Reuse at Design Level: Design Patterns III

Structural Patterns

- Adapter
  - Convert an interface to another
- Composite
  - Compose objects in a tree structure
- Facade
  - Provide a unified interface to a set of interfaces in a subsystem

Structural Patterns (Cont..)

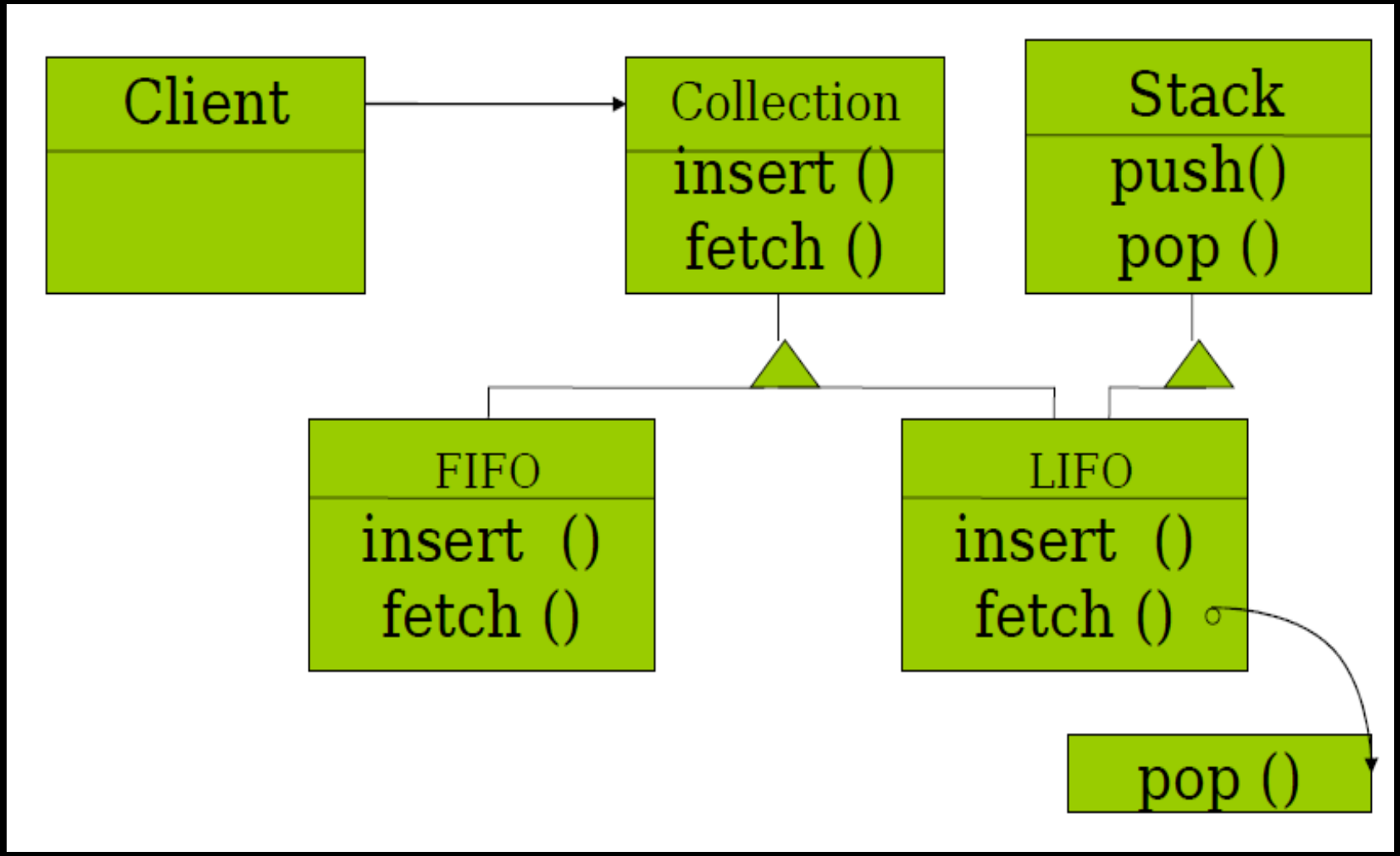
- Proxy
  - Provide a surrogate or placeholder for another object

- Bridge
  - Decouple abstraction from implementation, let them vary independently

