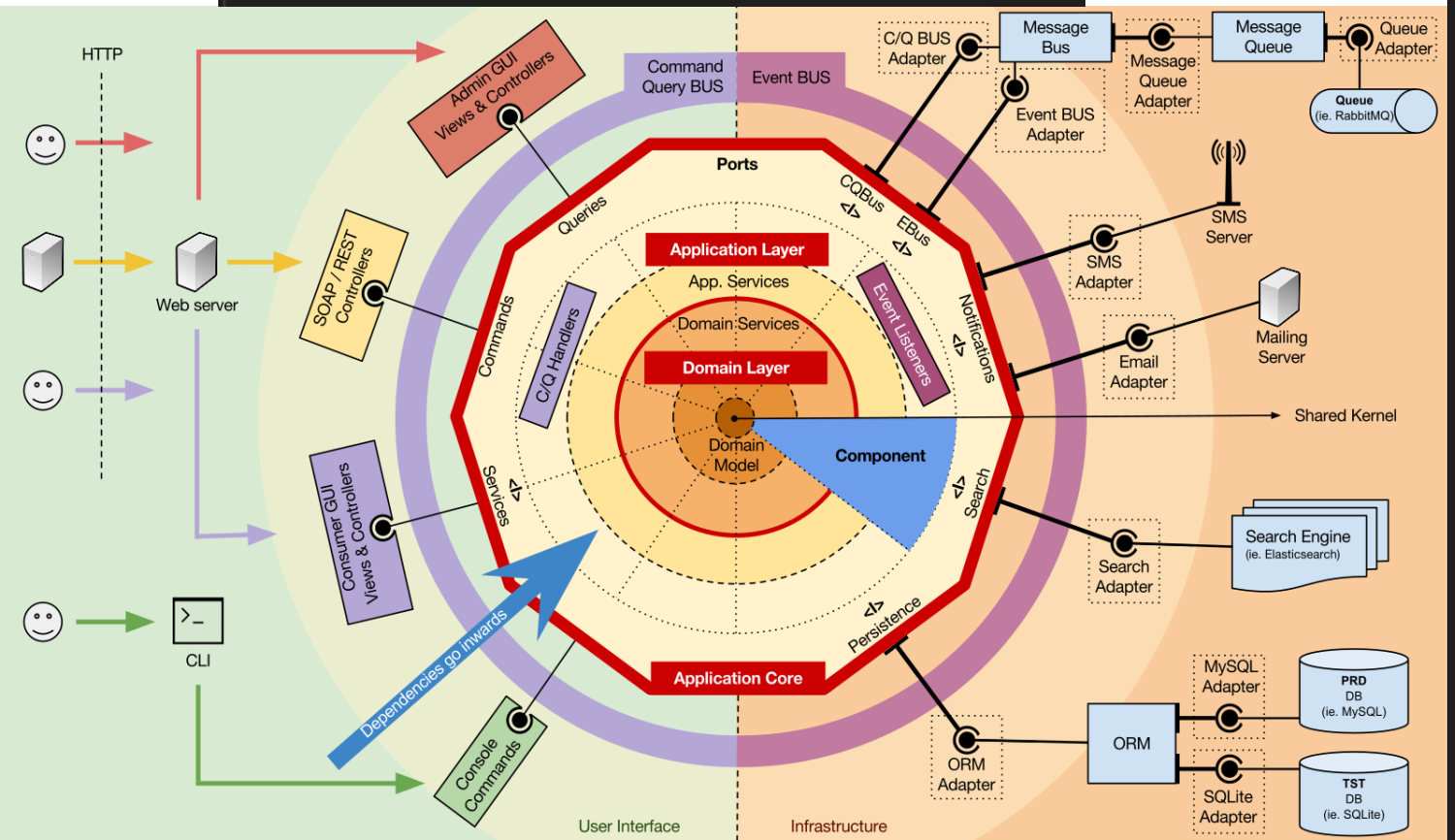
Čista arhitektura




Presentation
Domain
Data

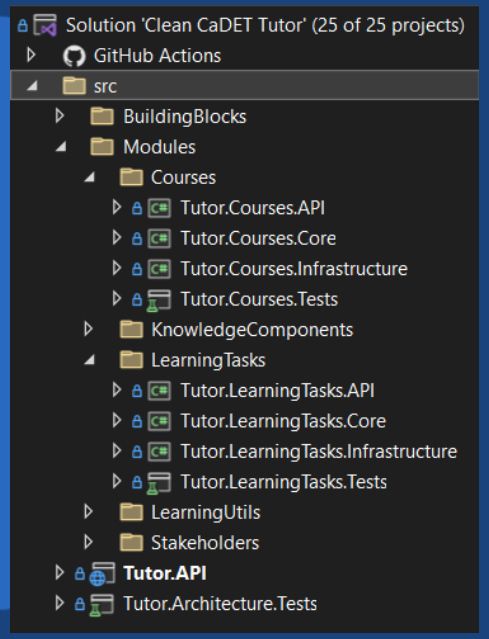

Solution 'Clean CaDET Tutor' (6 of 6 projects)
src
Tutor.Core
Tutor.Infrastructure
Tutor.Web
tests
Tutor.Core.Tests
Tutor.Infrastructure.Tests
Tutor.Web.Tests


Solution 'Clean CaDET Tutor' (25 of 25 projects)
GitHub Actions
src
BuildingBlocks
Modules
Courses
Tutor.Courses.API
Tutor.Courses.Core
Tutor.Courses.Infrastructure
Tutor.Courses.Tests
KnowledgeComponents
LearningTasks
Tutor.LearningTasks.API
Tutor.LearningTasks.Core
Tutor.LearningTasks.Infrastructure
Tutor.LearningTasks.Tests
LearningUtils
Stakeholders
Tutor.API
Tutor.Architecture.Tests


