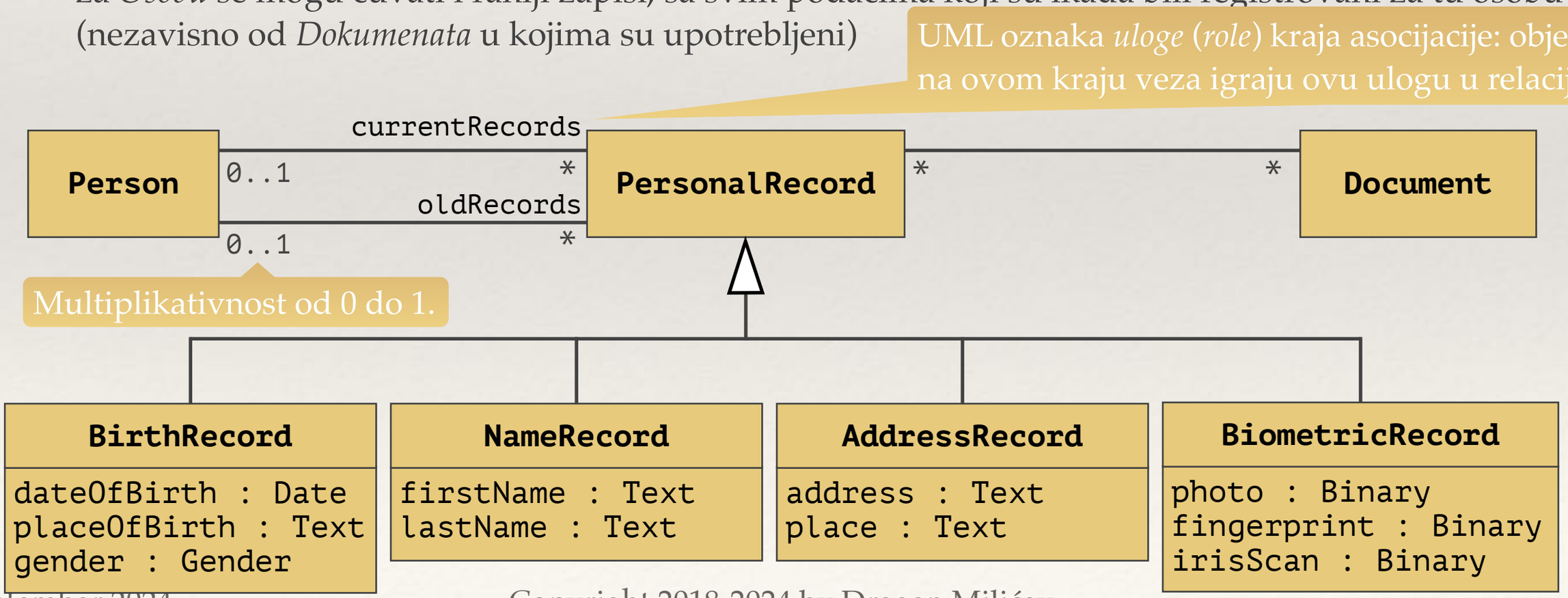
Klasa kao realizacija apstrakcije

- Ideja boljeg rešenja: uvesti koncept *Personalnog zapisa* (*Personal Record*) koji sadrži određene lične podatke:
  - umesto da sadrže kopije, *Osoba* i *Dokument* referenciraju (vezani su za) *Personalne zapise*: sve dok se podatak ne menja, ovi objekti vezani su za iste zapise
  - prilikom promene nekog podatka, pravi se nova kopija zapisa u kom se taj podatak menja
  - kako bi se dodatno smanjilo kopiranje, podaci su grupisani u različite specijalizacije zapisa, po tome koliko se često i da li se zajedno ili odvojeno menjaju
  - za *Osobu* se mogu čuvati i raniji zapisi, sa svim podacima koji su ikada bili registrovani za tu osobu (nezavisno od *Dokumenata* u kojima su upotrebljeni)



Copyright 2018-2024 by Dragan Milićev

Klasa kao realizacija softverske mašine

- U nekim slučajevima, reakcija objekta zavisi ne samo od *pobude* (pozvane operacije), nego i od *predistorije* pobuda, odnosno od trenutnog *stanja* tog objekta
- Primer: softver koji upravlja automatom za prodaju napitaka - reakcija na pritisak tastera za izbor artikla zavisi od toga šta se prethodno dogodilo
- Automat može da bude *ispravan* ili u stanju *van upotrebe* (npr. zbog problema ili punog kontejnera za novac) itd.
- Kako bi se ovakvo ponašanje implementiralo? Na primer:

```
void VendingMachine::insertCoin (int value) {
  if (this->inOrder)
    this->amount += value;
  else
    if (!this->isRefundBarrierOpen)
      this->openRefundBarrier();
}

void VendingMachine::selectArticle (int code) {
  if (this->inOrder)
    if (this->articles[code]>0 && this->prices[code]<=this->amount) {
      this->amount -= this->prices[code];
      this->deliverArticle(code);
    }
}

...
```

- Problem: kod složenijeg ponašanja, kod postaje nepregledan, težak za razumevanje, težak za održavanje (izmene, proširenja) i podložan greškama

Copyright 2018-2024 by Dragan Milićev