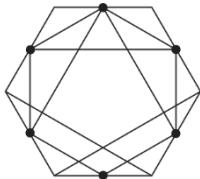
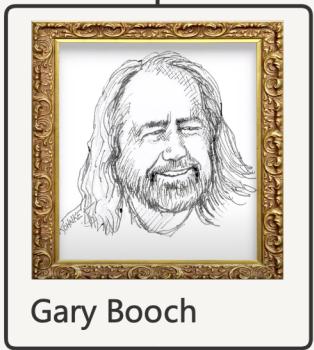


SINGLE RESPONSIBILITY



Clean Code

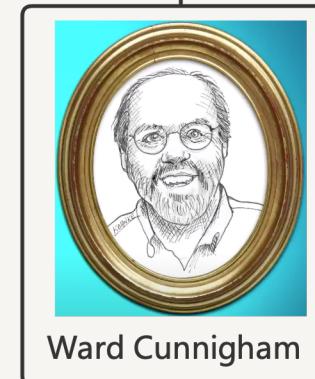


Simple & Direct

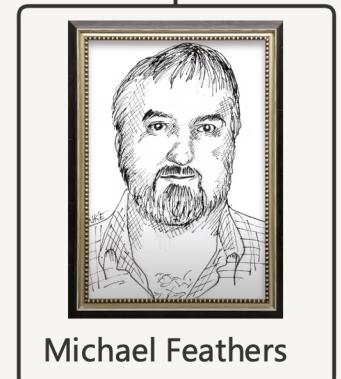
Reads as well-written prose



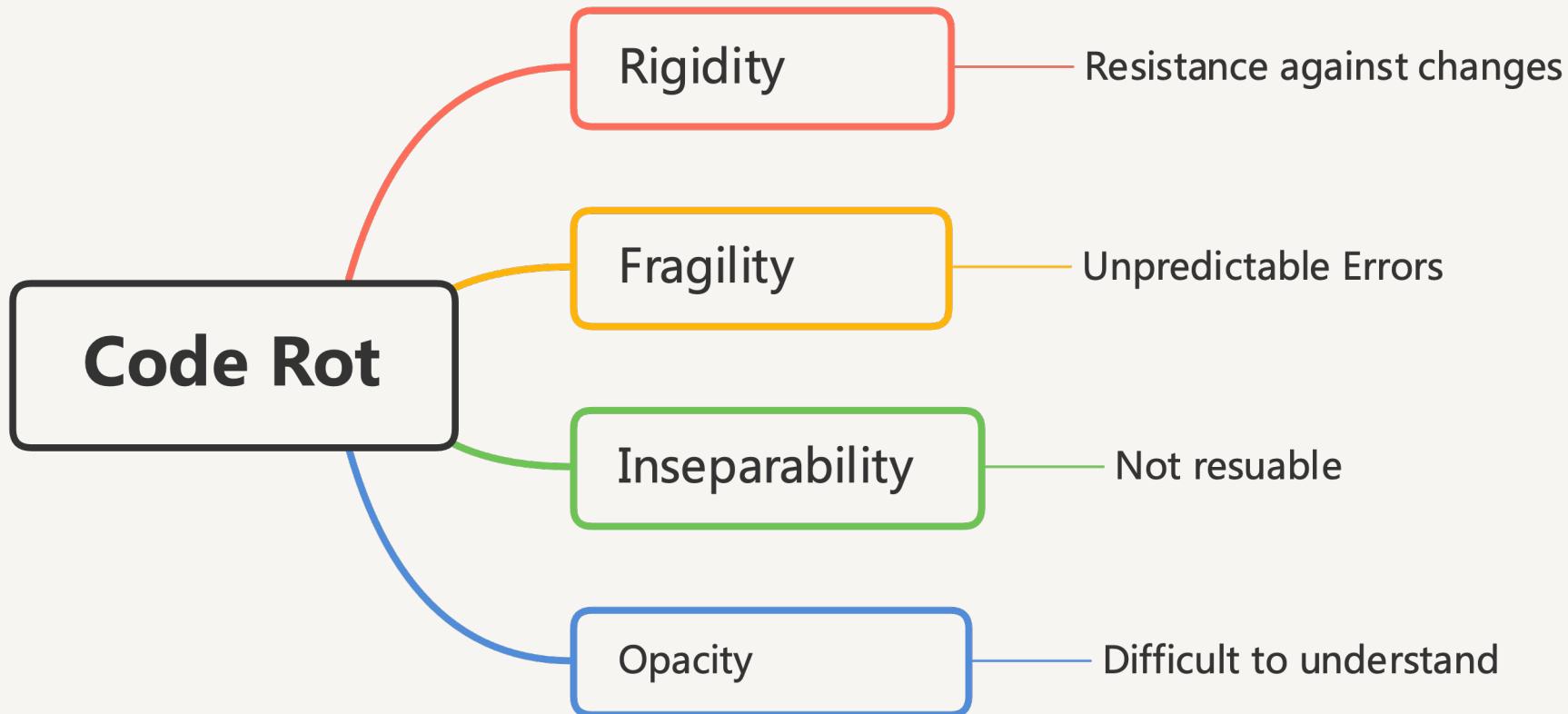
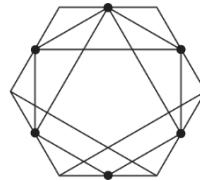
Elegant & Efficient. Should do one thing

