Object oriented design

Giorgio Brajnik

giorgio.brajnik@uniud.it

Dec 2019

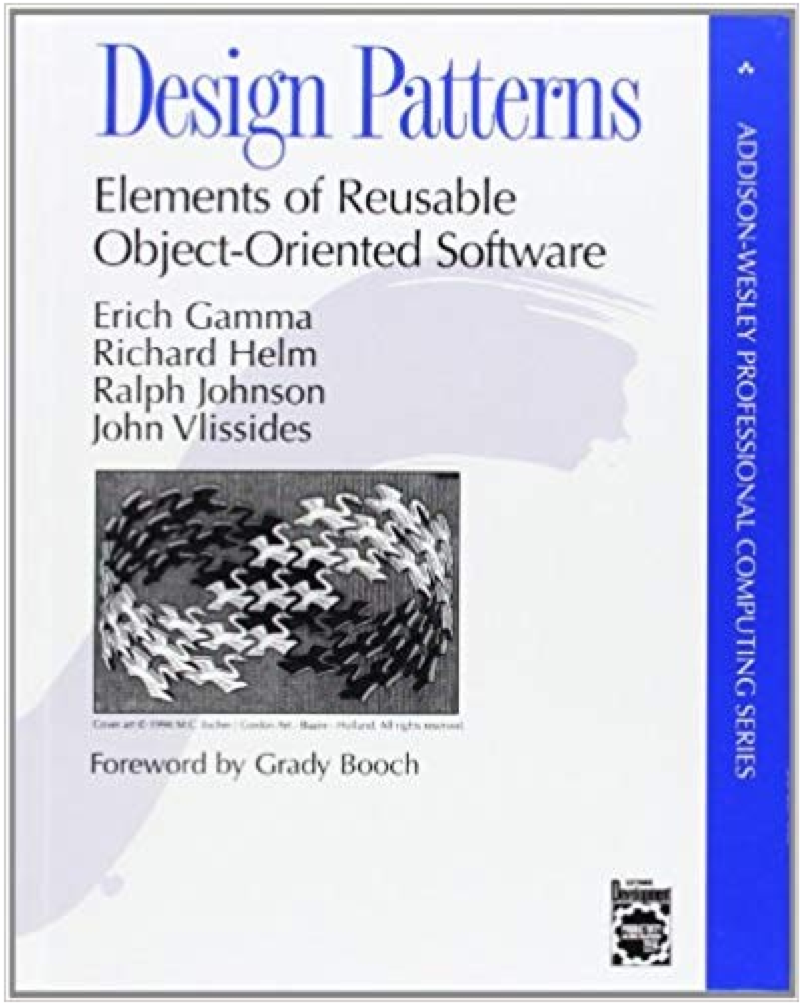
Design Patters

- structures of types (classes & interfaces)
- and appropriate protocols

- used very often
- in different domains and contexts
  - to create objects: factory, abstract factory, builder, ...
  - to create certain relationships: decorator, …
  - behavioral: iterator, observer, ...