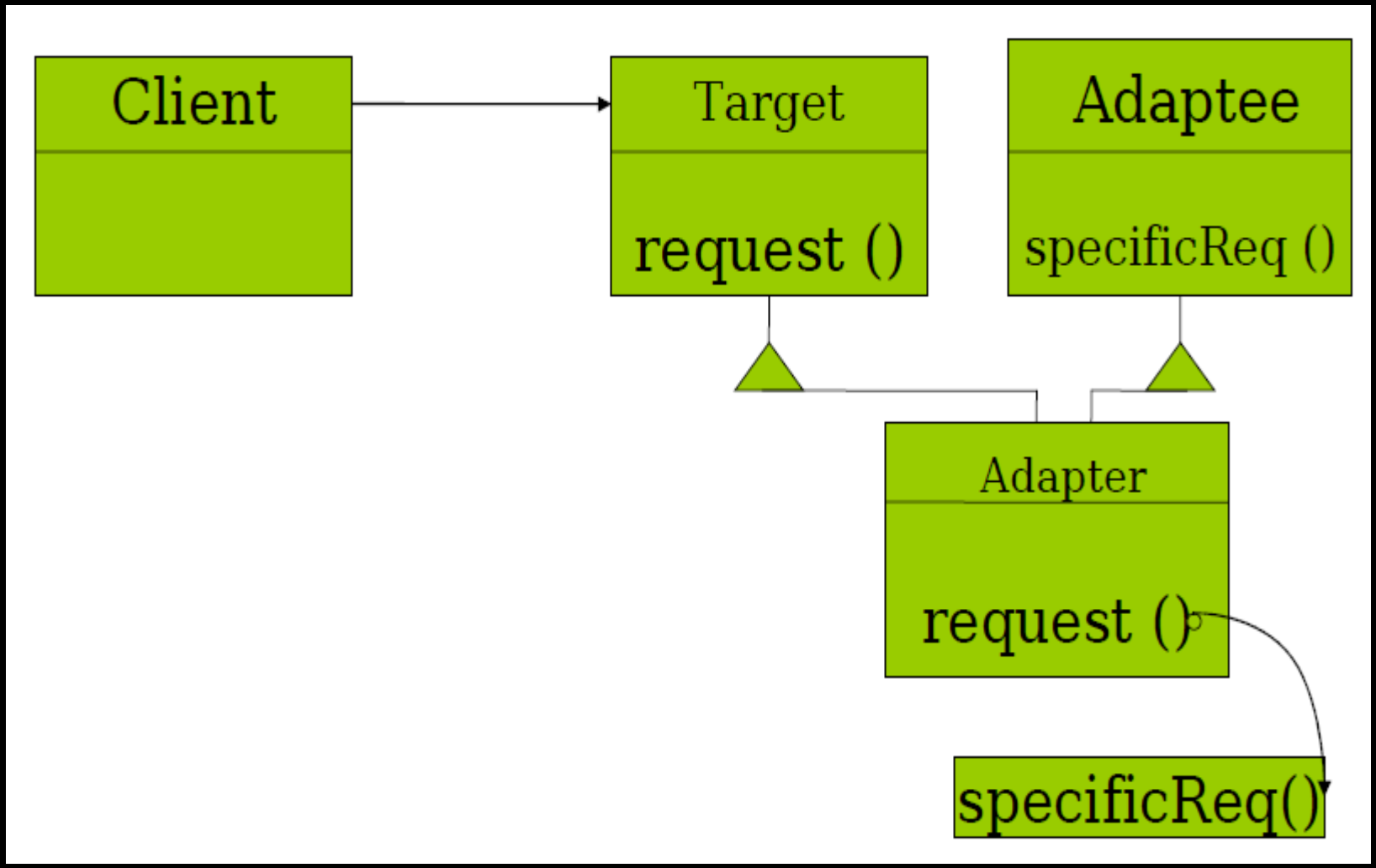
Adapter Pattern

- You are building a collection class hierarchy for collections such as FIFO, Set, LIFO
- You find that there is an existing class Stack which can be used for providing LIFO collection
- How do we adapt the existing class to the new interface of Collection classes?