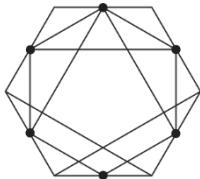


Every Routine is pretty
much what you expected



Written by someone who cares





SOLID

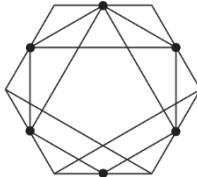
Single Responsibility Principle

Open Close Principle

Liskov Substitution Principle

Interface Segregation Principle

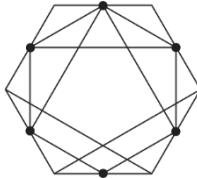
Dependency Inversion Principle



SINGLE RESPONSIBILITY PRINCIPLE



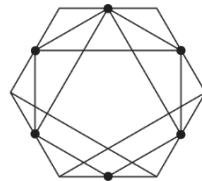
- For each class
- There should only be
- a single reason to change



SINGLE RESPONSIBILITY PRINCIPLE

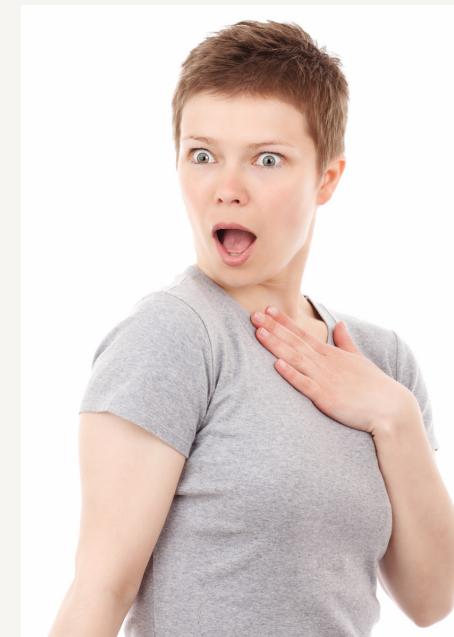


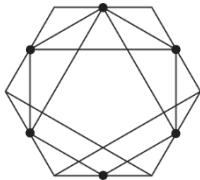
- The name is not quite correct!
- SRP does not state that each class
- may only have a single responsibility!



