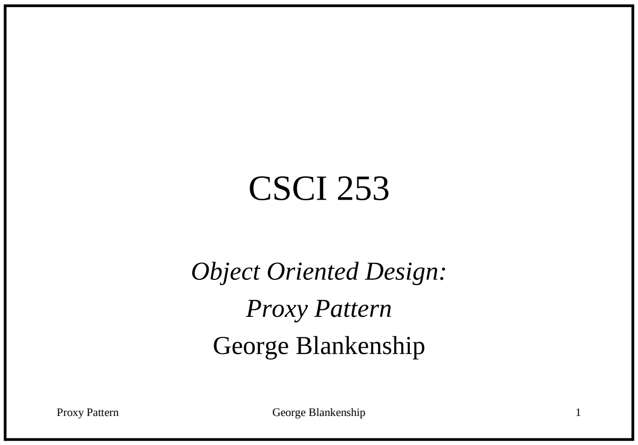
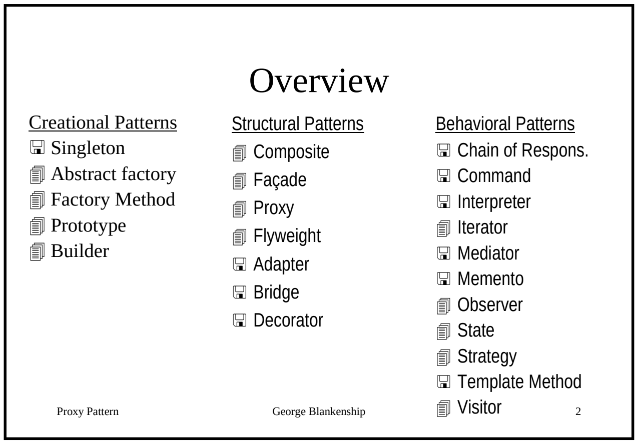


CSCI 253

Object Oriented Design:

Proxy Pattern

George Blankenship



Overview

Creational Patterns

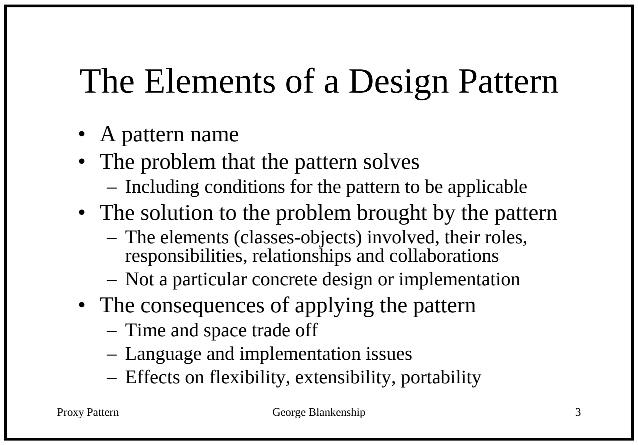
- 🖫 Singleton
- 🗐 Abstract factory
- 🗐 Factory Method
- 🗐 Prototype
- 🗐 Builder

Structural Patterns

- 🗐 Composite
- 🗐 Façade
- 🗐 Proxy
- 🗐 Flyweight
- 🖫 Adapter
- 🖫 Bridge
- 🖫 Decorator

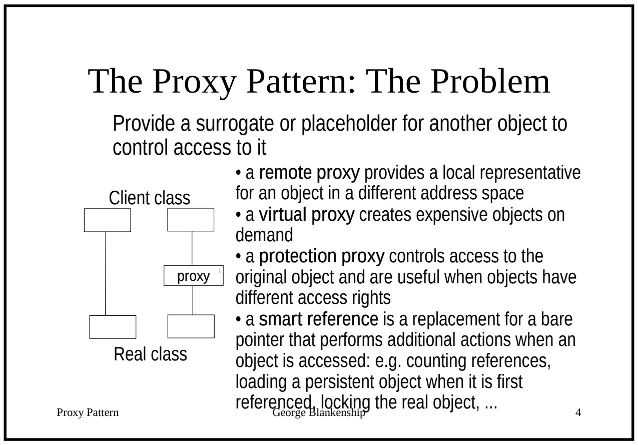
Behavioral Patterns

- 🖫 Chain of Respons.
- 🖫 Command
- 🖫 Interpreter
- 🗐 Iterator
- 🖫 Mediator
- 🖫 Memento
- 🗐 Observer
- 🗐 State
- 🗐 Strategy
- 🖫 Template Method
- 🗐 Visitor



The Elements of a Design Pattern

- A pattern name
- The problem that the pattern solves
  - Including conditions for the pattern to be applicable
- The solution to the problem brought by the pattern
  - The elements (classes-objects) involved, their roles, responsibilities, relationships and collaborations
  - Not a particular concrete design or implementation
- The consequences of applying the pattern
  - Time and space trade off
  - Language and implementation issues
  - Effects on flexibility, extensibility, portability