HTTP
Web server
CLI
Admin GUI Views & Controllers
SOAP / REST Controllers
Consumer GUI Views & Controllers
Console Commands
Command Query BUS
Event BUS
Ports
Queries
Commands
Services
CQBus
EBus
Notifications
Application Layer
App. Services
Domain Services
Domain Layer
Domain Model
C/Q Handlers
Event Listeners
Component
Shared Kernel
Search
Persistence
Application Core
Dependencies go inwards
User Interface
Infrastructure
C/Q BUS Adapter
Message Bus
Message Queue Adapter
Message Queue
Queue Adapter
Queue (ie. RabbitMQ)
Event BUS Adapter
SMS Server
SMS Adapter
Mailing Server
Email Adapter
Search Engine (ie. Elasticsearch)
Search Adapter
ORM
ORM Adapter
MySQL Adapter
SQLite Adapter
PRD DB (ie. MySQL)
TST DB (ie. SQLite)

Modularity

Modular monolith



Microservices

Monolithic big ball of mud

Distributed big ball of mud

Number of deployment units

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*

SRP

OCP

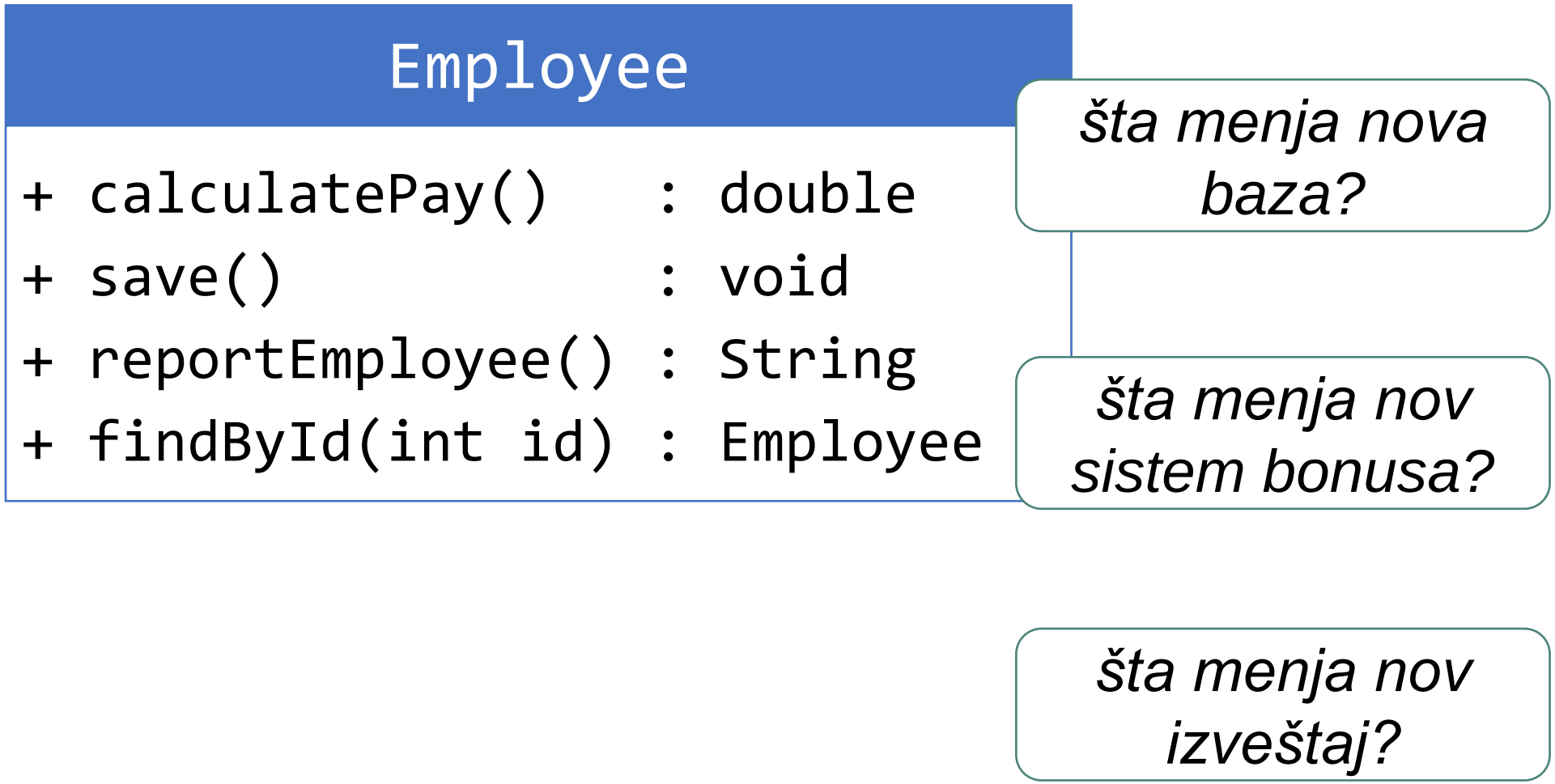
LSP

ISP

DIP

Skupi stvari koje se menjaju iz istih razloga

Razdvoj one koje se menjaju iz različitih



SRP

OCP

LSP

ISP

DIP

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

Employee

EmployeeRepository

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

šta menja nova baza?