SINGLE RESPONSIBILITY VIOLATION

Book
-title: string
-author: string
-page: int
+print(): void
+save(): void





Marketing
Department

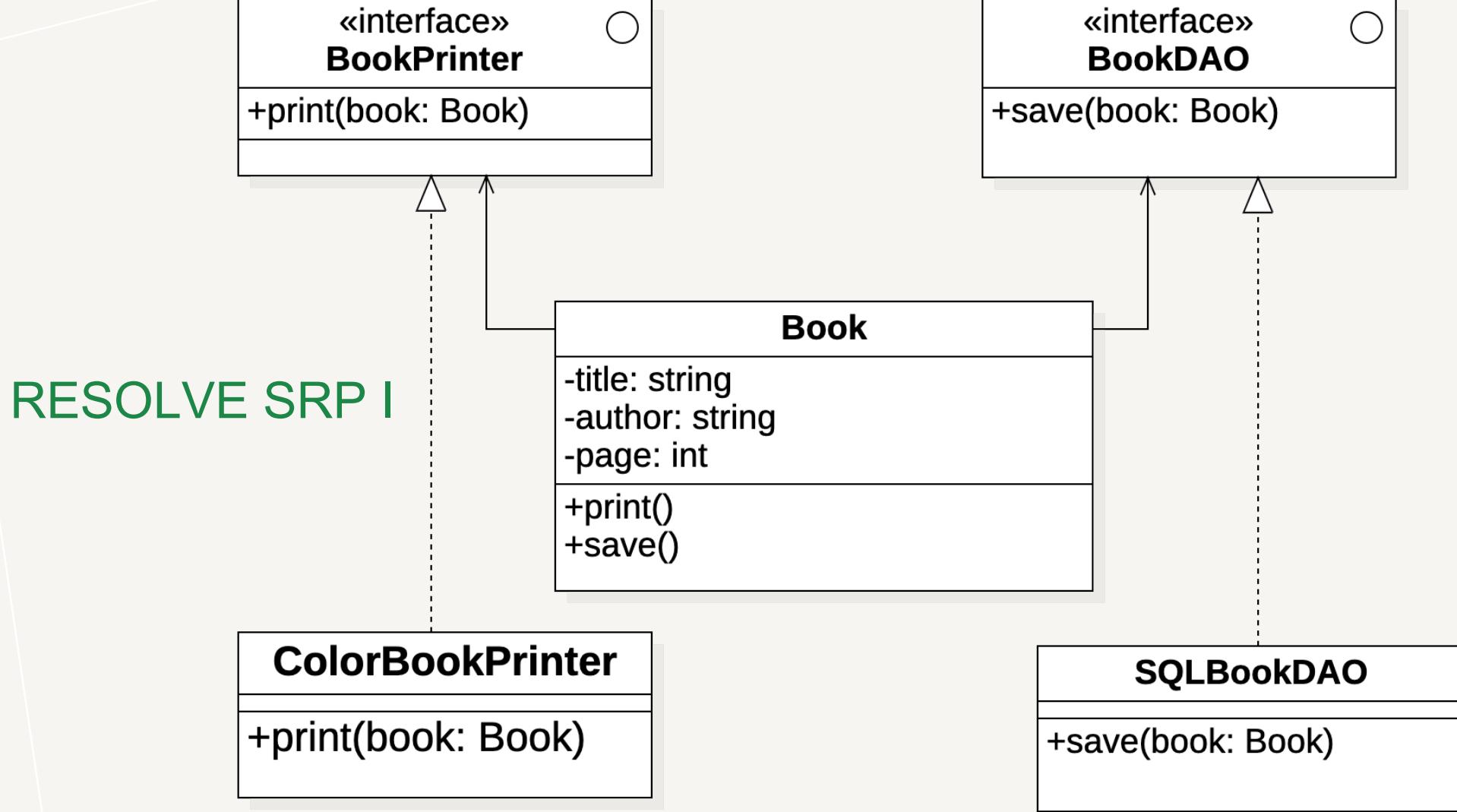
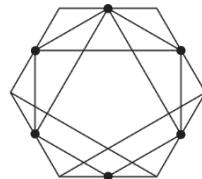
Database
Administrators

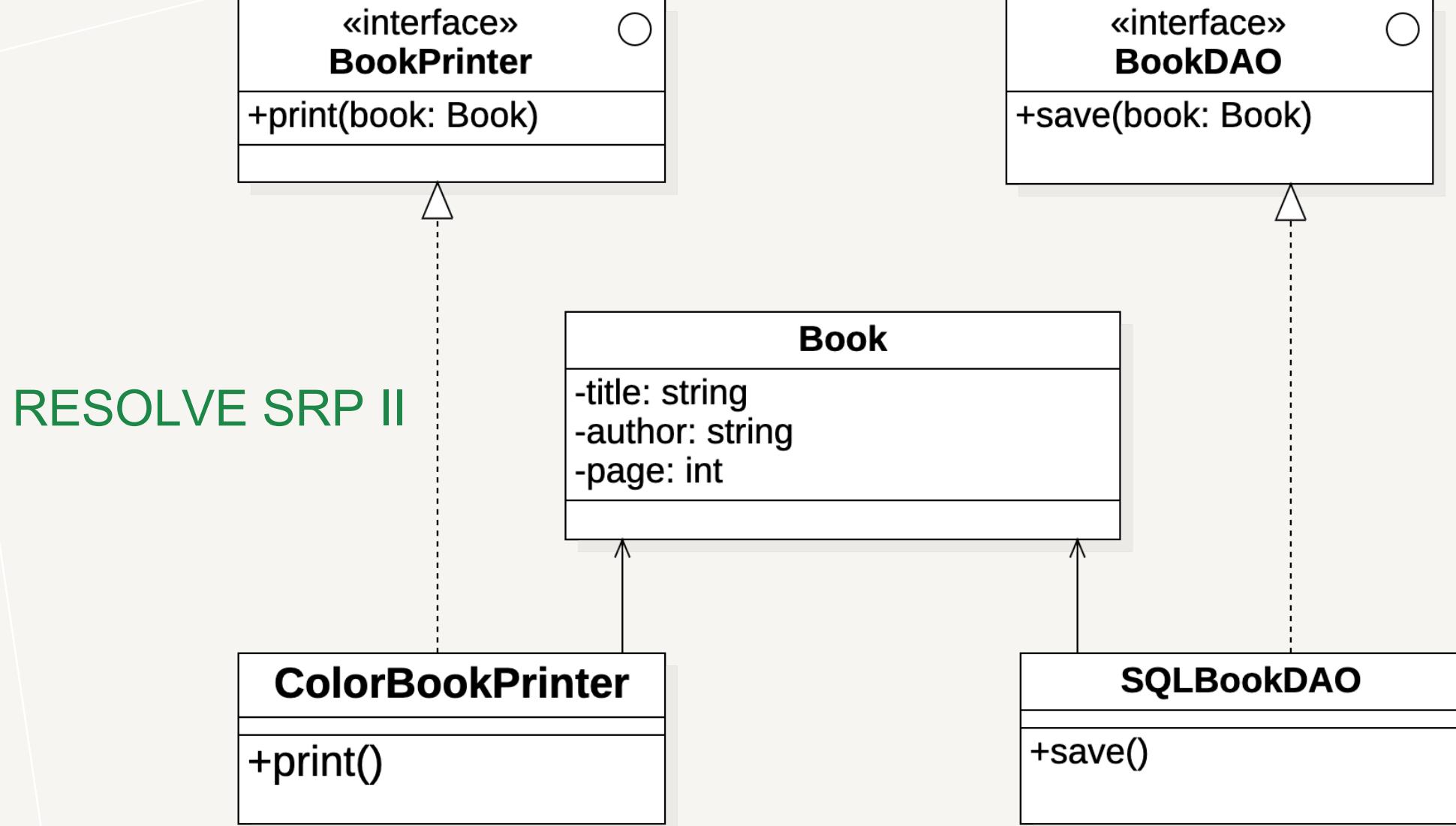
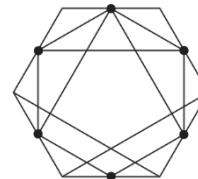