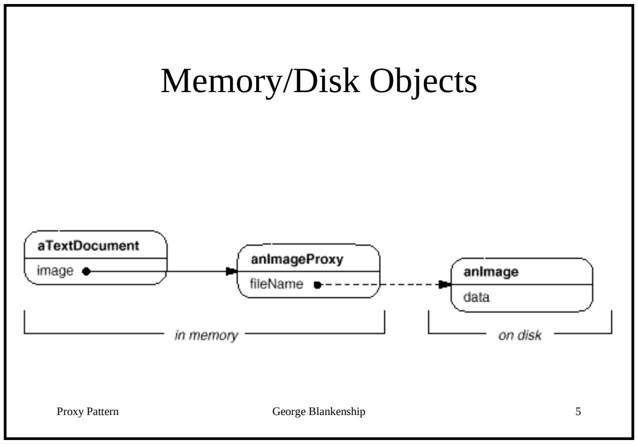


The Proxy Pattern: The Problem

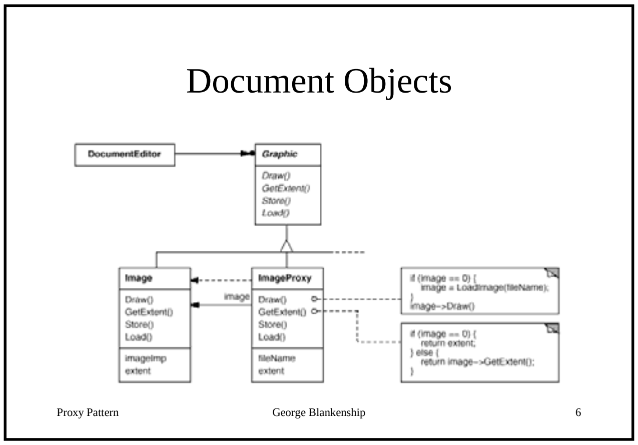
Provide a surrogate or placeholder for another object to control access to it

Client class
proxy
Real class

- a remote proxy provides a local representative for an object in a different address space
- a virtual proxy creates expensive objects on demand
- a protection proxy controls access to the original object and are useful when objects have different access rights
- a smart reference is a replacement for a bare pointer that performs additional actions when an object is accessed: e.g. counting references, loading a persistent object when it is first referenced, locking the real object, ...



Memory/Disk Objects

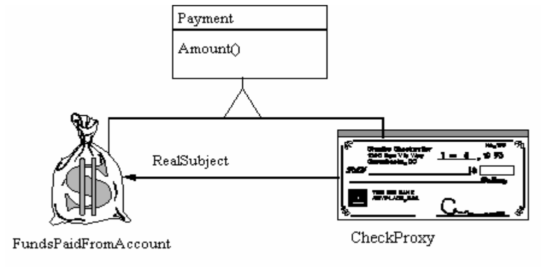


Document Objects

```
if (image == 0) {
    image = LoadImage(fileName);
}
image->Draw()
```

```
if (image == 0) {
    return extent;
} else {
    return image->GetExtent();
}
```

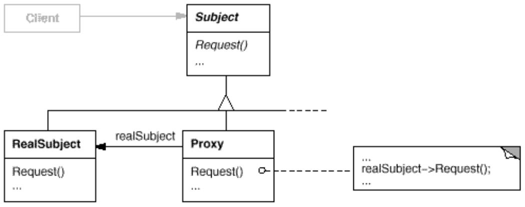
Financial Proxy



The Proxy Pattern Participants

- Maintains a reference that lets the proxy access the real subject
- Provides an interface identical to the Subject’s so that a proxy can be substituted for the real subject
- Controls access to the real subject and may be responsible for creating and deleting it
- Remote proxies are responsible for encoding a request and its arguments and for sending the request to the real subject in the other address space
- Virtual proxies may cache information about the real subject so that they can postpone accessing it
- Protection proxies check that the caller has the access permission to perform a request

Participant Map

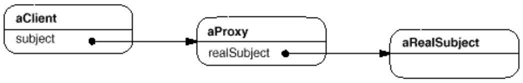


The Proxy Pattern Participants and Collaboration

- Subject:
  - Defines a common interface for Realsubject and Proxy so that a Proxy can be used anywhere a Realsubject is expected
- Realsubject
  - Defines the real object that the proxy represents
- Proxy forwards the request to Realsubject when appropriate (depends on the type of proxy)

Proxy Pattern George Blankenship 10

Proxy and Real



Proxy Pattern George Blankenship 11

The Proxy Pattern Consequences

- + The Proxy pattern introduces a level of indirection when accessing an object. This indirection has many uses:
  - A remote proxy can hide the fact that the object resides in a different address space
  - A virtual proxy can perform optimisations
  - Both protection proxies and smart pointers allow additional housekeeping
- + The proxy patterns can be used to implement “copy-on-write”: to avoid unnecessary copying of large objects the real subject is referenced counted; each copy requests increments this counter but only when a clients requests an operation that modifies the subject the proxy actually copies it

Proxy Pattern George Blankenship 12