MEDIATOR PATTERN

GUVENC USANMAZ

Dictionary Definition

Dictionary Definition

Mediator: A person or organization that tries to end a quarrel between two people, groups, countries etc by discussion. (Longman Dictionary)

"First, we can agree that it is a big back yard?"

Gang of Four Definition

You can use the Mediator pattern to, "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."

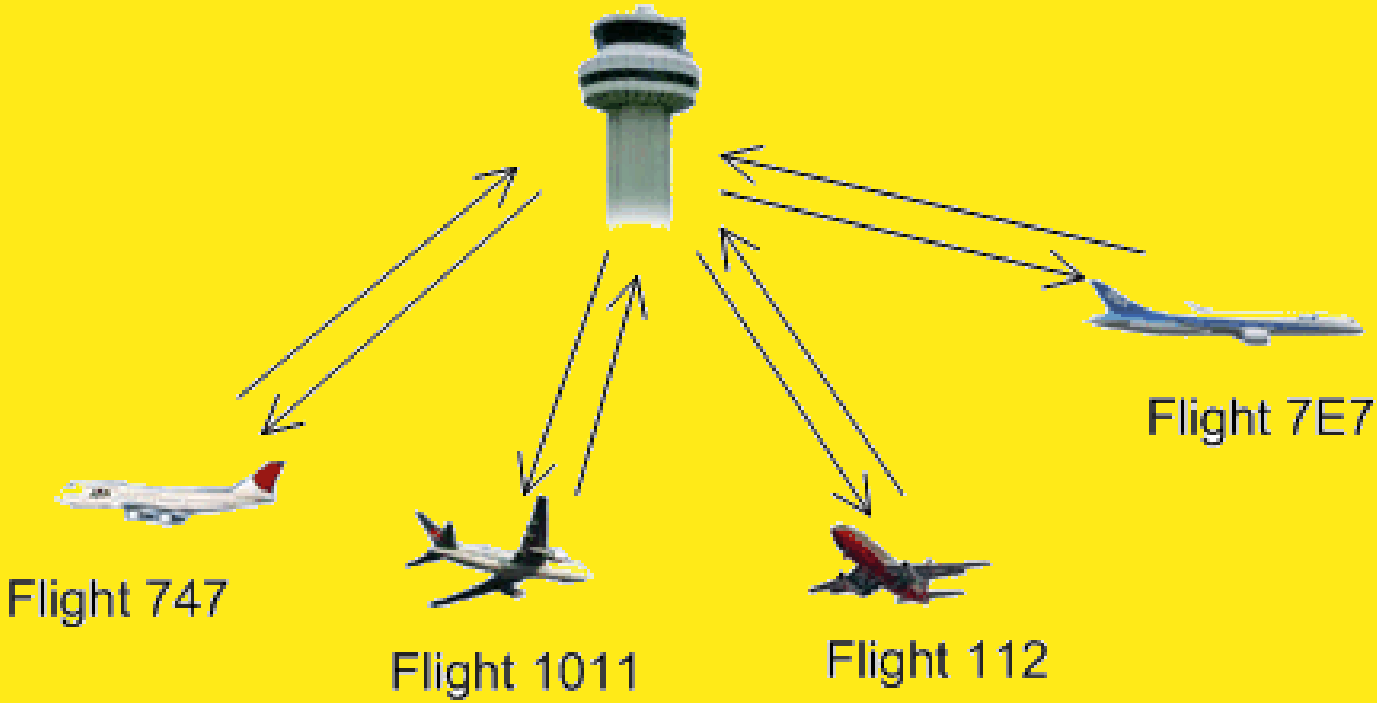
Motivation

In order to avoid tight coupled frameworks, we need a mechanism to facilitate the interaction between objects in a manner in that objects are not aware of the existence of other objects.

