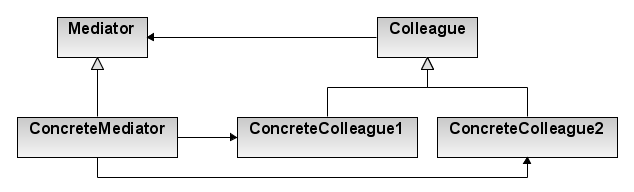
Refresh:

This pattern defines a separate (mediator) object that encapsulates the interaction between a set of objects and the objects delegate their interaction to a mediator object instead of interacting with each other directly.

*Architecture*



*Design participants*

* **Mediator** – defines the interface for communication between Colleague objects
* **ConcreteMediator** – implements the Mediator interface and coordinates communication between Colleague objects. **It is aware of all of the Colleagues and their purposes with regards to inter-communication**.
* **Colleague** – defines the interface for communication with other Colleagues through its Mediator
* **ConcreteColleague** – implements the Colleague interface and communicates with other Colleagues through its Mediator

Exercise:

In this **Java mediator pattern exercise**, we are simulating the chat application where users can send messages to other users in one-to-one fashion. All users must be registered to the chat application to send or receive messages.

There is missing an appropriate interface at MediatorExercise.zip to get our program to start working properly, code it and run the application to see the next output:

Alex :: Sending Message : Hello brian

Brian :: Received Message : Hello brian

Brian :: Sending Message : Hey buddy

Alex :: Received Message : Hey buddy

Hint:

Our concrete mediator in this exercise is ChatRoom.java, there is missing our mediator interface, create one accordingly with the name of our concrete mediator.

Check the complete solution at:

[Mediator Design Pattern - Mediator Pattern in Java - HowToDoInJava](https://howtodoinjava.com/design-patterns/behavioral/mediator-pattern/)