Useful book

- Gamma, Helm, Johnson, Vlissides, 1994
  Design Patterns
  Addison-Wesley



Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

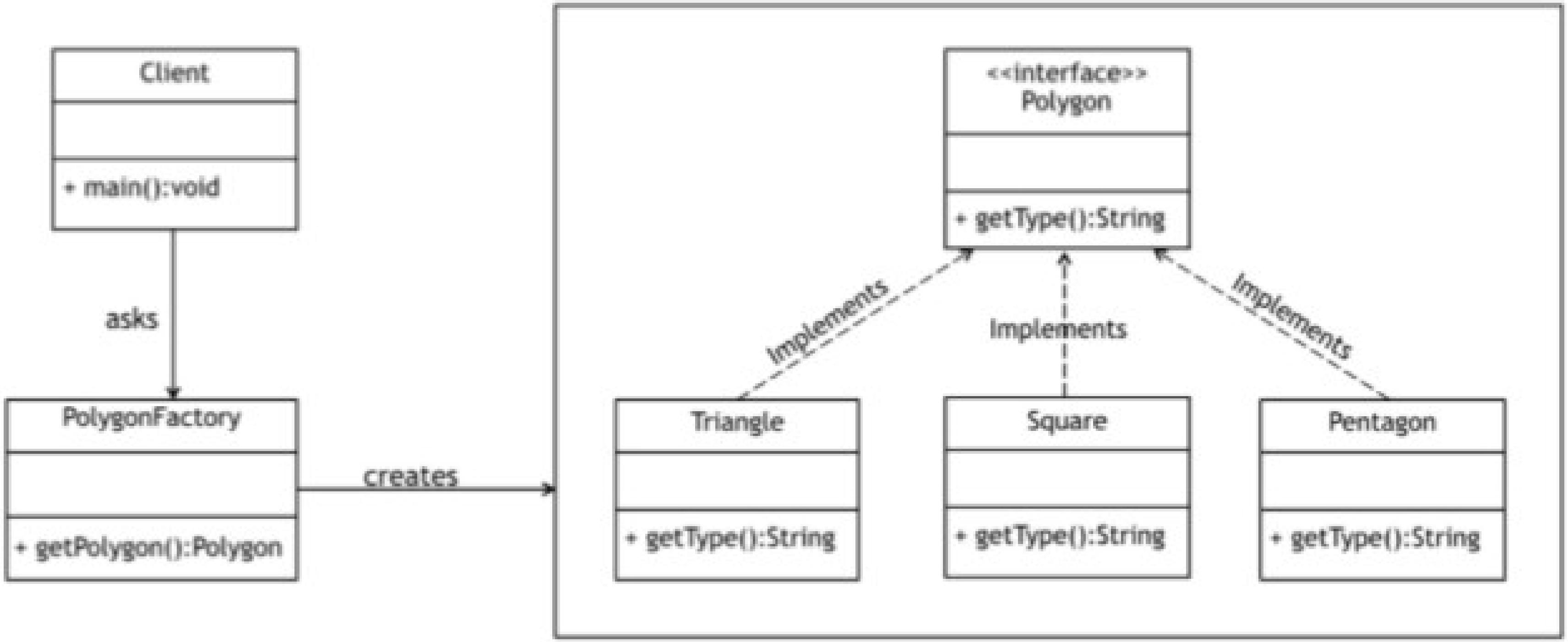
Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

