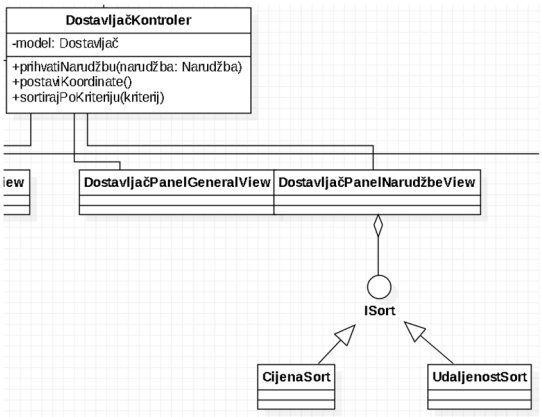
Patterni ponašanja

Strategy pattern

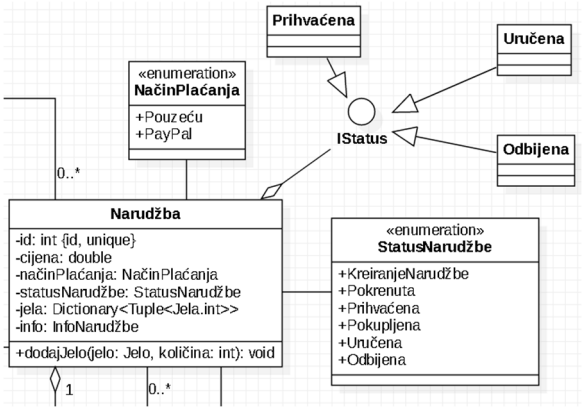
Korisno je implementirati strategy pattern da bi se različiti algoritmi koji služe za rješavanje nekog problema izdvojili u posebne klase.
Moguće bi ga bilo iskoristiti kod algoritama sortiranja proizvoda odabranog restorana.



State pattern

Predstavlja dinamičku verziju strategy patterna i postiže se promjenom podklase unutar hijerarhije klasa. U našem slučaju prikladno bi bilo implementirati ga kod promjene stanja narudžbe (pokrenuta, prihvaćena

itd.).



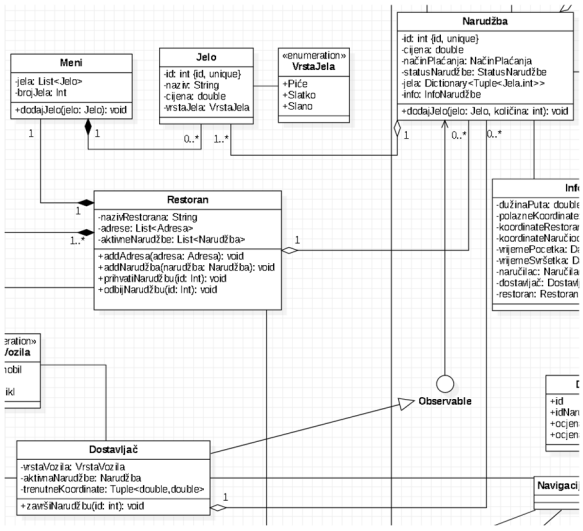
Template method pattern

Navedeni pattern omogućava izdvajanje određenih koraka nekog algoritma u upotrebi u više zasebnih podklasa pri čemu se struktura algoritma ne mijenja. Nismo pronašli potrebu za izvedbom navedenog

patterna.

Observer pattern

Observer pattern definiše one-to-many zavisnost između objekata, takvu da kada se promjeni stanje jednog objekta, zavisni objekti bivaju obaviješteni i ažurirani. U našoj aplikaciji klasa Narudžba bi imala ulogu Observera te, u zavisnosti od lokacije dostavljača, ažurirala status narudžbe.



Iterator pattern

Potreba za iterator patternom se javlja kod sekvencijalnog pristupa kolekciji objekata bez otkrivanja interne strukture objekata prisutnih u kolekciji i poznavanja strukture kolekcije. Hipotetska izvedba ovog patterna bi bila implementirana za klasu Meni, tačnije sekvencijalni pristup kolekciji jela nekog

restorana.

Chain of responsibility pattern

Pattern omogućuje objektu da pošalje instrukciju bez znanja koji objekat će primiti i obraditi poslanu instrukciju. Svaki objekat uvezan
u lanac(chain) može obraditi instrukciju, proslijediti je drugom objektu ili oboje. Kad korisnik kreira narudžbu mogla bi se, kao kod
primjera sa CV-em sa laboratorijske vježbe, provesti obrada

narudžbe.

Mediator pattern

Izvedba mediator patterna definiše objekat koji enkapsulira interakciju skupa objekata. Rješava se problem tight coupling-a jer se izbjegava
direktna komunikacija između objekata, te omogućava presretanje i modifikaciju same interakcije odvojeno od objekata.
Mediator pattern je poželjno izvesti kod bilo kakve upotrebe formi (log in, sign up...).