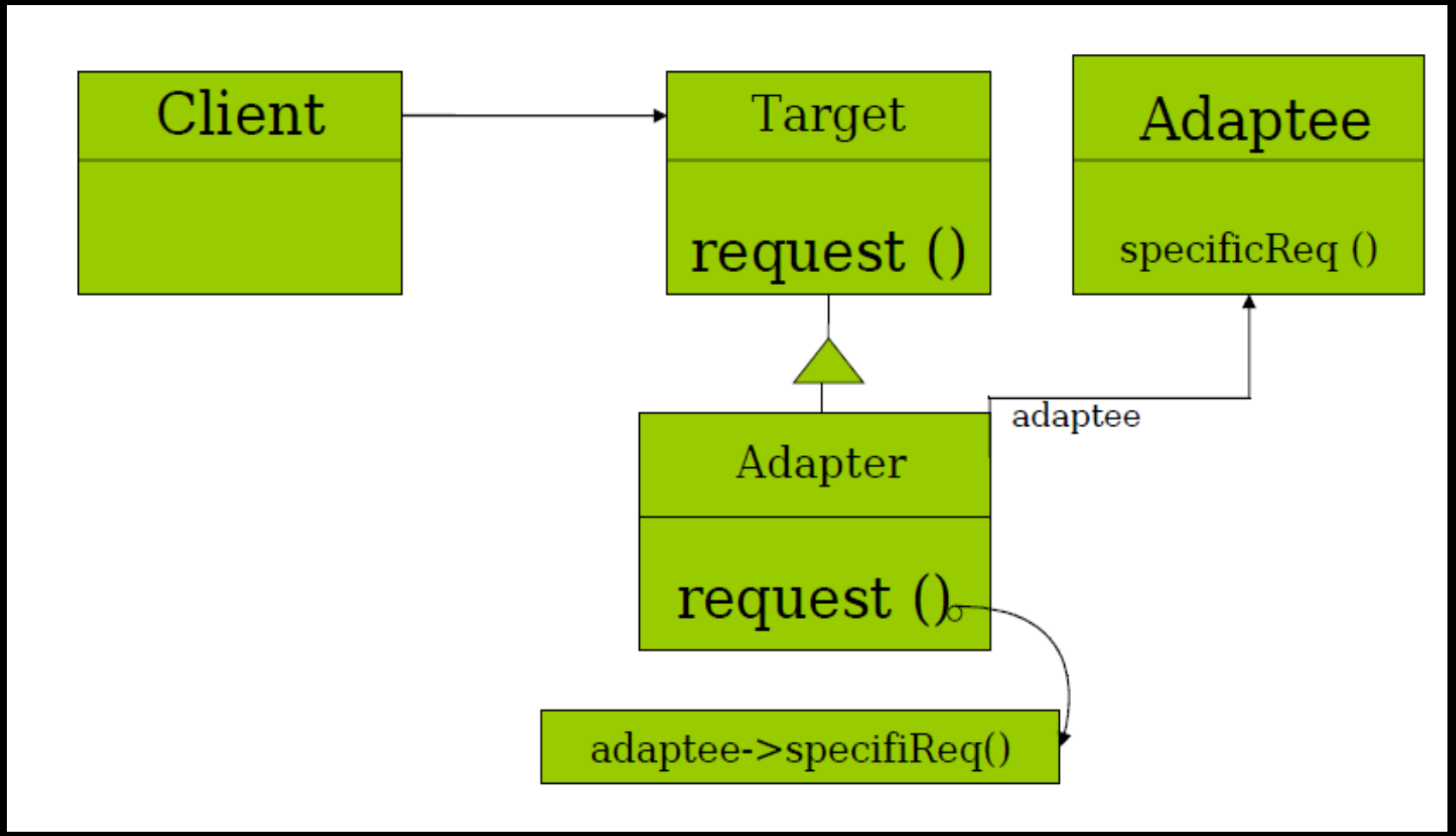
The Solution



The Adapter Pattern Class Adapter



The Adapter Pattern Object Adapter

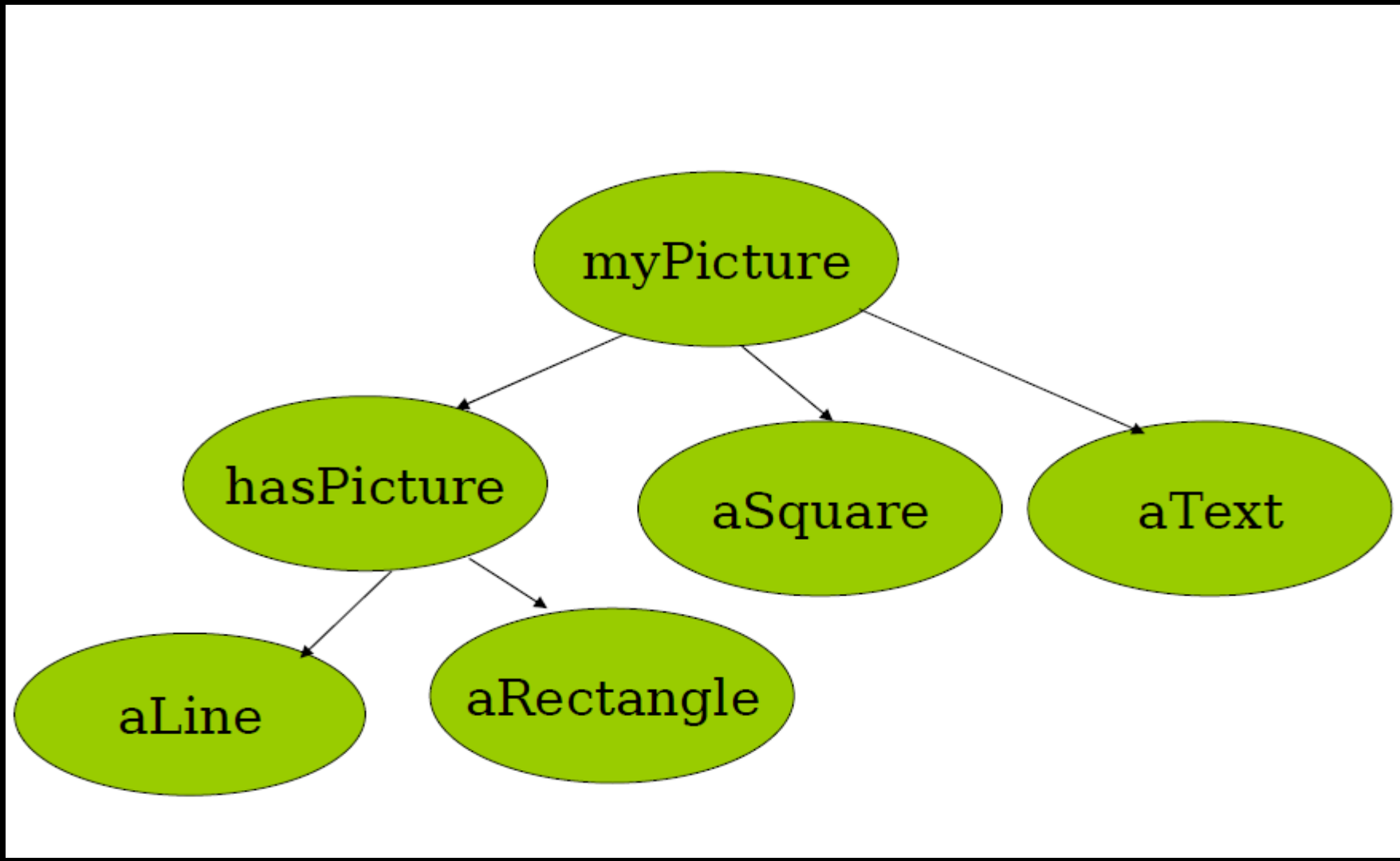


Composite Pattern