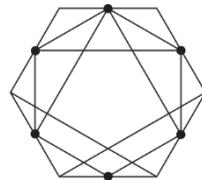
class Book

+ print

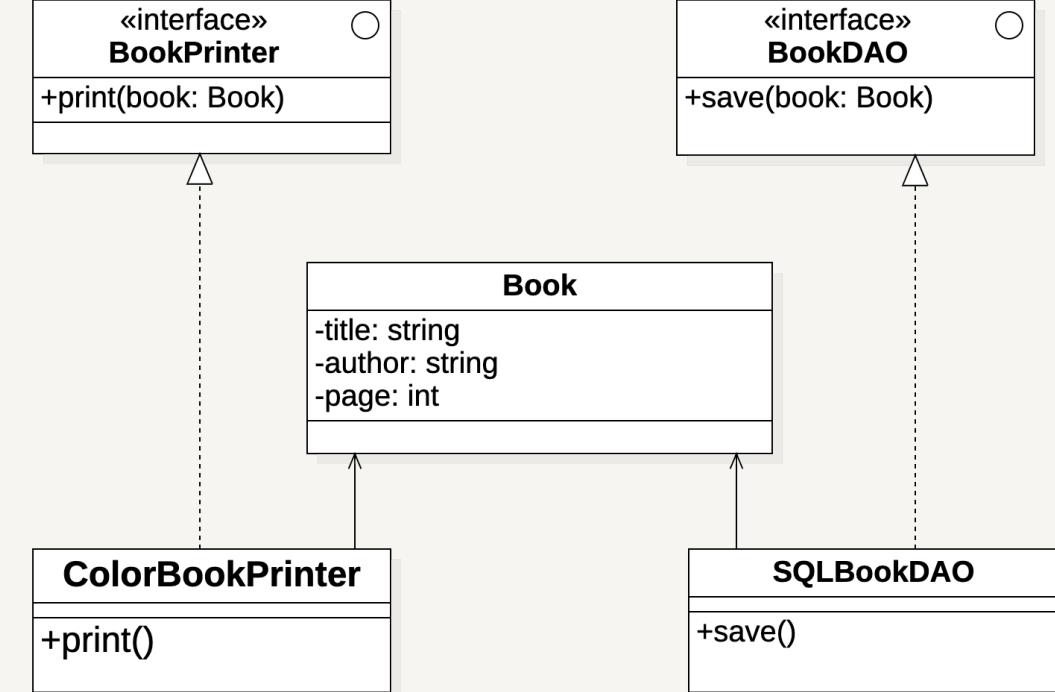
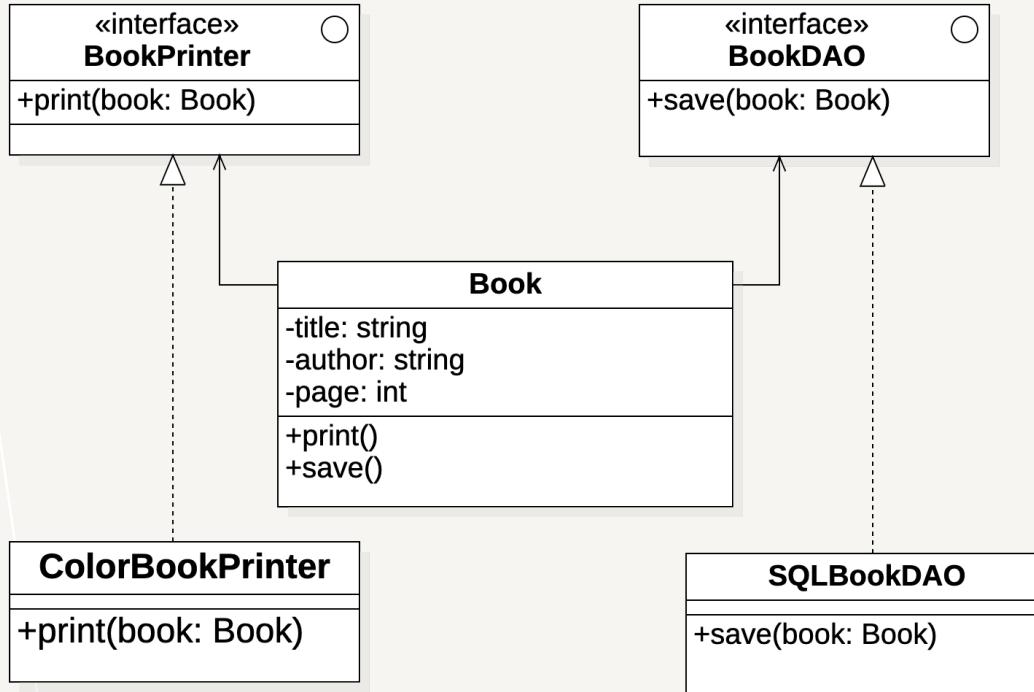
+ save