PaycheckCalculator

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

šta menja nov sistem bonusa?

EmployeeReporter

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

šta menja nov izveštaj?

SRP

OCP

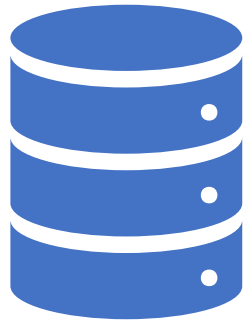
LSP

ISP

DIP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija

Aplikacioni i domenski sloj



Baza podataka

SRP

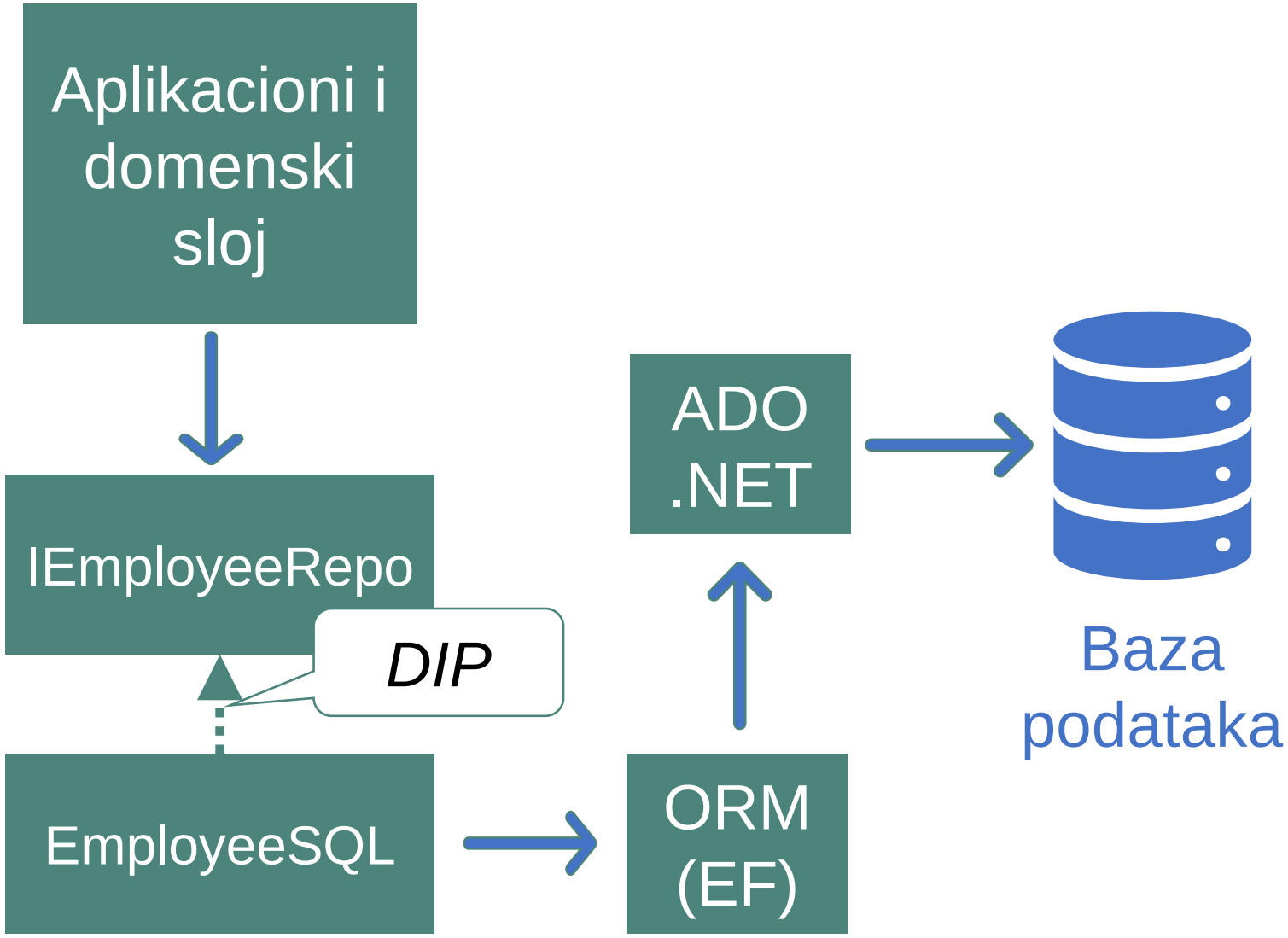
OCP

LSP

ISP

DIP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