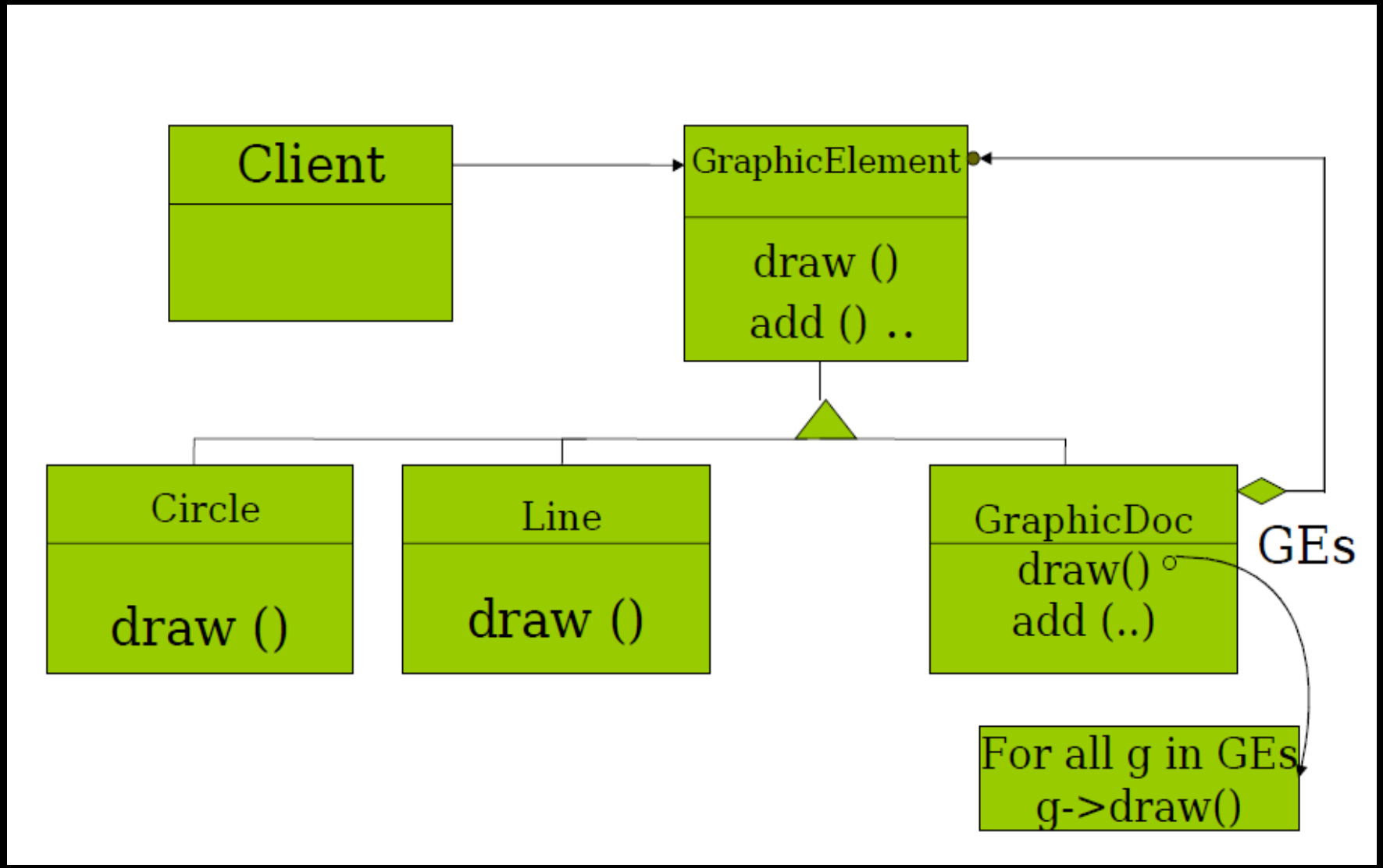
- An Example: A Graphic Document is composed of graphical objects such as Line, Rectangle, circle, Text, Image or another Graphical Document

