**Princip pojedinacne odgoornosti** – sa dijagrama je jasno uocljivo da je ovaj princip zadovoljen jer se klase brinu po pojedinacnom stvarima (samo o svojim atributima). To mozemo vidjeti na primjeru klase *Karta* ili klase *Atrakcija* gdje se metode navedenih klasa brinu samo o argumentima pripadajucee klase.

**Otvoreno zatvoren princip –** ovaj princip je primjenjen kreiranjem klase *Mapa* gdje je moguca nadogranja (dodavanjem dodatnih atributa za detaljniji opis mape) a da pri tome ne dolazi do modifikacije klase *Atrakcija*

**Liskov princip zamjene -** ovim dizajnom smo vodili računa da sve klase koje su vezi generalizacije sa nekom drugom klasom očuvaju osobinu korektnog naslijeđivanja. To možemo vidjeti na primjeru klase *Karta*. Jasno je da se klase *TipA* i TipB može zamijeniti osnovnim tipom iz kojeg su naslijeđene zbog očuvanja jednakosti atributa i metoda koje su svim klasama koje učestvuju u toj vezi.

**Princip inverzije ovisnosti –** s obzirom da su se interfejsi i apstraktne klase koristile, ovim smo zadovoljili princip inverzije ovisnost. Na ovaj nacin smo postigli da buduce promjene u sistemu budu jednostavnije i trebaju se izvrsiti na manje mjesta cime postizemo smanjenje krutosti koda.

**Princip izoliranja interfejsa –** ne postoji interfejs ili apstraktna klasa sa velikim brojem metoda. Na ovaj nacin smo postigli da prilikom nasljedjivanja, klase ne nasljedjuju i metode koje joj nisu potrebne vec su razdvojene po funkcionalnostima koje trebaju da izvrsavaju.