SRP

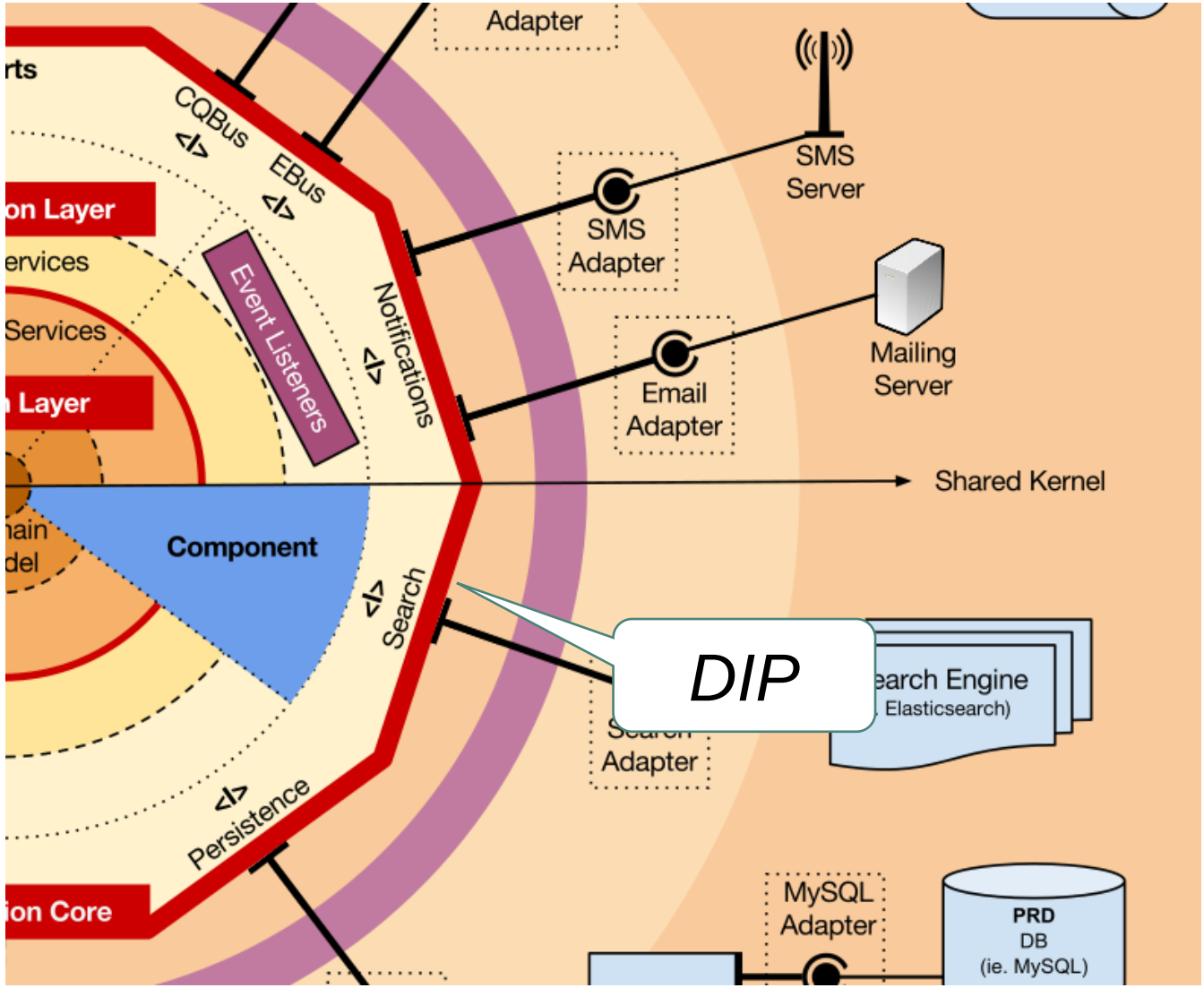
OCP

LSP

ISP

DIP

Apstrakcije ne zavise od implementacije, implementacije zavise od apstrakcija



SRP

OCP

LSP

ISP

DIP

Ponašanje modula je moguće menjati bez da se menja postojeći 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

SRP

OCP

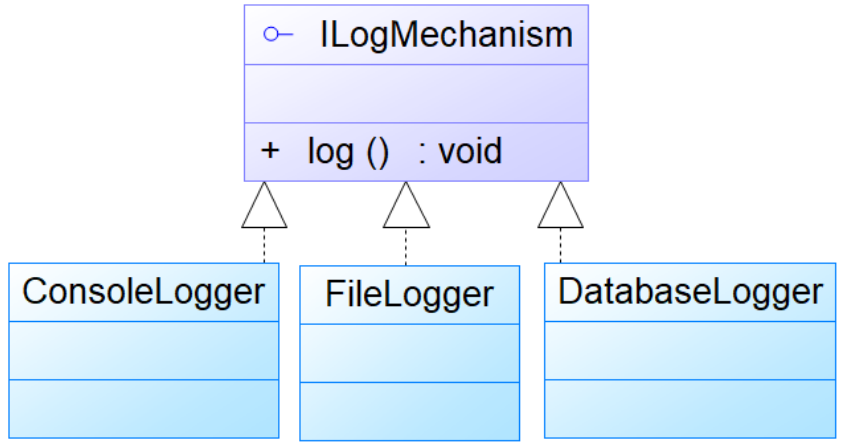
LSP

ISP

DIP

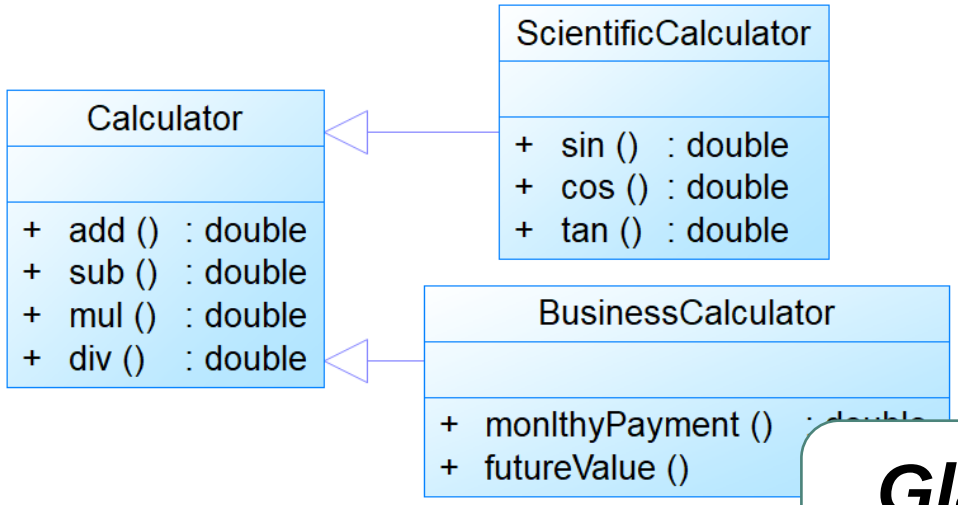
Ponašanje modula je moguće menjati bez da se menja postojeći kod