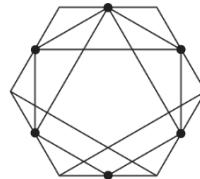






RESOLVE THE SRP VIOLATION I VS. II

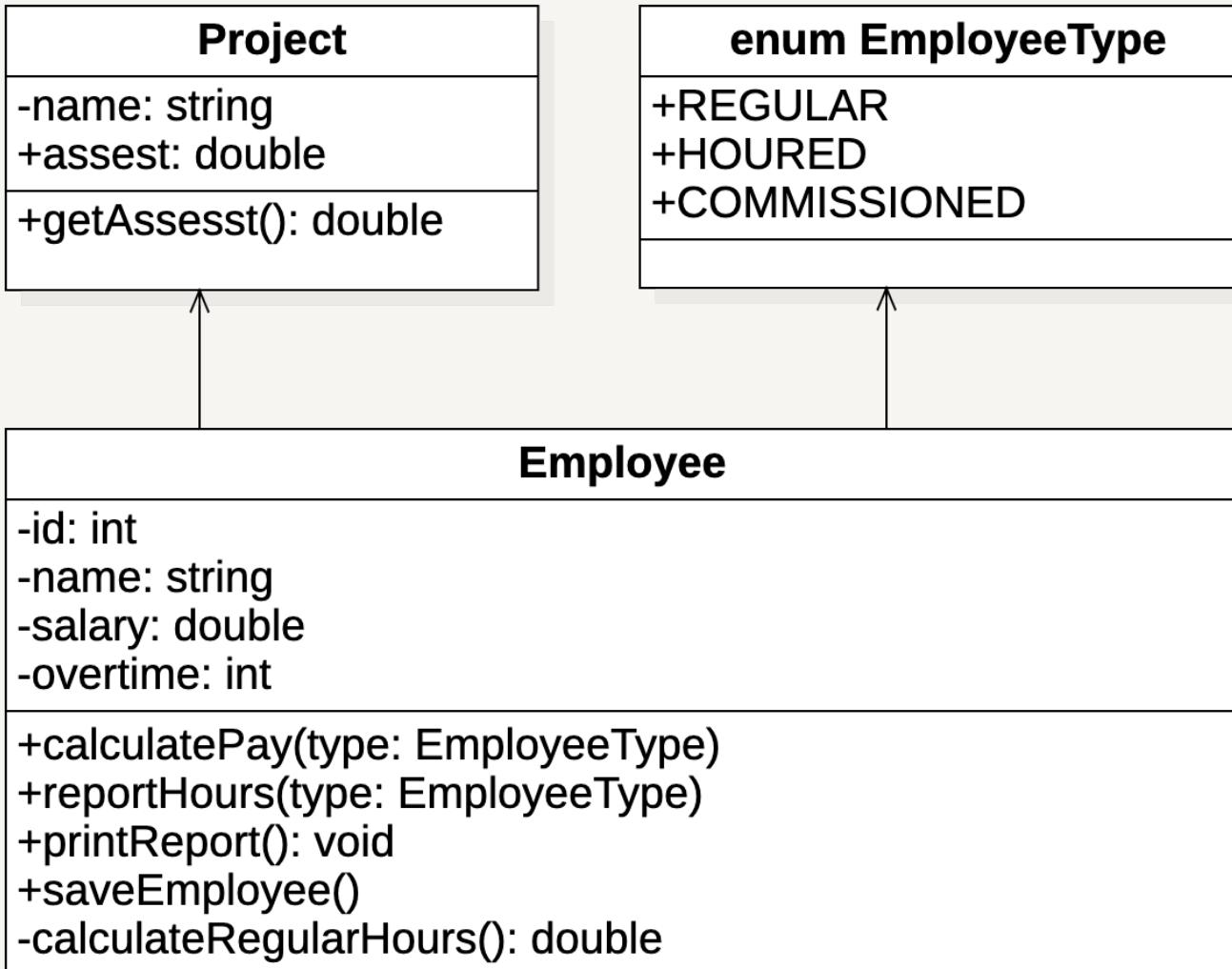
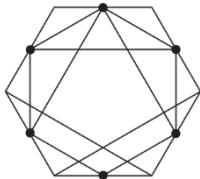


