InterBit
Training & Consulting Ltd.

Mediator Pattern

Chapter Content

InterBit
Training & Consulting Ltd.

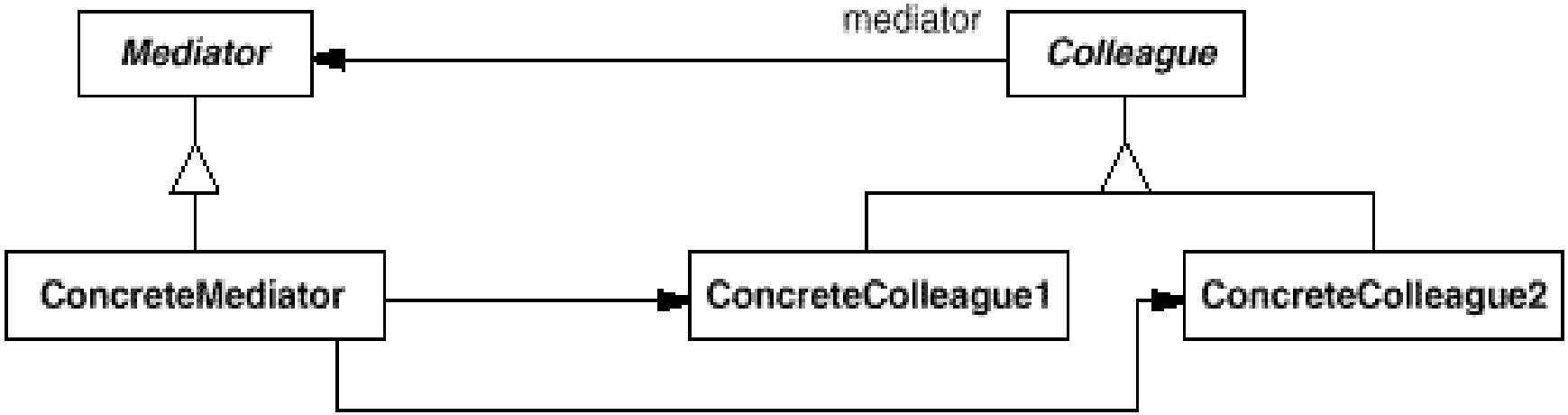
- Mediator Pattern Overview
- Mediator Pattern UML Diagram
- Mediator Examples
- Mediator Conclusion

Mediator Pattern Overview

InterBit Training & Consulting Ltd.

- A program may involve a complex graph of communication (message-passing) between many classes/objects.
- Mediator centralizes this communication
  - Objects notify the mediator of any changes. The mediator holds the logic for deciding how such changes should affect other objects.
  - Hence, loose-coupling (classes no longer need to know of each other; only of the mediator).

Mediator Pattern UML Diagram



Mediator - defines an interface for communicating with Colleague objects.

ConcreteMediator - knows the colleague classes and keep a reference to the colleague objects. implements the communication and transfer the messages between the colleague classes

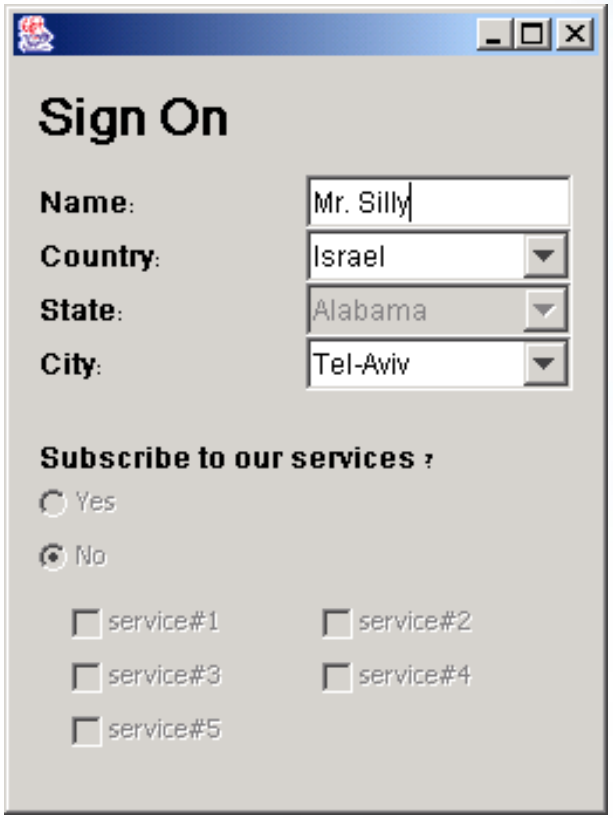
Colleague classes - keep a reference to its Mediator objectcommunicates with the Mediator whenever it would have otherwise communicated with another Colleague.