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

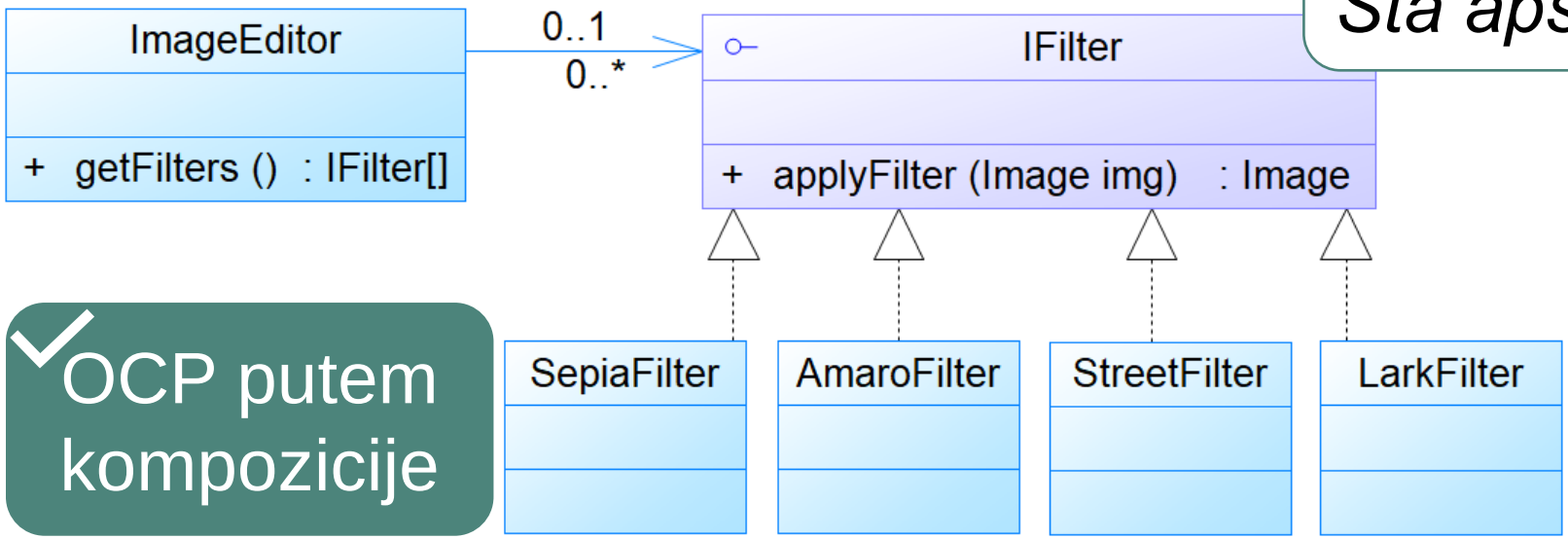


Ponašanje modula je moguće menjati bez da se menja postojeći kod

OCP putem nasleđivanja



***Glavni izazov**:*
Šta apstrahovati?



OCP putem kompozicije

SRP

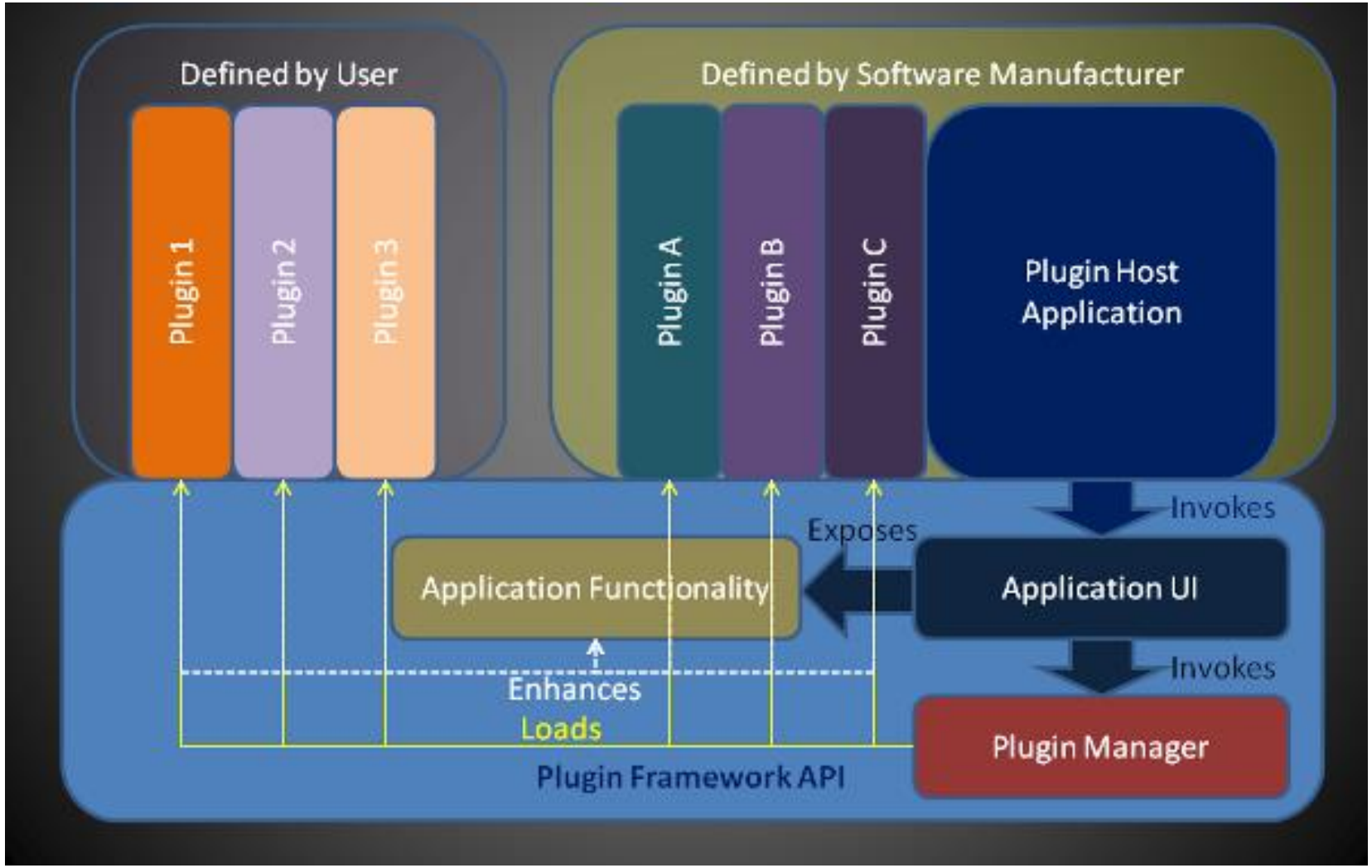