- Thus a graphic document is a tree structured composition

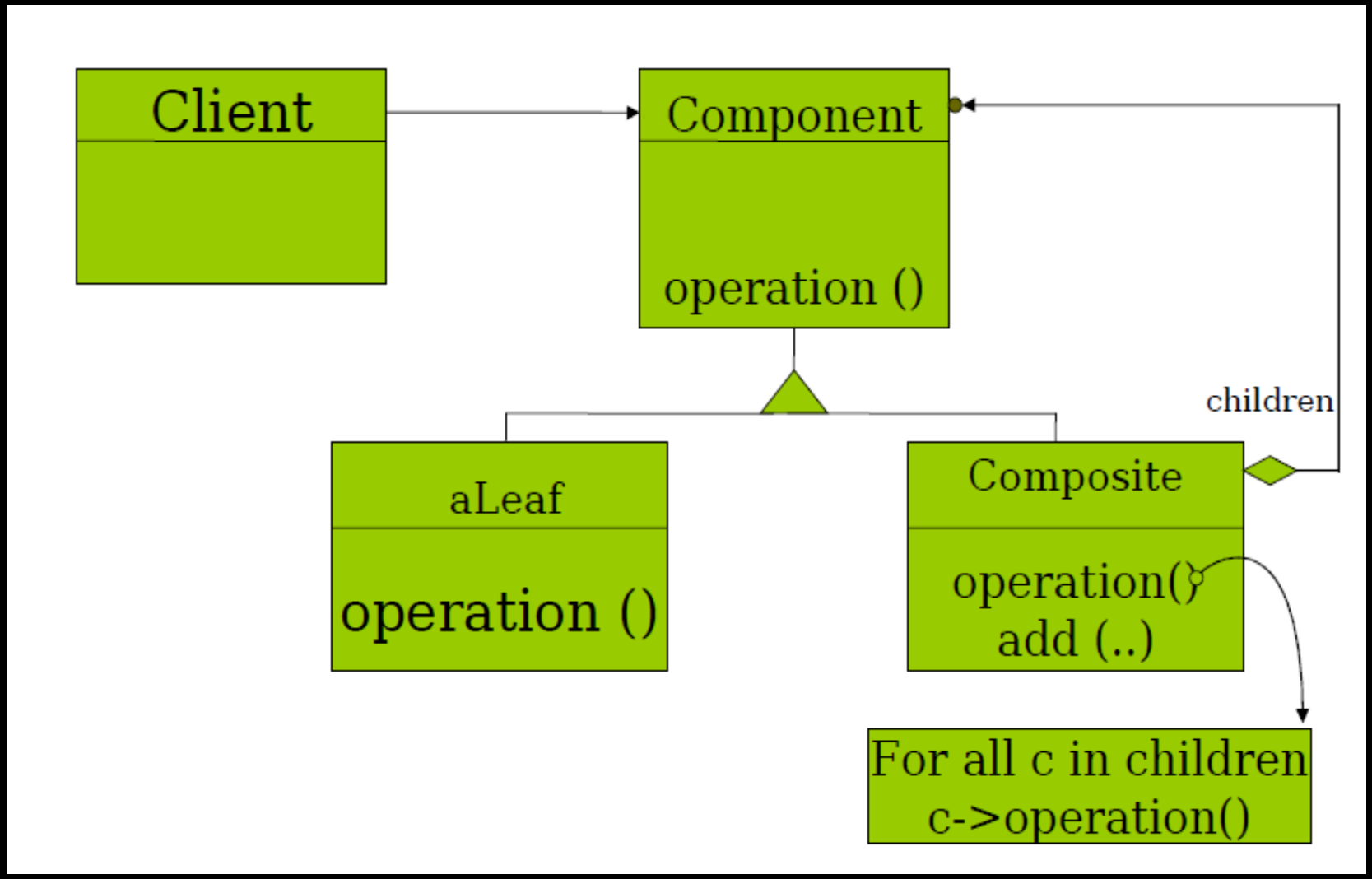
Instance of a Composite Class



The Solution