Mediator Example #1

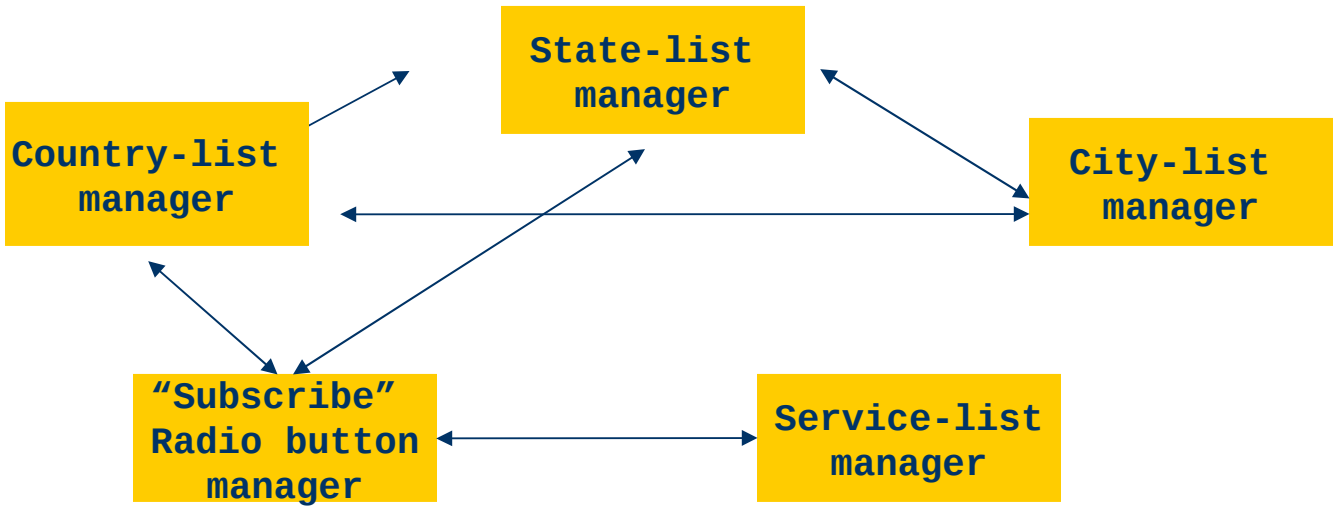
InterBit Training & Consulting Ltd.

- Consider the following form:

- “State” list depends on country (& may be disabled)
- “City” list changes depending on country & state
- Subscription is only offered for USA & CANDA, but excluding Alaska
- “Services” are disabled if user cannot (or does not wish to) subscribe