OCP

LSP

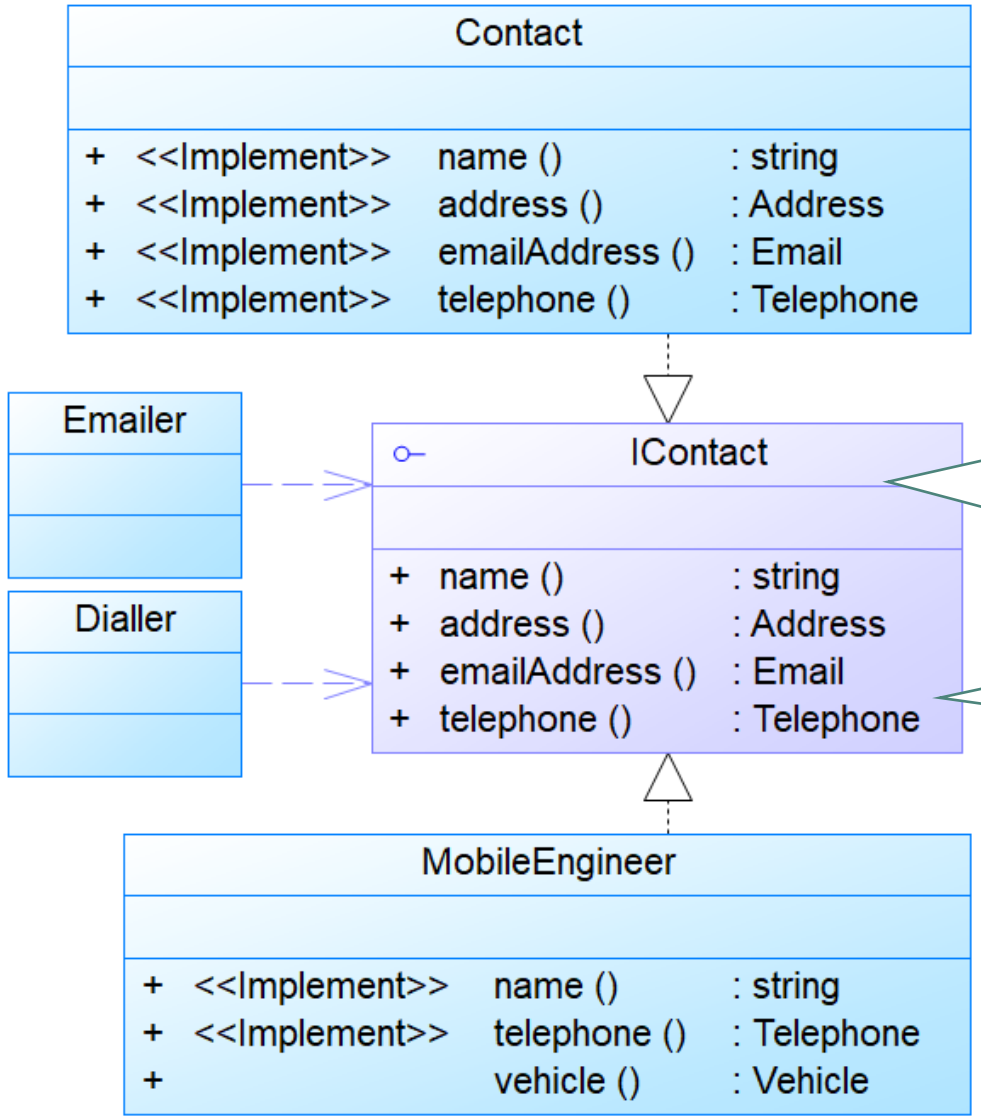
ISP

DIP

Ponašanje modula je moguće menjati bez da se menja postojeći kod



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



header interface – dobijen iz klase

Na osnovu čega je definisan interfejs – klijent ili impl.?

