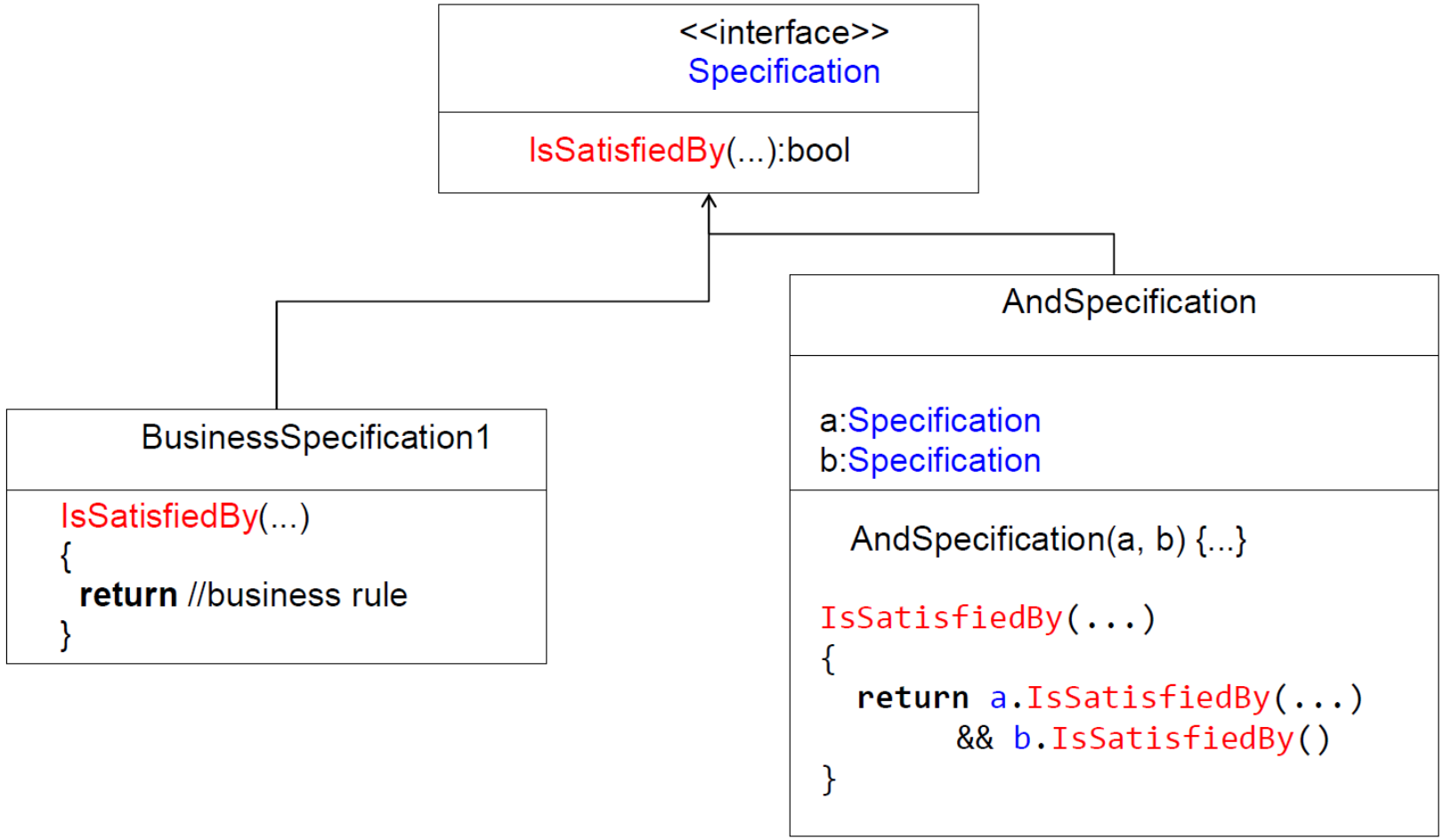
">

Specification

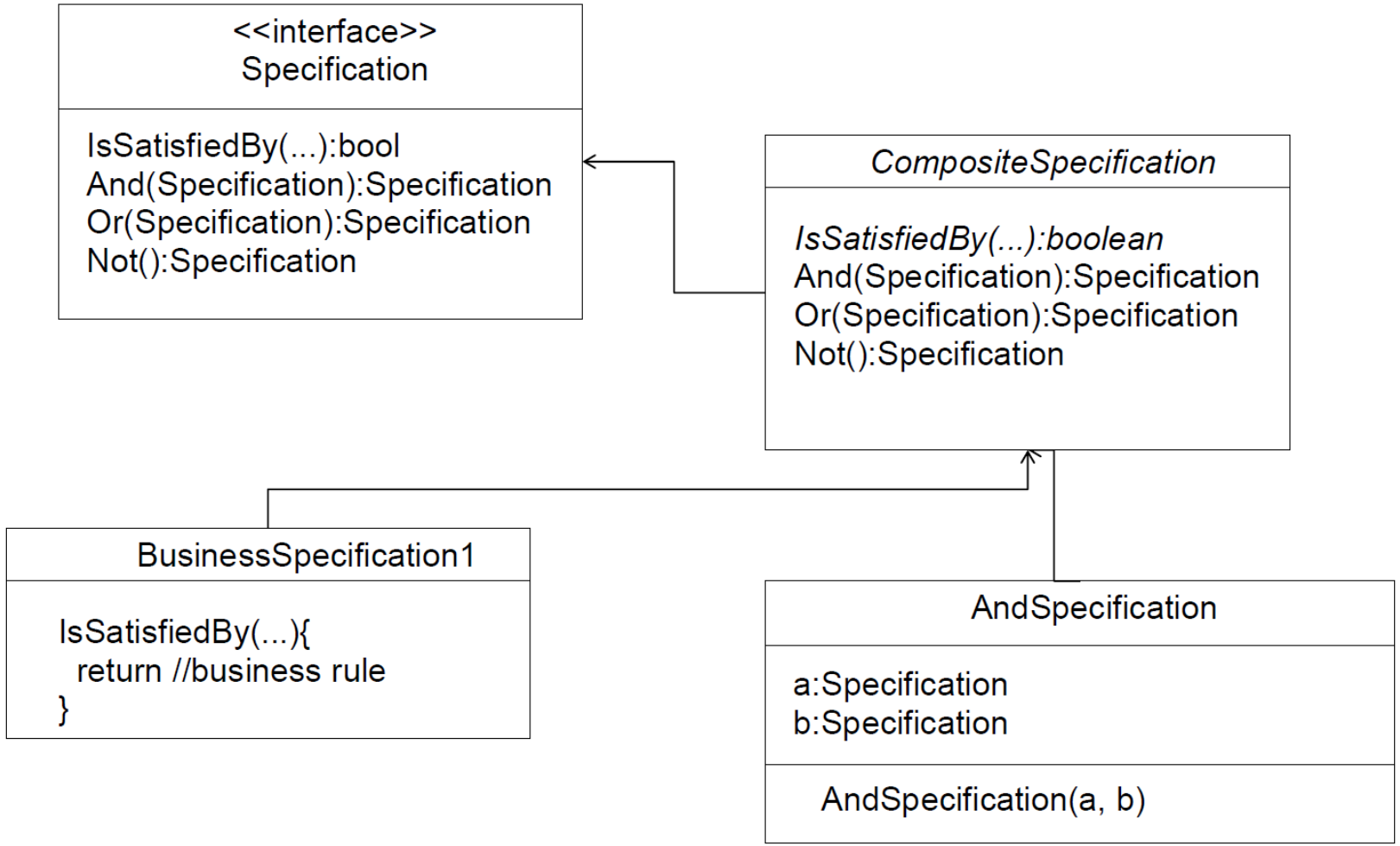
Specification

- wzorzec behawioralny
- pozwala na składanie logiki biznesowej z mniejszych elementów
  - logika polega na udzieleniu odpowiedzi czy pewien obiekt spełnia specyfikację
- pomiędzy elementami mogą zachodzić relacje logiczne
  - jeden z Building Block w Domain Driven Design

Specification - struktura



Specification - struktura



Specification - przykład

```
public interface ISpecification<T>
{
  bool IsSatisfiedBy(T candidate);
  ISpecification<T> And(ISpecification<T> other);
  ISpecification<T> Or(ISpecification<T> other);
  ISpecification<T> Not();
}
```

```
ISpecification<User> grandpaRockersSpecification =
   new GenderSpecification(Gender.Male)
    .And(new AgeSpecification(60, 100))
    .And(new BikeSpecification(MotorbikeType.Suzuki).Not())
    .And(
      new BikeSpecification(MotorbikeType.HarleyDavidson)
      .Or(new BikeSpecification(MotorbikeType.Honda)));
```

Specification - przykład

- system decyduje o udzieleniu kredytu
- decyzja jest podejmowana na podstawie zestawu wielu kryteriów
  - zestawy kryteriów różnią się u różnych klientów (w różnych wdrożeniach)