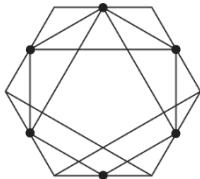


SRP => MONSTER KLASSEN VERMEIDEN!

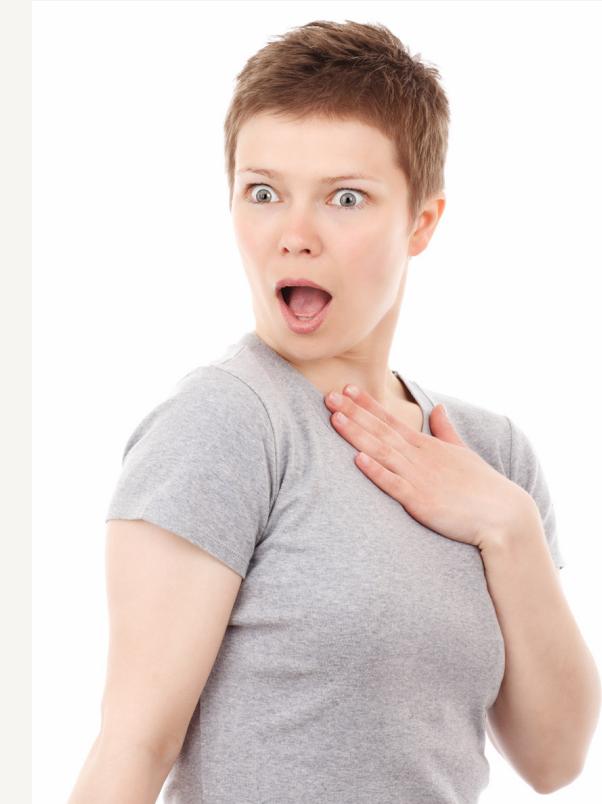
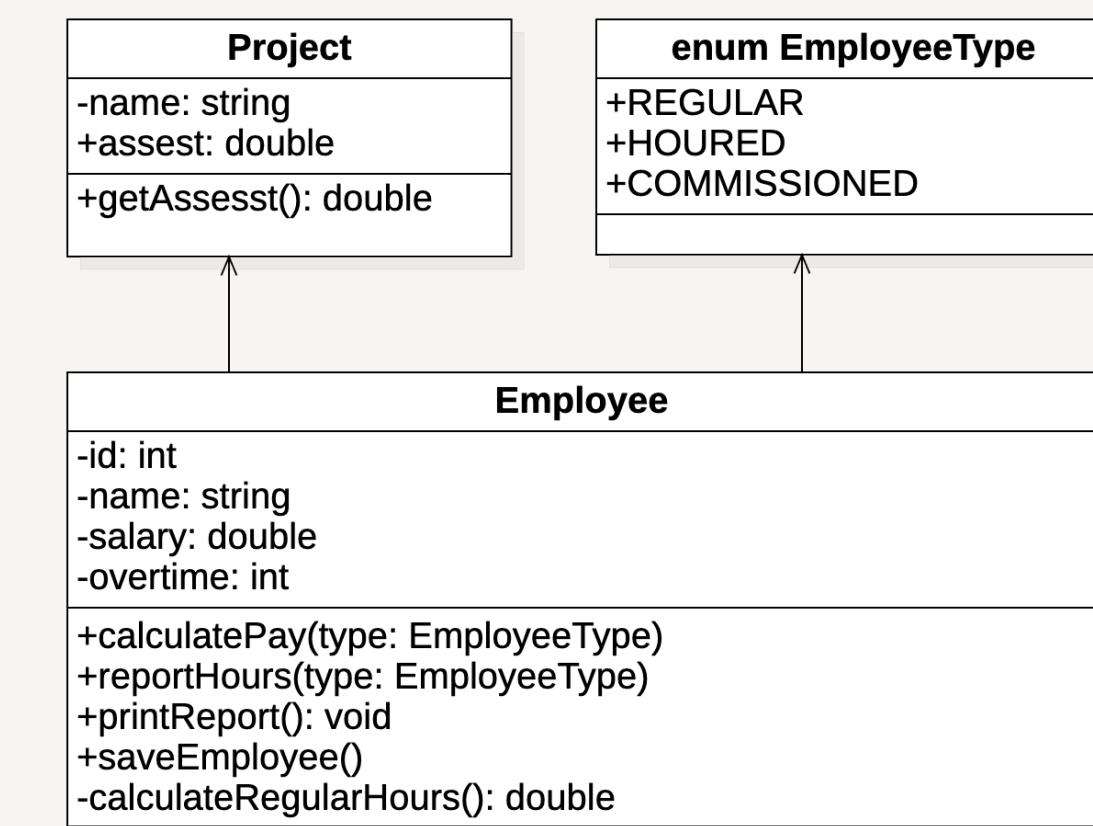