Šta je problem?

Kako da ga rešimo?

SRP

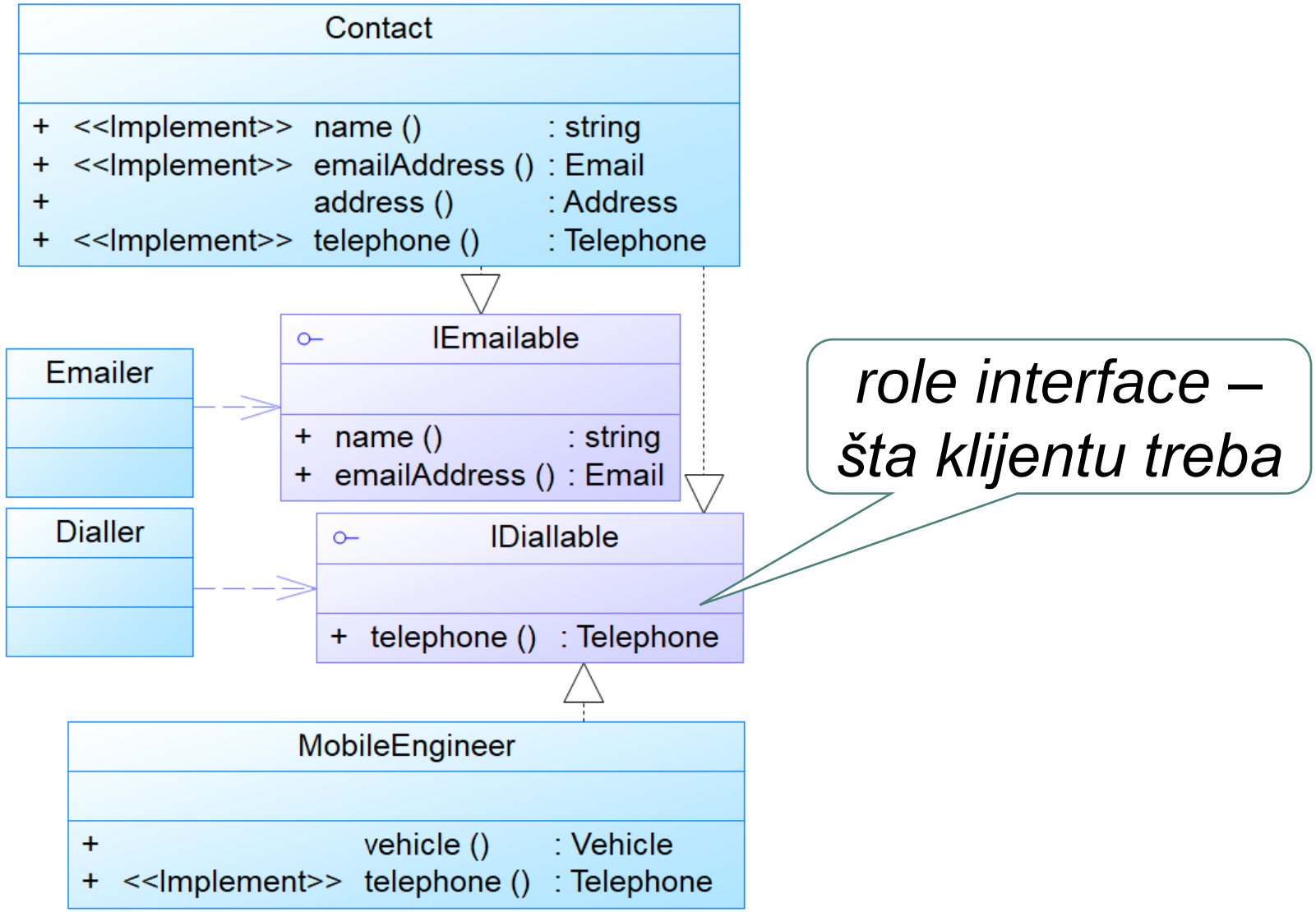
OCP

LSP

ISP

DIP

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



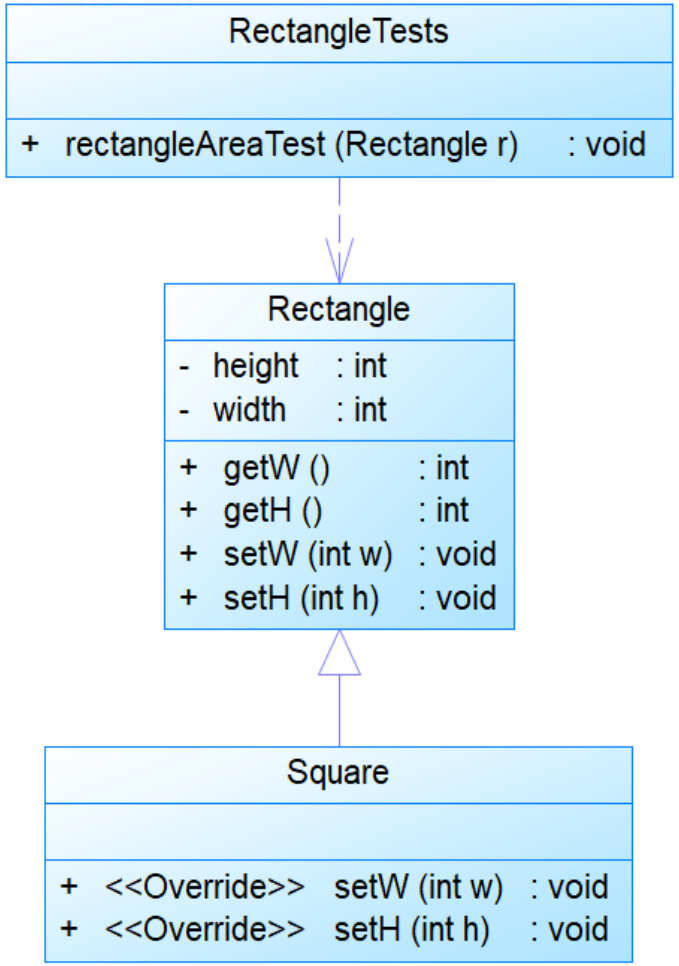
Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

```
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;
  }}
```