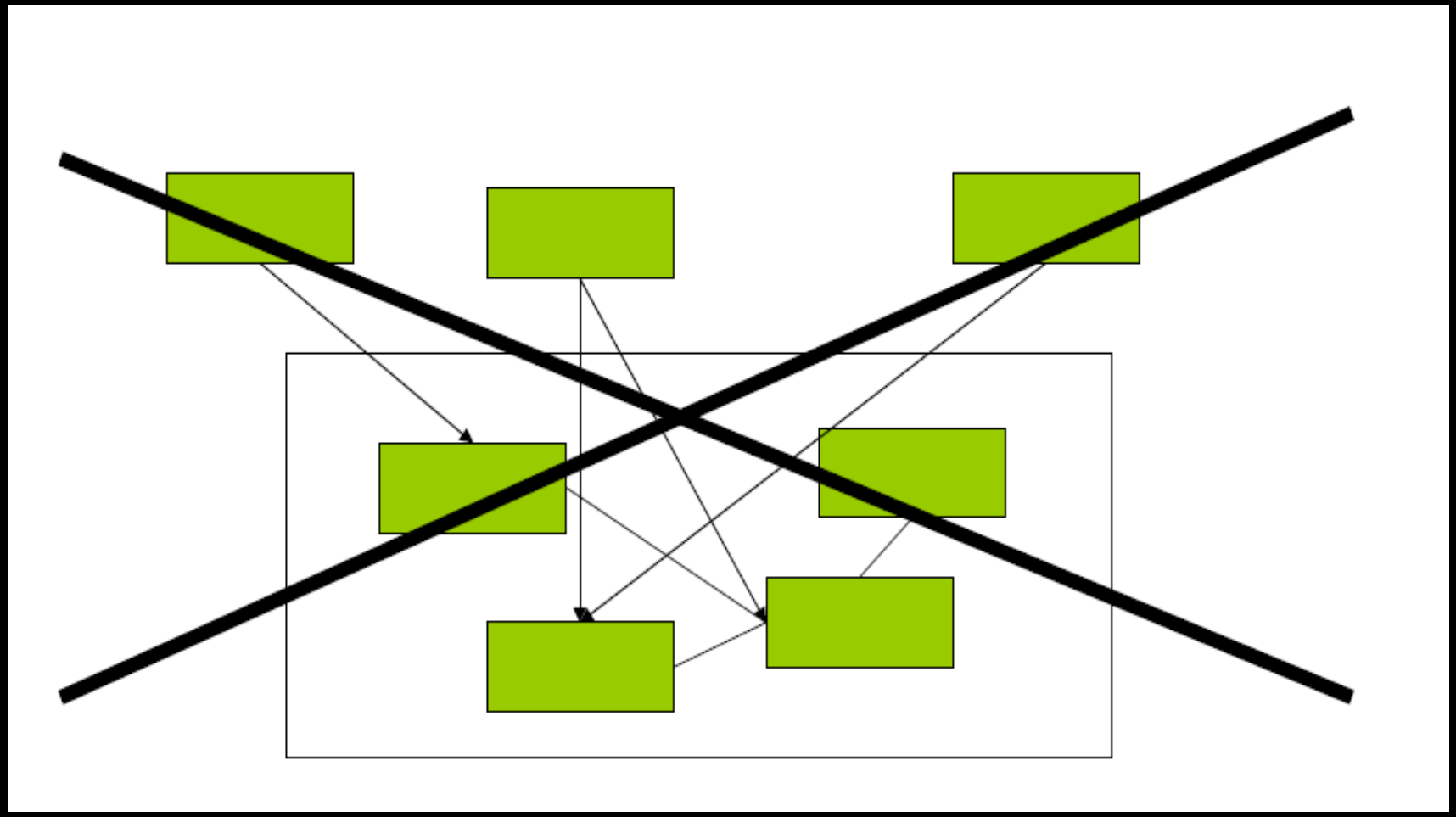


The Composite Pattern



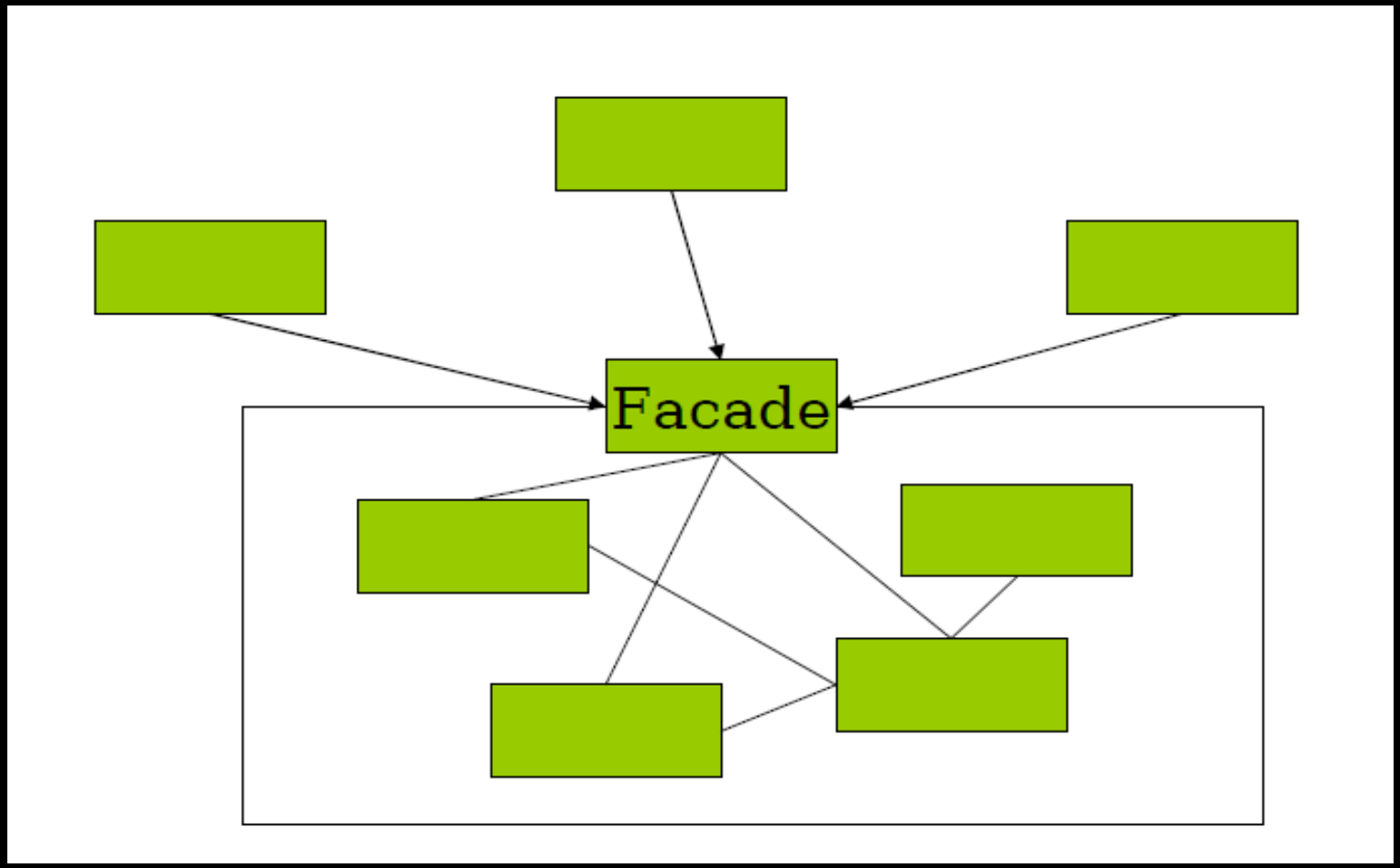
components within a subsystem?