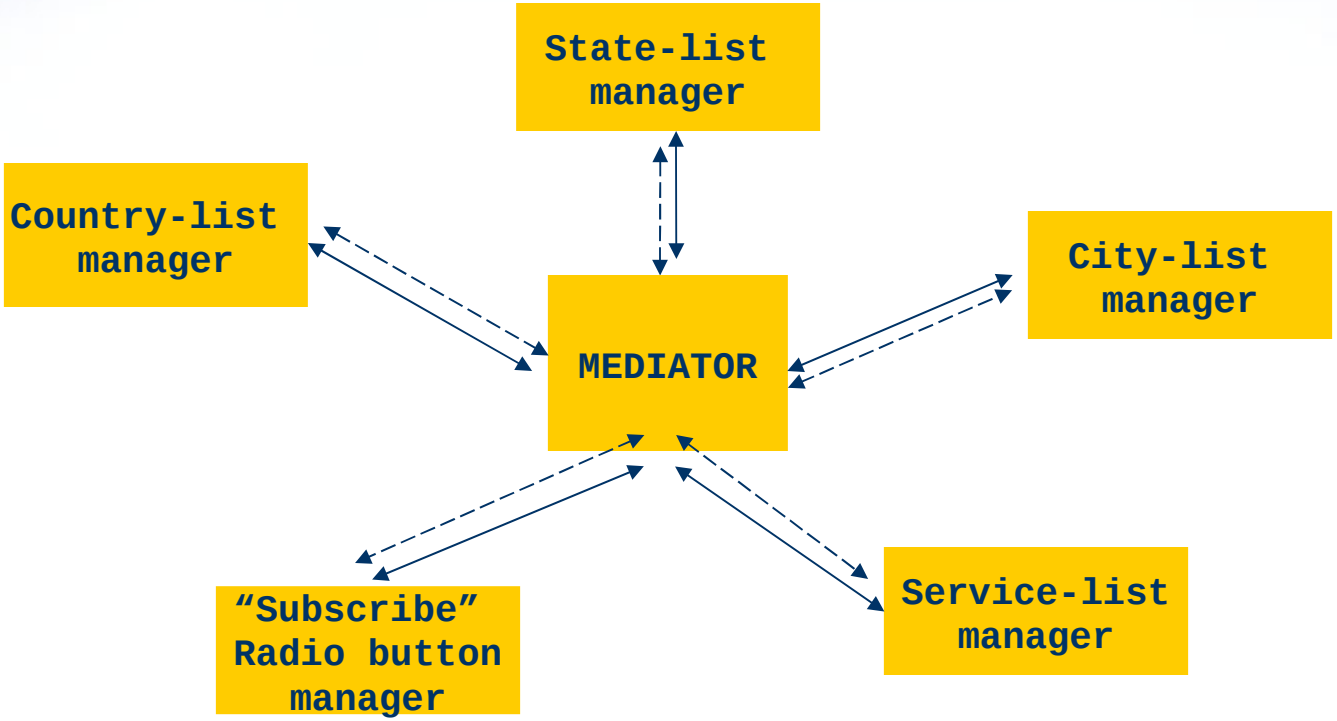


Without a mediator:



Arrow= listens-to ⟷

With a mediator:



Arrow= listens-to

Dotted=updates

Example #2

InterBit Training & Consulting Ltd.

- Consider the following sub-systems of a company:
  - Personnel (hire, fire).
  - Accounting (salaries, company's assets…).
  - Transportation (company's bus/taxi fleet).
  - Projects (deadlines, assigning personnel to projects …).
  - Courses (send employees to courses).

Example #2 (cont.)

InterBit
Training & Consulting Ltd.

- Consider the pros & cons of centralizing the following communication:
  - When employee is hired/fired: update all other sub-systems.
  - Stressed project deadlines: order late-night transportation; cancel non-urgent Courses.
  - Courses in remote locations: make transportation arrangements ...

Problems and implementation

InterBit Training & Consulting Ltd.

Abstract Mediators

- There is no need to create an Abstract Mediator class or an interface as long as the colleagues are going to use only one mediator.

- The definition of an abstract Mediator is required only if the colleagues needs to work with different mediators.

Problems and implementation

InterBit Training & Consulting Ltd.

Communication of mediators and colleagues

There are different ways to realize the communication between the colleagues and its mediator:One of the most used methods is to use the Observer pattern.

- The mediator can be also an observer and the Colleagues can be implement an observable object.
- Each time an change is made in the state of the observable object, the observer(mediator) gets notified and it notify all other colleagues objects.

Problems and implementation

InterBit Training & Consulting Ltd.

Communication of mediators and colleagues

Alternative methods can be used to send messages to the mediator.

- For example a simple delegation can be used and specialized methods can be exposed by the mediator
- In more complex implementations asynchronous messages can be added to to a message queue, from where they can be picked up by the mediator object

Problems and implementation

InterBit Training & Consulting Ltd.

Complexity of Mediator object

- The mediator object handles all the interaction between the participants objects
- One potential problem is complexity of the mediator when the number of participants is a high and the different participant classes is high
- If you created custom dialogs for GUI applications you remember that after some time the dialogs classes become really complex because they had to manage a lot of operations

Mediator - conclusion

InterBit Training & Consulting Ltd.

Advantages:

- Comprehension - The mediator encapsulate the logic of mediation between the colleagues. From this reason it' more easier to understand this logic since it is kept in only one class.
- Decoupled Colleagues - The colleague classes are totally decoupled. Adding a new colleague class is very easy due to this decoupling level.

Mediator - conclusion

InterBit Training & Consulting Ltd.

Advantages (continued)

- Simplified object protocols - The colleague objects need to communicate only with the mediator objects. Practically the mediator pattern reduce required communication channels (protocols) from many to many to one to many and many to one.

- Limits Subclassing - Because the entire communication logic is encapsulated by the mediator class, when this logic need to be extended only the mediator class need to be extended.

Mediator - conclusion

InterBit Training & Consulting Ltd.

Disadvantages

- Complexity - in practice the mediators tends to become more complex and complex.
- A good practice is to take care to make the mediator classes responsible only for the communication part.
- For example when implementing different screens - the the screen class should not contain code which is not a part of the screen operations (this should be put in some other classes)