Mediator Pattern

Problem

- Sometimes you have a set objects and **every object wants to know about every other**.

  You have **n-to-n relation of notifications**.

- A set of object generate events and all others need to be informed.

- It can result in many object connections and you want to have **loose coupling** by keeping objects from referring to each.

Mediator Pattern

- “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.” Gamma et al.

Use Mediator Pattern

When we want to

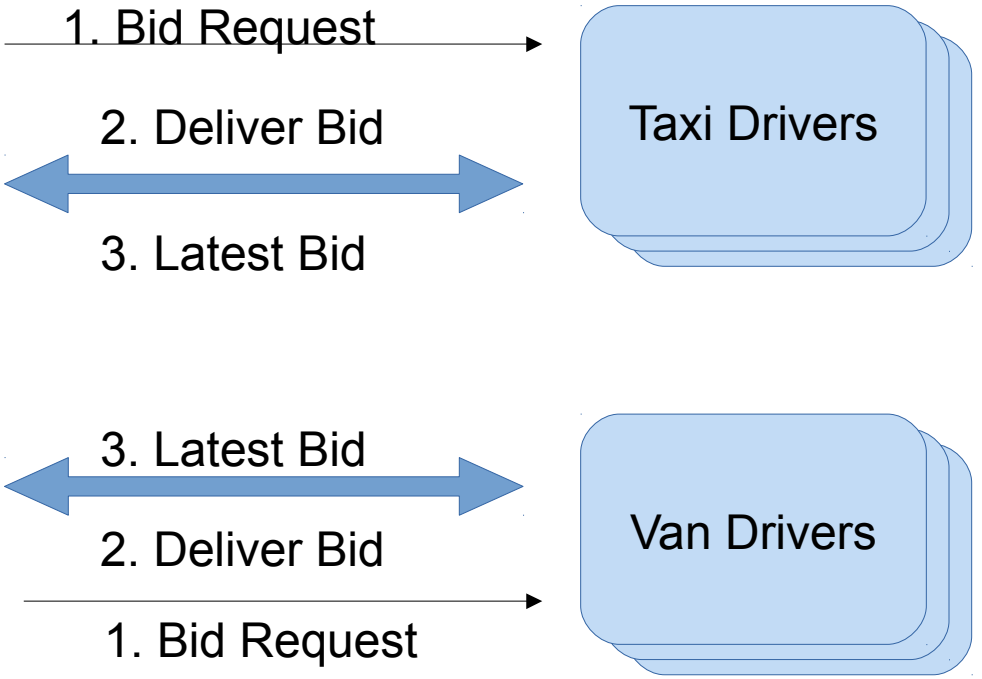
- have a set of objects communicate in well-defined but complex ways. This can cause unstructured and difficult to manage dependencies.

- reuse an object but it is challenging because it refers to **many other communicating objects**.

- a distributed behavior should be customized **without a lot of subclassing**.

