Mediator

Objective

Define an object that encapsulates the interaction between others, promoting a low coupling; since it prevents objects from explicitly referencing each other by varying independently of their interaction.

Function

Define an object that coordinates communication between others of different kinds, but that functions as a whole.

Structure

As shown in figure 1

- Client: Component that initiates communication with the rest of the components through the mediator.

- Components: Components that are part of the communication network through the mediator, these can be various objects that share the same mediator to communicate.

- Mediator: Component that serves as a mediator between the rest of the components, its main role is to channel the incoming messages to the corresponding recipient.

The structure that meets this pattern is shown in Figure 1

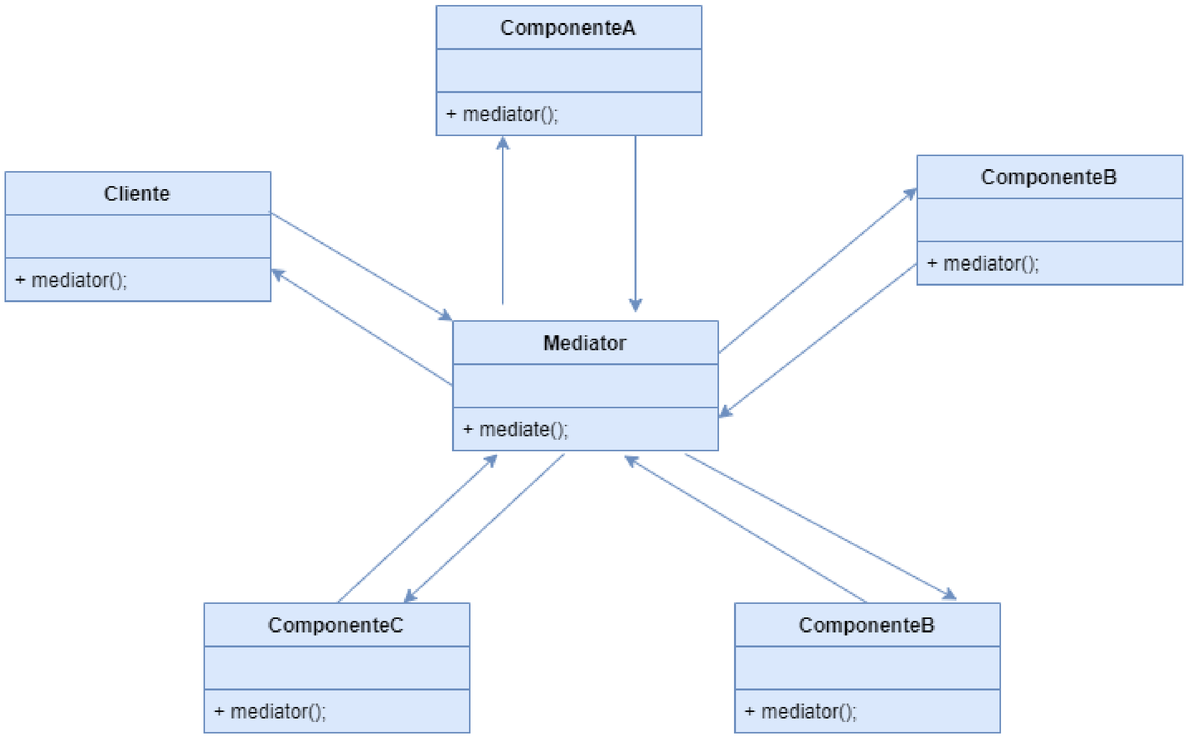

Figure 1: UML Diagram Mediator Pattern

Applications

The use of the Mediator pattern is recommended when:

- The objects of the application communicate in a well-structured but potentially complex way.

- The identities of objects must be protected even when they communicate with each other.

- El comportamiento de algunos objetos puede ser agrupado y personalizado.

- The reuse of an object is complicated because this reference and communicates with many other objects.

Design Patterns Collaborators

- Mediator classes frequently use the Observer design pattern to receive notifications of different requests from interacting classes.

- You can use the Adapter pattern to make the Mediator class independent of the specific classes it manages.

Scope of action

Applied at the object level.

Problem

All object-oriented applications base their operation on the interaction between objects. If there is no clearly defined mechanism of interaction, a highly coupled application can be generated, contradicting the principles of object-oriented programming.

Solution

The Mediator design pattern coordinates interactions between related objects; centralizing in one class the logic that performs the state changes of the objects, so that it offers a systematized way to increase cohesion (the logic is centralized) and reduce the coupling between classes (the dependencies between them are reduced).

Diagram or Implementation

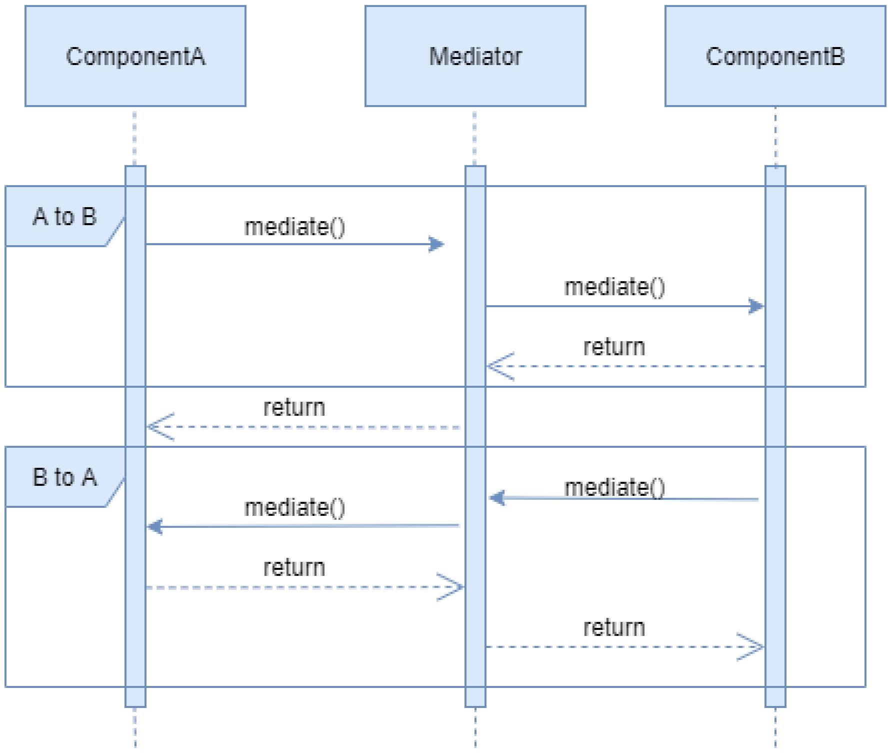

Figure 2: UML Diagram Mediator Pattern

Figure 2 explains the behaviour of the pattern by means of a sequence diagram.

- The class ComponentA wants to communicate with ComponentB and sends it a message via the mediator.

- The componete mediator can analyze the message for debugging purposes, tracking or to channel the message to the recipient.

- The message is delivered to the recipient and returns a response to the mediator component.

- The mediator receives the response and redirects it to ComponentA.

- Similarly, the process can be repeated from ComponentB to ComponentA by repeating the previous steps achieving two-way communication.