- Study the following scenario



The Facade Pattern

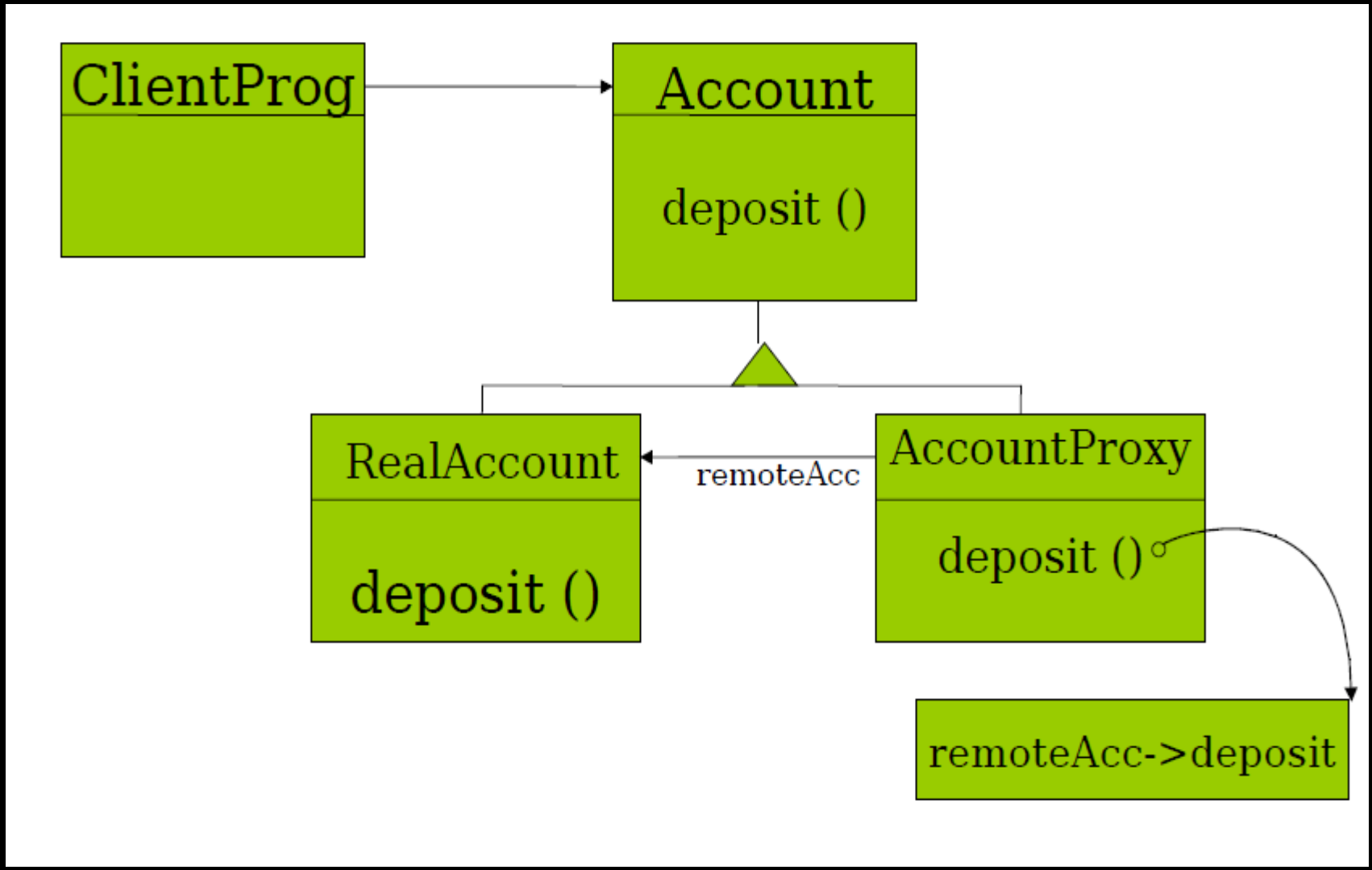
- Provide a unified interface for a subsystem