- see also
  https://en.wikipedia.org/wiki/Software_design_pattern

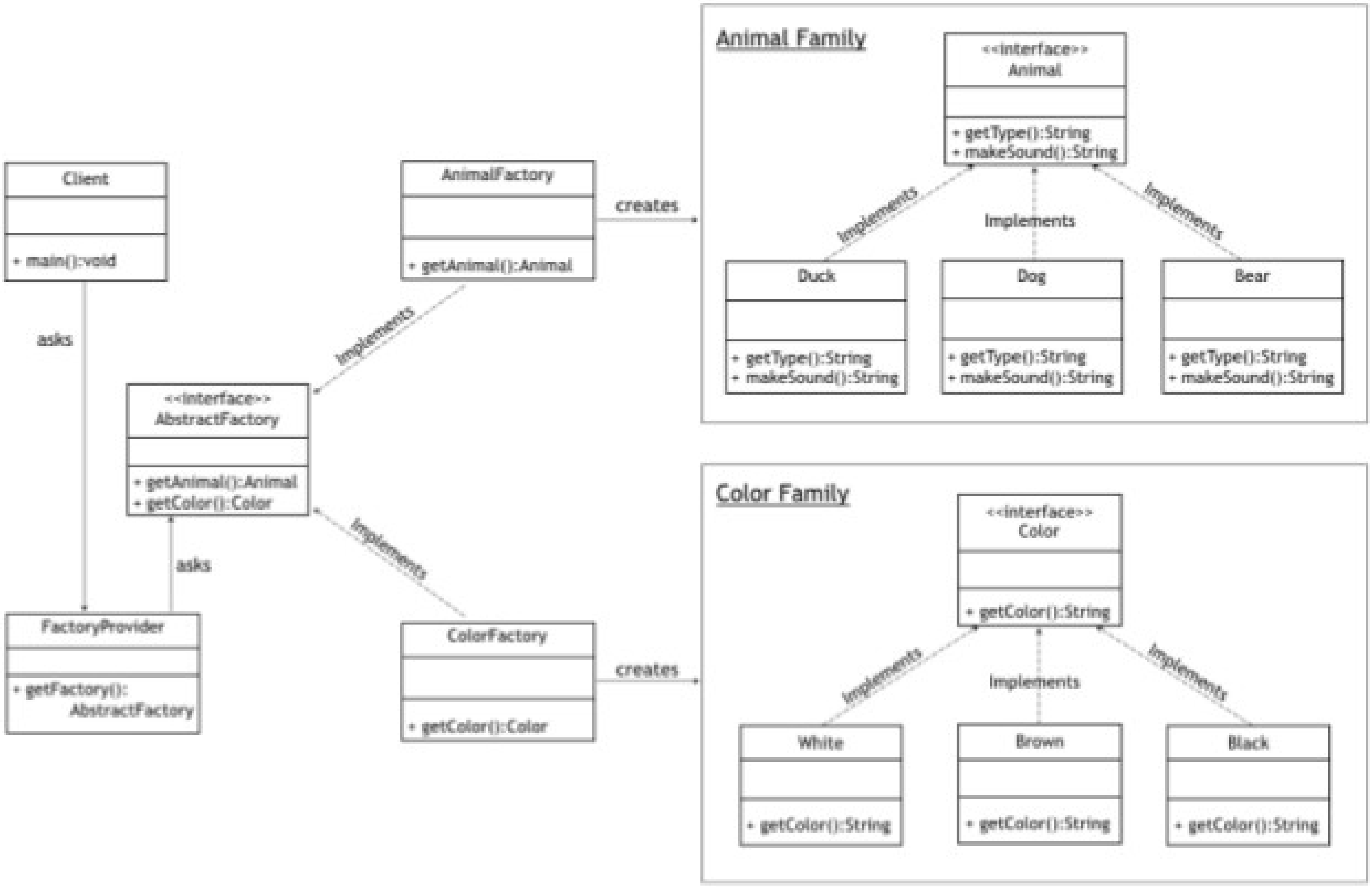
Factory

- how to hide details on how to create instances?



Abstract Factory

- how to hide details on how to create families of different instances?

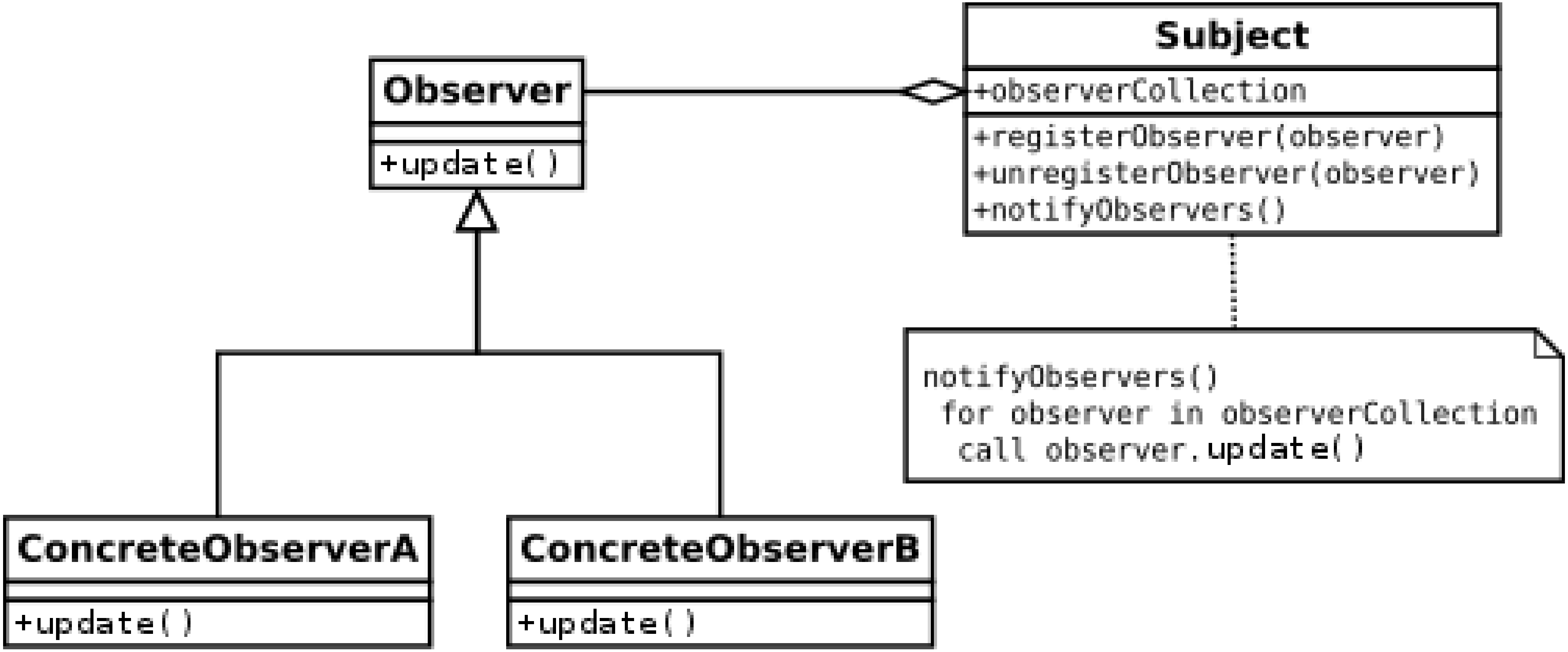


Builder

- how to provide flexibility when instantiating a type that has many attributes?
  - the constructor accepts a “builder”
  - the builder has a constructor with mandatory fields
  - the builder has methods to set other fields
  - the builder methods use a “fluent” style

