CSCI 253

Object Oriented Design:
Mediator 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 Mediator Pattern: The Problem

Define an object that encapsulates how a set of objects interact. Mediator promotes loose coupling by keeping objects from referring to each other explicitly, and it lets you vary their interaction independently.

-The problem that commonly occurs in dialog boxes
-If each object in the dialog takes responsibility for the dependencies it is associated with, the result is a highly coupled set of objects with low cohesion.

Mediator Pattern George Blankenship 4

Mediator Relationship

- Prior to the use of a mediator there is a complex relationship between all of the classes
- The mediator disconnects the classes and presents a single target interface or set of targets

Mediator Pattern George Blankenship 5

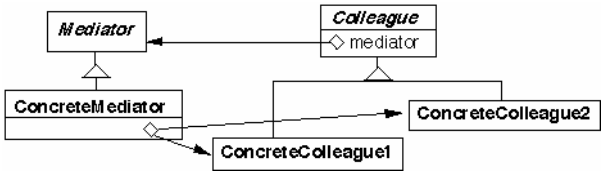
Mediator and Façade

- Mediator is similar to Façade in that it abstracts functionality of existing classes.
- Mediator abstracts/centralizes arbitrary communication between colleague objects, it routinely "adds value", and it is known/referenced by the colleague objects
- Mediator defines a multidirectional protocol).
- Façade defines a simpler interface to a subsystem, it doesn't add new functionality, and it is not known by the subsystem classes
- Façade defines a unidirectional protocol where it makes requests of the subsystem classes but not vice versa

Mediator Pattern George Blankenship 6

The Mediator Pattern: Structure

- A mediator is responsible for controlling and coordinating the interactions of a group of objects (not data structures)



Mediator Pattern George Blankenship 7

The Mediator Pattern: Participants

- **Mediator**
  - Defines an interface for communicating with Colleague objects
- **ConcreteMediator**
  - Implements cooperative behavior by coordinating Colleague objects
  - Knows and maintains its colleagues
- **Colleague classes**
  - Each Colleague class knows its Mediator object
  - Each colleague communicates with its mediator whenever it would have otherwise communicated with another colleagu

Mediator Pattern George Blankenship 8

The Mediator Pattern: Collaboration

- It limits subclassing
  - The communication behavior would otherwise have to be distributed among many sub-classes of the widgets
  - Instead, it's all in the Mediator
- It decouples colleagues
  - They don't have to know how to interact with each other
- It simplifies object protocols
  - A Mediator replaces many-to-many communication with a one-to-many paradigm
- It abstracts how objects cooperate
  - How objects communicate is abstracted into the Mediator class
- It centralizes control
  - As it's all in the Mediator
  - This can make the Mediator quite large and monolithic in a large system

Mediator Pattern George Blankenship 9

The Mediator Pattern: Consequences

- Most of the complexity involved in managing dependencies is shifted from other objects to the Mediator object. This makes the other objects easier to implement and maintain.
- Colleague classes are more reusable because their core functionality is not interlinked with dependency-handling code.
- Because dependency-handling code is usually application specific, Mediator classes are not usually reusable.

Mediator Pattern George Blankenship 10

The Mediator Pattern: Implementation

- Partition a system into pieces or small objects.
- Centralize control to manipulate participating objects(a.k.a colleagues)
- Clarify the complex relationship by providing a board committee.
- Limit subclasses.
- Improve objects reusabilities.
- Simplify object protocols.
- The relationship between the control class and other participating classes is multidirectional.
- Related patterns include
  - Facade, which abstracts a subsystem to provide a more convenient interface, and its protocol is unidirectional, whereas a mediator enables cooperative behavior and its protocol is multidirectional.
  - Command, which is used to coordinate functionality.
  - Observer, which is used in mediator pattern to enhance communication.

Mediator Pattern George Blankenship 11