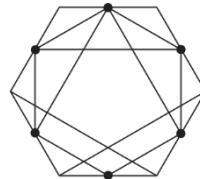
- No classes anymore
- Which know too much!
- Like the Swiss Knife!





WHAT IS WRONG?

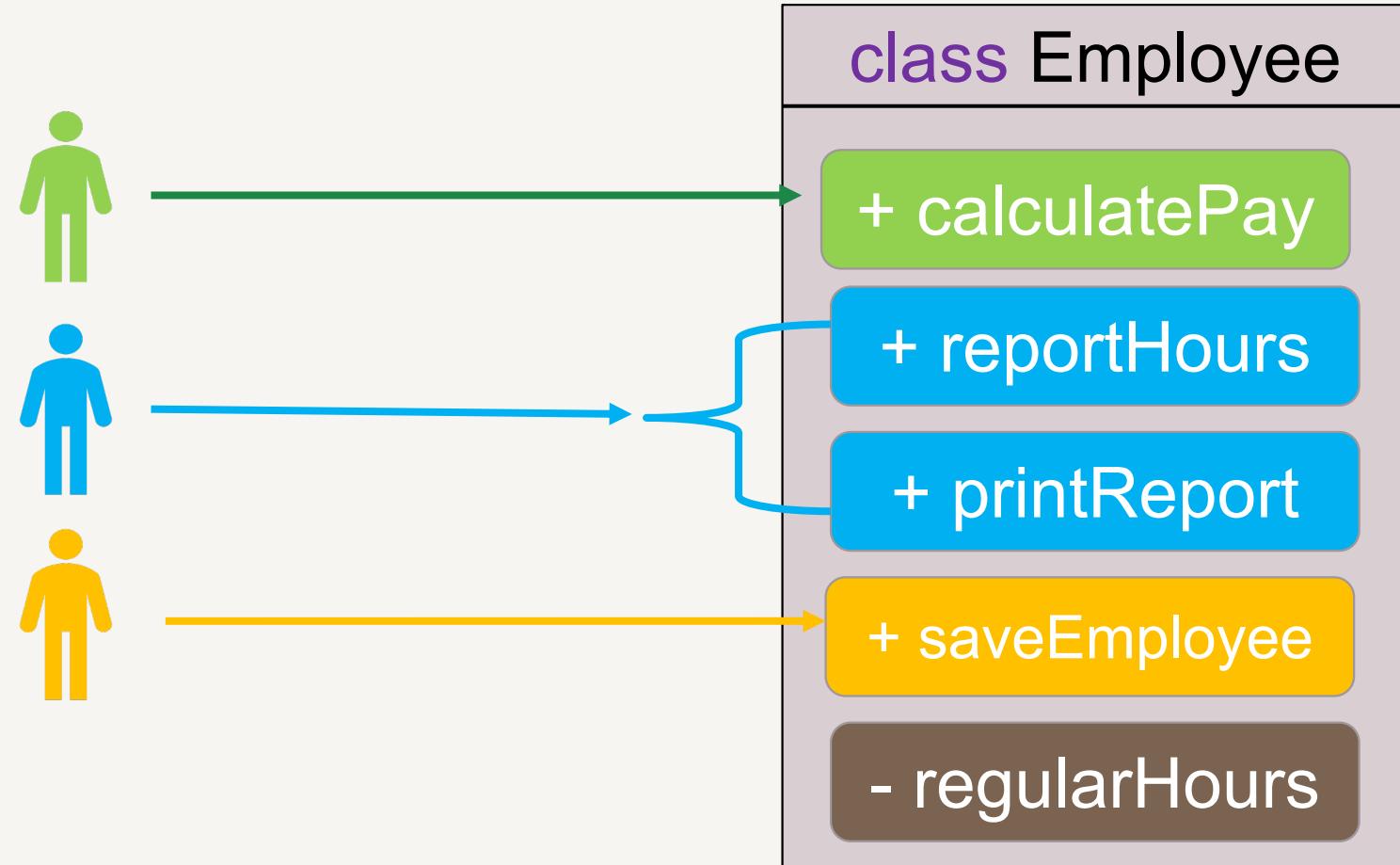


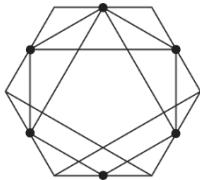


Accounting
Department

Human
Resources
Department

Database
Administrators





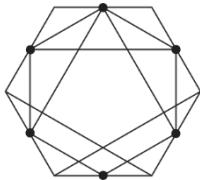
class Employee

+ calculatePay

+ reportHours

- regularHours