Observer

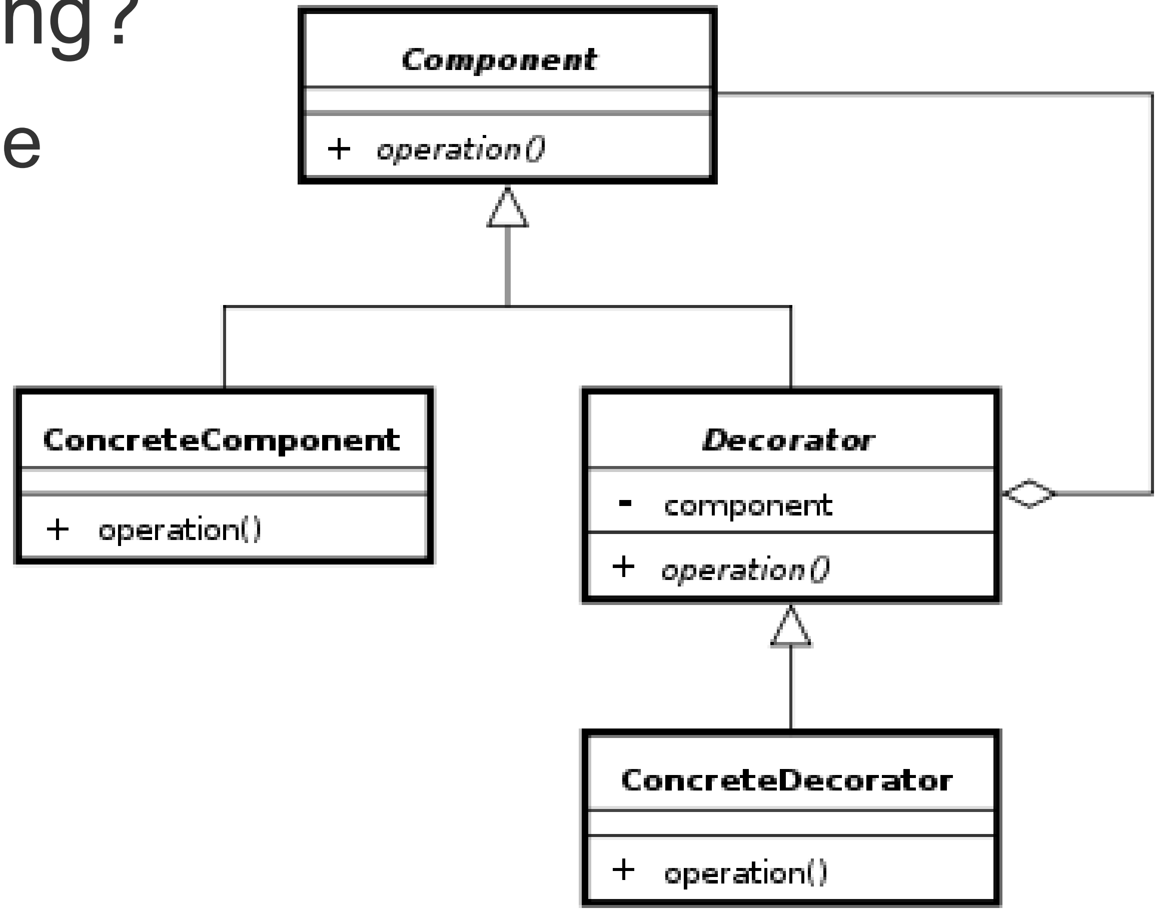
- how to decouple the object that produces data from the object that uses them?
  - NewsAgency vs NewsChannel



```
notifyObservers()
 for observer in observerCollection
  call observer.update()
```

Decorator

- how to extend functionalities without subclassing?
  - decorators can be added at runtime
  - there can be more than one decorator



Component
+ operation()
ConcreteComponent
+ operation()
Decorator
- component
+ operation()
ConcreteDecorator
+ operation()