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku




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

šta dodati da radi za oba?

SRP

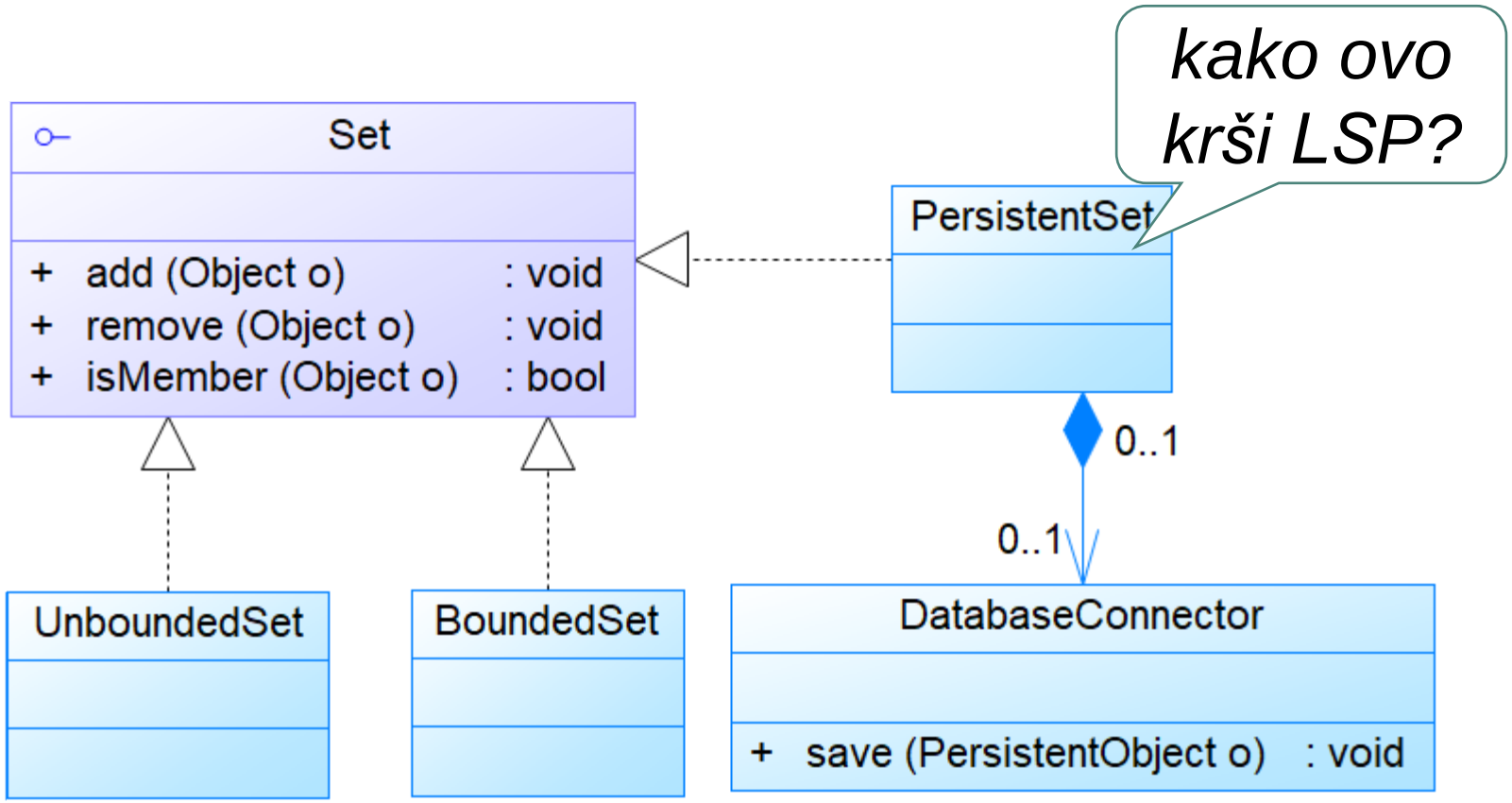
OCP

LSP

ISP

DIP

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

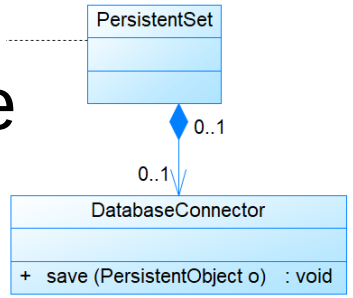


kako bismo rešili dati problem?

Podklase se koriste umesto nadklase bez da klijent nadklase uočava razliku

Kršenje LSP-a

- (Često) *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*)



Navedeno važi i za interfejse

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*

2. Šta radimo?

Pravimo softversko rešenje sastavljeno od **održivog** koda, podržani odabranim **tehničkim i metodološkim alatima**.