Flower Delivery Example

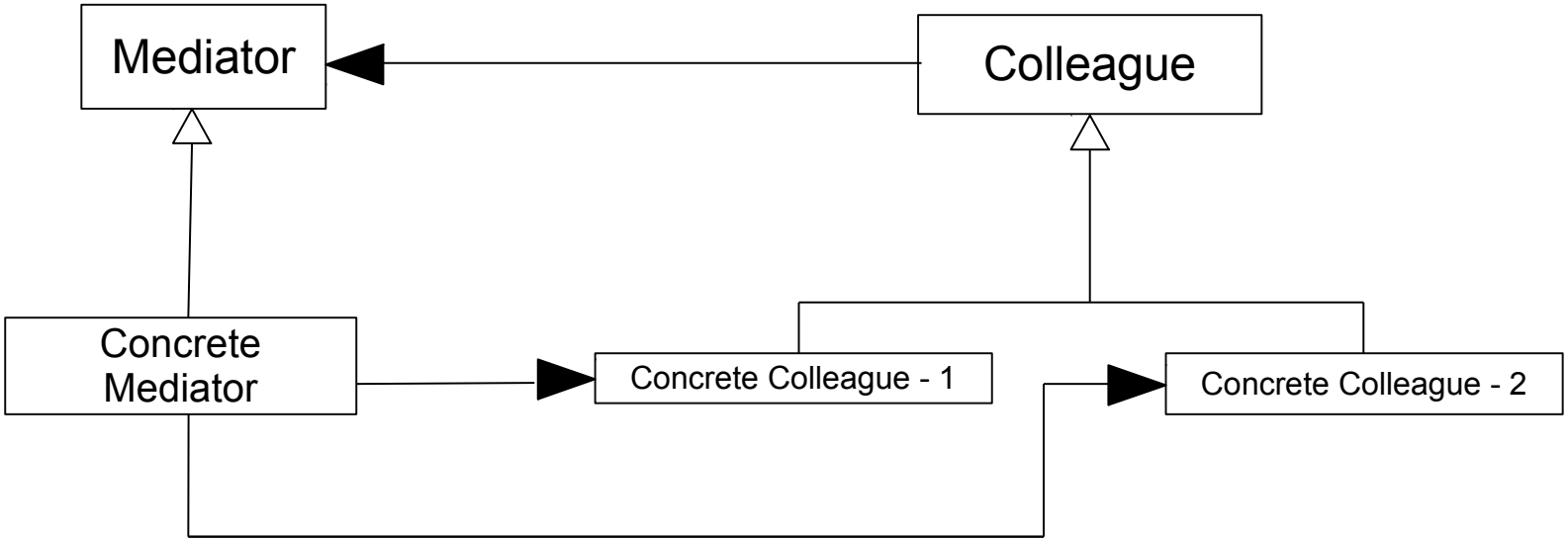
Flower Shop



Automatic Delivery Bidding

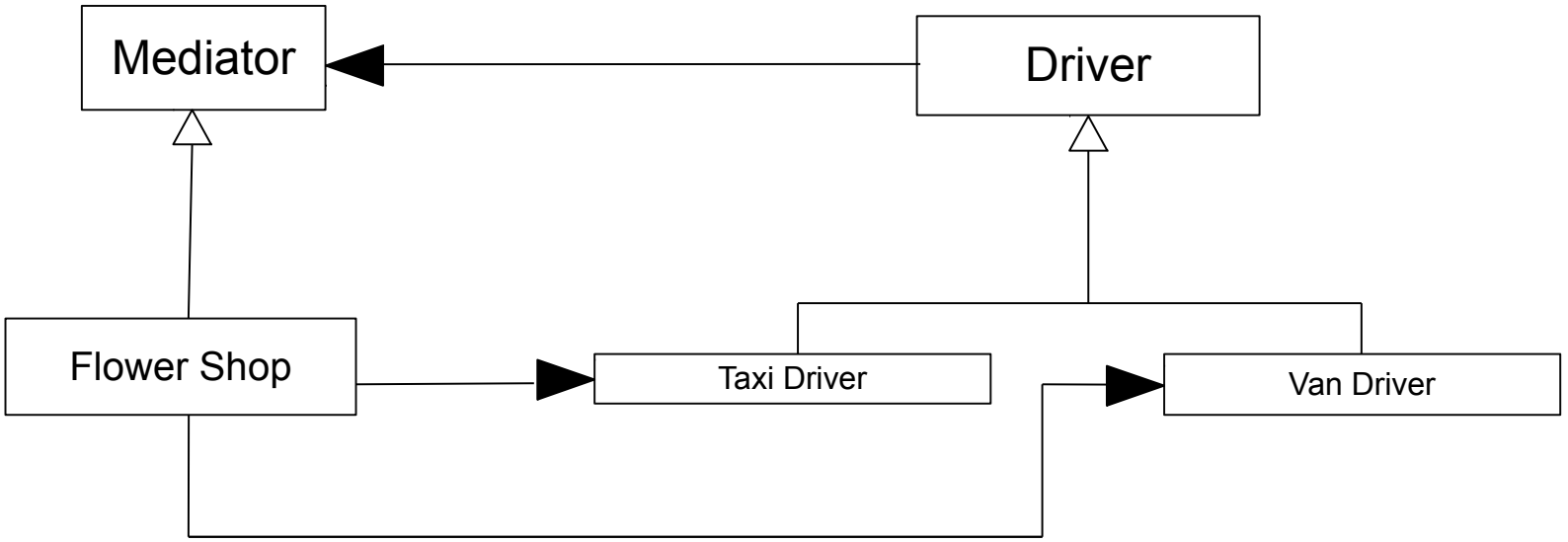
- Multiple Drivers are bidding on a flower delivery
- Drivers send their bids and need to be informed about the latest bid to send their next bids.

Structure of Mediator pattern



- **Mediator**
  - defines an interface for communicating with Colleague objects.
- **Concrete Mediator**
  - **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 colleague.

Flower Delivery Example



Consequences

+ It decouples colleagues. A mediator promotes loose coupling between colleagues. You can vary and reuse Colleague and Mediator classes independently.

+ It simplifies object protocols.

Replaces many-to-many interactions with one-to-many interactions

+ It abstracts how objects cooperate.

We focus more on how objects interact apart from their individual behavior.

It helps to clarify how objects interact in a system.

+ It limits subclassing. A mediator includes behavior that otherwise are distributed among several objects. Alternative to mediator is subclassing Mediator.

- It centralizes control. Because a mediator encapsulates protocols, it can become more complex than any individual colleague. The mediator is a **monolith** that's hard to maintain.

(Centralized system vs. Distributed System)

(Orchestration vs Choreography)

Related Patterns

- **Facade** abstracts a subsystem of objects to provide a simple interface. **Facade** objects make requests of the subsystem classes but not vice versa.
  - **Mediator** is **multidirectional**. Facade is **unidirectional**.

- **Colleagues can communicate with the mediator using the Observer pattern.**

Summary

- "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." Gamma et al.

- Mediator Pattern is considered to be a **behavioral pattern.**