Relacije i zavisnosti između klasa

- Veze između objekata mogu biti kratkotrajne, tj. takve da ne nadživljavaju trajanje metode u kojoj se koriste, tako što posmatrana operacija:
  - kao argument prima pokazivač na objekat-server:

```
void Controller::main (Reader* myReader, Translator* myTranslator) {
  Command* cmd = myReader->read();
  while (cmd!=nullptr) {
    myTranslator->translate(cmd);
    int out = cmd->getOut();
    mySMs[out]->perform(cmd);
  }
}
```

  - dobija pristup do datog objekta putem neke druge operacije (npr. kao njenu povratnu vrednost), ili traži takav objekat na neki drugi način:

```
void Controller::main () {
  Command* cmd = myReader->read();
  while (cmd!=nullptr) {
    myTranslator->translate(cmd);
    int out = cmd->getOut();
    mySMs[out]->perform(cmd);
  }
}
```

Relacije i zavisnosti između klasa

- Sa druge strane, veze mogu da nadžive izvršavanje operacije, pa se moraju skladištiti unutar objekta:

```
class Controller {
  …
private:
  Reader* myReader;
  Translator* myReader;
  StackMachine* mySMs[…];
};
```

- Ukoliko je gornja granica multiplikativnosti odgovarajućeg kraja asocijacije veća od 1, za implementaciju se mora koristiti neka kolekcija:

```
class Teacher {
public:
  …
  void addCourse(Course*);
  void removeCourse(Course*);

private:
  list<Course*> myCourses;
  …
};
```