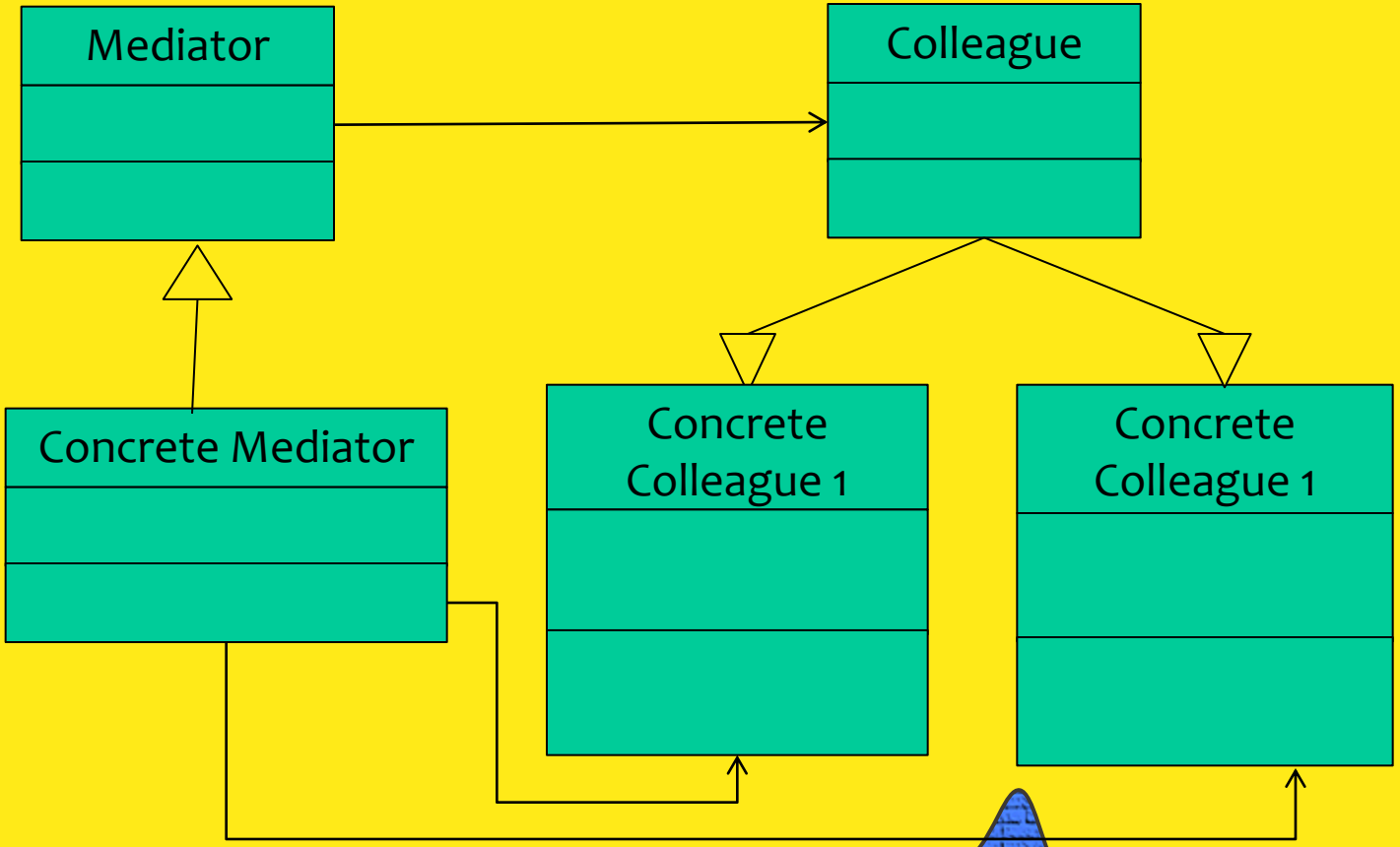
Intent

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.

A Real Life Mediator



Structure


Mediator
Colleague
Concrete Mediator
Concrete Colleague 1
Concrete Colleague 1

Participants

- **Mediator** - defines an interface for communicating with Colleague objects.
- **Concrete Mediator** - 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 object - communicates with the Mediator whenever it would have otherwise communicated with another Colleague.

Type

What type of pattern, mediator is ?

Type

Mediator pattern is a behavioral pattern.

Applicability

According to Gamma et al, the Mediator pattern should be used when:

- A set of objects communicate in well-defined but complex ways. The resulting interdependencies are unstructured and difficult to understand.
- Reusing an object is difficult because it refers to and communicates with many other objects.
- A behavior that's distributed between several classes should be customizable without a lot of subclassing.

Specific Problems and Implementation

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.

Specific Problems and Implementation

Communication between mediators and colleagues

There are different ways of 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.
- 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 a message queue, from where they can be picked up by the mediator object

Specific Problems and Implementation

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.

Sample Code

ADVANTAGES

- Increases the reusability of the objects supported by the Mediator by decoupling them from the system.
- Simplifies maintenance of the system by centralizing control logic.
- Simplifies and reduces the variety of messages sent between objects in the system.
- Partition a system into pieces or small objects. Centralize control to manipulate participating objects.
- Most of the complexity involved in managing dependencies is shifted from other objects to the Mediator object. This makes other objects easier to implement and maintain.

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.

Related Patterns

There are a few design patterns that are closely related to the Mediator pattern and are often used in conjunction with it.

Related Patterns

Facade Pattern

A simplified mediator becomes a facade pattern if the mediator is the only active class and the colleagues are passive classes. A facade pattern is just an implementation of the mediator pattern where mediator is the only object triggering and invoking actions on passive colleague classes. The Facade is being call by some external classes

Related Patterns

Adapter Pattern

The mediator pattern just "mediate" the requests between the colleague classes. It is not supposed to change the messages it receives and sends; if it alters those messages then it is an Adapter pattern.

Related Patterns

Observer Pattern

The difference between them is that Observer distributes communication by introducing "observer" and "subject" objects, whereas a Mediator object encapsulates the communication between other objects. We've found it easier to make reusable Observers and Subjects than to make reusable Mediators.

Questions

Any questions ?


That's all Folks!

References

- http://pages.cpsc.ucalgary.ca/~heatond/mediator/
- Wikipedia
- http://www.oodesign.com/mediator-pattern.html
- http://sourcemaking.com/design_patterns/mediator
- http://www.avajava.com/tutorials/lessons/mediator-pattern.html?page=1