    - ale część z nich jest wspólna
- można zauważyć, że istnieje pewna pula kilkudziesięciu kryteriów
  - w danym wdrożeniu używa się kilkunastu kryteriów z puli

Specification - FluentValidation

```
public class CustomerValidator : AbstractValidator<Customer> {
  public CustomerValidator() {
    RuleFor(x => x.Surname).NotEmpty();
    RuleFor(x => x.Forename).NotEmpty().WithMessage("Please specify a first name");
    RuleFor(x => x.Discount).NotEqual(0).When(x => x.HasDiscount);
    RuleFor(x => x.Address).Length(20, 250);
    RuleFor(x => x.Postcode).Must(BeAValidPostcode).WithMessage("Please specify a valid postcode");
  }

  private bool BeAValidPostcode(string postcode) {
    // custom postcode validating logic goes here
  }
}
```

```
CustomerValidator validator = new CustomerValidator();

validator.ValidateAndThrow(customer);
// or
ValidationResult results = validator.Validate(customer);
string allMessages = results.ToString();
// or
ValidationResult results = validator.Validate(customer);
if (!results.IsValid) {
  foreach(var failure in results.Errors) {
    Console.WriteLine("Property " + failure.PropertyName + " failed validation. Error was: " + failure.ErrorMessage)
  }
}
```