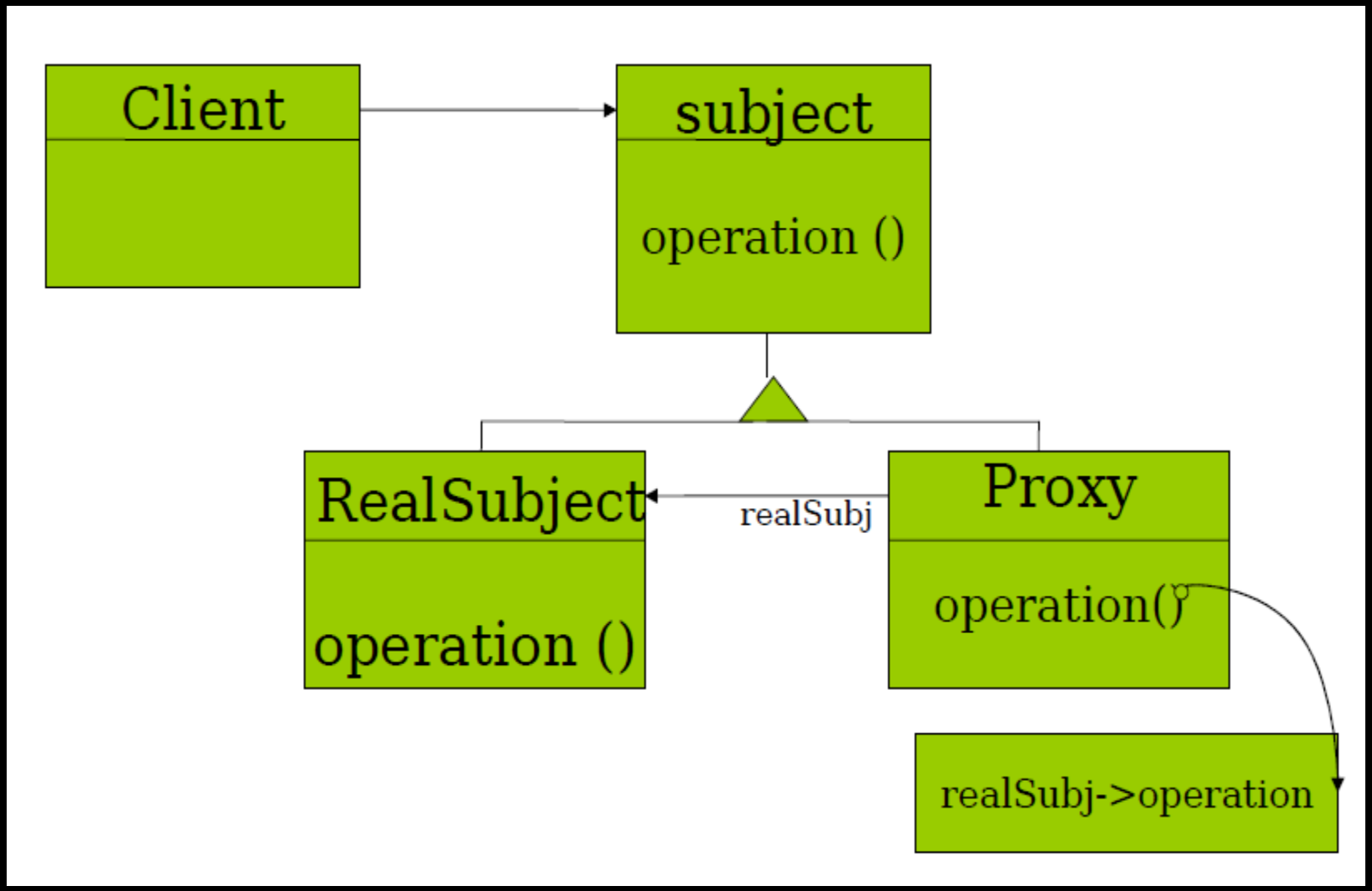
A Paradigm for Remoting

- Distribution transparency –
  - Client unaware of the distributed nature of the server
- Location Transparency
  - Client unaware of the location of the server
- A client invokes methods on an object as if it is a local object
- Proxy Handles provide a mechanism to implement this paradigm

Designing Surrogate Objects



The Proxy Pattern



The Proxy

- Both real and proxy objects inherit from an abstract superclass
- Thus, they both provide the same interface
- Their implementations are different
- A client can handle anyone of them through generalization, i.e. a superclass pointer
- Internally proxy carries out the communication with the remote object

The Pattern

- Client has a pointer to the Subject
- Subject is the abstract superclass
- RealSubject is the server implementation
- Proxy is the proxy implementation available at the client process
- Proxy has a handle to RealSubject
- Operation() is implementation differently by RealSubject and Proxy classes