

SOLID principi

1. S – Single Responsibility Principle (SRP)

Ukoliko bi na našem dijagramu klase postojala klasa *Uposlenik* u kojoj bi imali metode:

- *promijeniCijenu*
- *prihvatiNarudzbu*
- *dajMenu*
- *zavrsiNarudzbu*

Vidimo da ovu klasu možemo podijeliti u dvije klase *Konobar* i *VlasnikKafica* jer klasa *Uposlenik* 'zna previše', npr. ukoliko bi došlo do promjene u metodi *promijeniCijenu* to bi uticalo i na druge uposlenike koji ne koriste tu metodu.

2. O – Open-closed Principle (OCP)

Klase *Konobar* i *Pice* su direktno povezane. Ukoliko nakon nekog vremena bude potrebno da se doda i hrana ili nešto slično morali bi mijenjati kod, tj. klasu *Konobar*. Ovo možemo riješiti dodavanjem *KonobarInterface* interfejsa. *Konobar* klasa koristi apstrakcije, međutim objekti *Konobar* klase će korisiti objekte izvedene od *Pice* klase. Ako želimo da *Konobar* objekti koriste različite usluge (za sada *Pice*, ali možda se kasnije bude dodavala i hrana) tada novi derivati od *KonobarInterface* klase se mogu kreirati.

3. L – Liskov Substitution Principle (LSP)

Kada bi imali klasu *Uposlenik* i iz te klase naslijedili dvije klase *Konobar* i *VlasnikKafica*, LSP bi bio prekršen. Naprimjer, instanca klase *VlasnikKafica* ne može biti zamijenjena instancom klase *Uposlenik*.