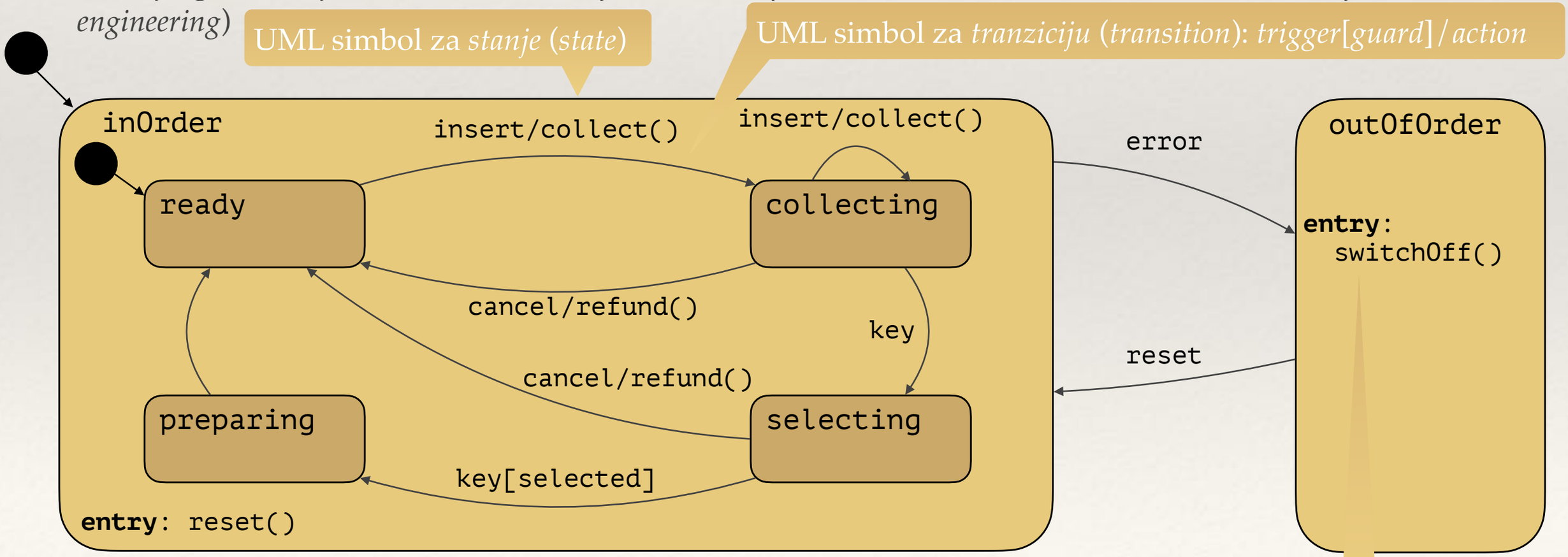
Klasa kao realizacija softverske mašine

- U ovakvim sistuacijama, za modelovanje ponašanja objekata neke klase koristi se *mašina stanja* (*state machine*) — koncept višeg nivoa apstrakcije koji modeluje ponašanje, uz mogućnost predstavljanja modela dijagramom
- Mašine stanja su napredan softverski koncept koji potiče od konačnih automata, ali dodaje mnoge druge, naprednije koncepte, kao što su hijerarhijska (ugnežđena) stanja, ulazne i izlazne akcije i mnoge druge
- Nažalost, nijedan klasičan OO programski jezik ne podržava mašine stanja: ovakvo ponašanje se mora implementirati ručno, što je naporno i podložno greškama
- Rešenje: generisanje koda iz modela - *softversko inženjerstvo zasnovano na modelima* (*model-based software engineering*)



Klasa kao realizacija softverske mašine

- Pretpostavimo da pravimo softver za neku arkadnu igricu, u kojoj se različiti objekti, tzv. likovi, kreću uporedo po igračkom polju. Kako da implementiramo to nezavisno i uporedo kretanje koje se (makar prividno) dešava istovremeno?
- Ideja:
  - svaki lik (*character*) predstavimo objektom, po potrebi određene izvedene klase
  - svaki lik implementira operaciju koja izvodi jedan korak (*step*) svog kretanja; ova operacija treba da bude kratka i da izvrši jedan elementaran pomeraj, po što je moguće kraći
  - jedna glavna petlja "proziva" sve likove na igračkom polju i svakome daje da se pomeri za po jedan korak
- Na ovaj način možemo da stvorimo privid uporednog kretanja objekata kao *aktivnih* entiteta