Relacije i zavisnosti između klasa

- Umesto da objekat-klijent sam kreira potrebne objekte-servere sa kojima sarađuje, ili sam traži pristup do njih i identifikuje ih (npr. pozivom nekih usluga drugih klasa), praksa pokazuje da je bolji pristup da neko drugi, i to onaj ko koristi posmatranu klasu, spolja, *injektira* te veze, odnosno definiše zavisnosti od servera
- Ovo se može uraditi na sledeće načine:
  - kroz konstruktor posmatrane klase, ali samo ako su te veze obavezne (minimalna multiplikativnost je veća od 0, odnosno objekat *mora* biti vezan):

```
class Controller {
public:

  Controller (Reader* reader, Translator* translator);
  …

};

Controller::Controller (Reader* reader, Translator* translator) {
  this->myReader = reader; this->myTranslator = translator;
}
```

  - kroz posebnu operaciju klase kojom se ova veza uspostavlja, ako se takva veza može menjati tokom života objekta:

```
class Controller {
public:

  void setReader (Reader* reader);
  void setTranslator (Translator* translator);
  …

};

void Controller::setReader (Reader* reader) {
  this->myReader = reader;
}
```

  - ili oba, ako važi i jedno i drugo:

```
Controller::Controller (Reader* reader, Translator* translator) {
  this->setReader(reader); this->setTranslator(translator);
}
```

Relacije i zavisnosti između klasa

- Ovaj princip naziva se *injekcija zavisnosti* (*dependency injection*)
- Na ovaj način se postiže bolja fleksibilnost, jer se objekti posmatrane klase mogu upotrebljavati na različite načine i u različitim kontekstima, odnosno mogu se menjati konfiguracije njihovih veza sa drugim objektima, bez izmena njihovih klasa
- Drugim rečima, odgovornost za povezivanje objekata u *kolaboraciju* (*collaboration*), dodeljuje se nekoj drugoj klasi — opet je važno dobro identifikovati i raspodeliti odgovornosti
- Na primer, u zavisnosti od situacije, dati objekat klase *Controller* može se povezati sa jednim ili drugim objektom klase *Reader* i *Translator*, ili čak njihovim specijalizacijama; on neće trpeti nikakvu izmenu i ne zavisi od svega toga, pošto nije ni svestan tih različitih konfiguracija u kojima je upotrebljen:

```
class FileReader : public Reader {…};

class ExtendedTranslator : public Translator {…};

…
Reader* simpleReader = new Reader(…);
FileReader* fReader = new FileReader(…);
ExtendedTranslator* translator = new ExtendedTranslator(…);

Controller* ctrler1 = new Controller(simpleReader,translator);
Controller* ctrler2 = new Controller(fReader,translator);
```