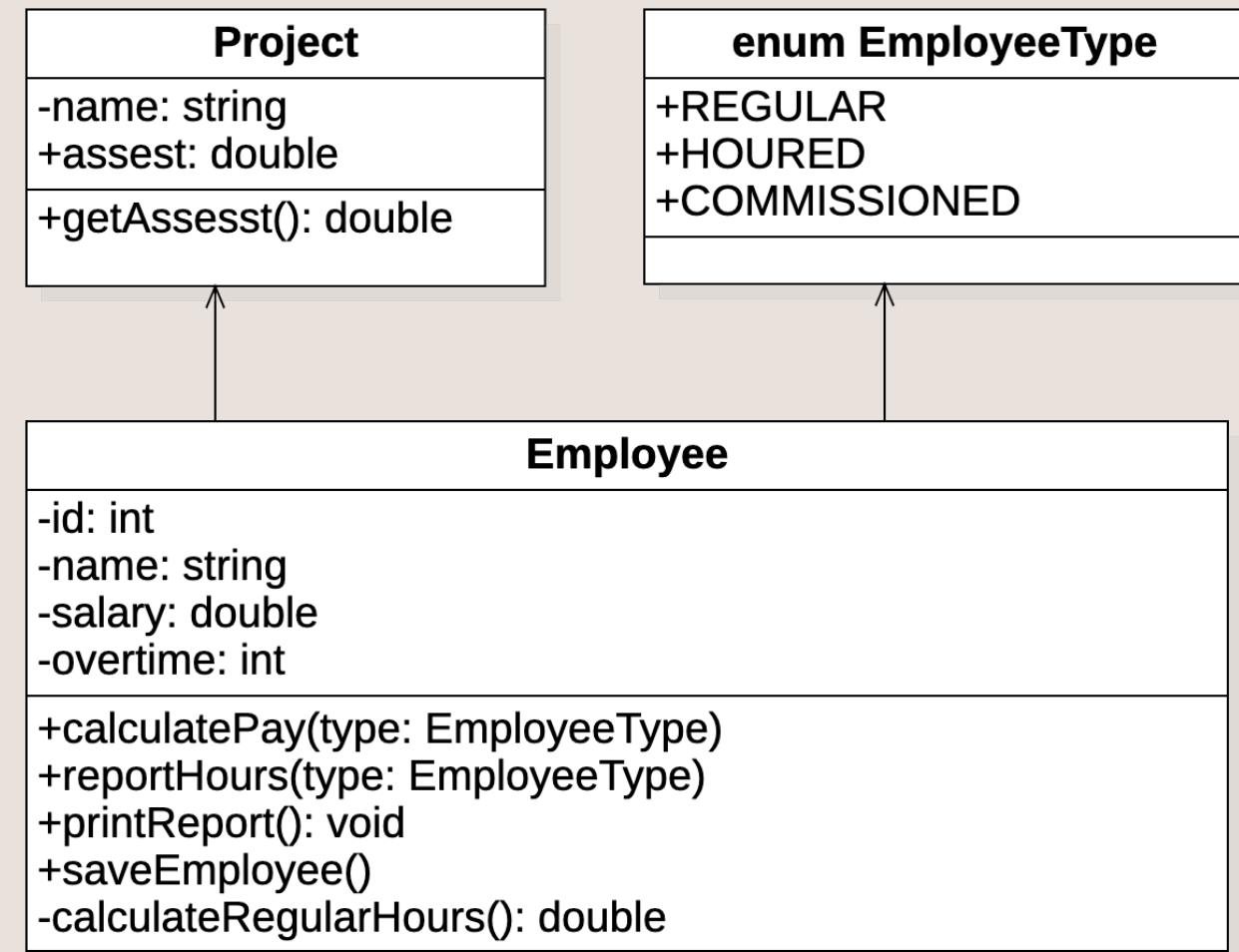
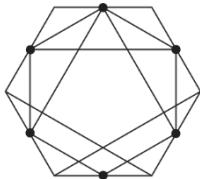
+ saveEmployee



SINGLE RESPONSIBILITY PRINCIPLE WORKSHOP

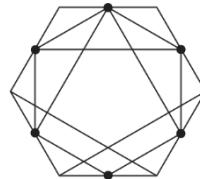


- Master the Workshop!



SRPBESTEHENDES DESIGN

- Design breaks SRP !
- Re-Design it to pass SRP
- Implements the new Design ?



DISCOUSIONÖSUNGEN UND DISKUSSIONEN





VIELEN DANK FÜR IHRE AUFMERKSAMKEIT.