Proxy

Objective

Provide a substitute for an object so that access to it can be controlled.

Function

Maintains a representative of an object

Structure

The Client must work with services and proxies through the same interface. This way, you can pass a proxy to any code that wait for an item of service. The proxy class has a reference field that points to an object of service.

The structure that meets this pattern is shown in Figure 1

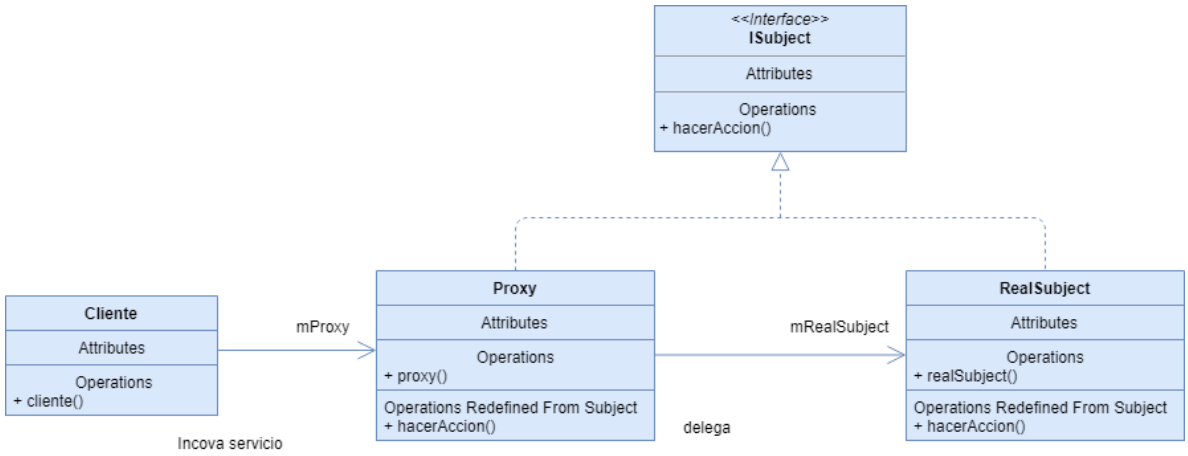

Figure 1: UML Diagram Proxy Pattern

Applications

The use of the proxy pattern is recommended when

- The system requires a remote representation for an object in a different place.
- You want to hide the complexity of an object by representing it with a simple that does not require further knowledge in order to facilitate its use.
- Objects must have different access points, controlling access to the original object, without this meaning that the object is instantiated in different places.

Design Patterns Collaborators

- Both the Proxy and Decorator patterns have similar applications; however care should be taken since they serve different purposes, a Decorator adds one or more responsibilities to an object while a Proxy controls access to the object.

Scope of action

Applied at the object level.

Problem

In order to support the objects at the moment they are required, all objects must be pre-installed in such a way that they find ready for the client to use them; however, it implies a high resource consumption.

Solution

The proxy pattern allows you to control access to an object by instantiating it at the time the object is actually used, using a proxy" image object that is always present as a representation of the original, and is in charge of urging it only when required; in such a way that it puts provision a more versatile and sophisticated reference than a simple access point to an object; using three options: a "remote proxy" hiding the fact that a object resides in a different space or "virtual proxy" optimizing the creation of a object at the time it is required or both ways using references intelligent performing additional tasks when the object is accessed.

Diagram or Implementation

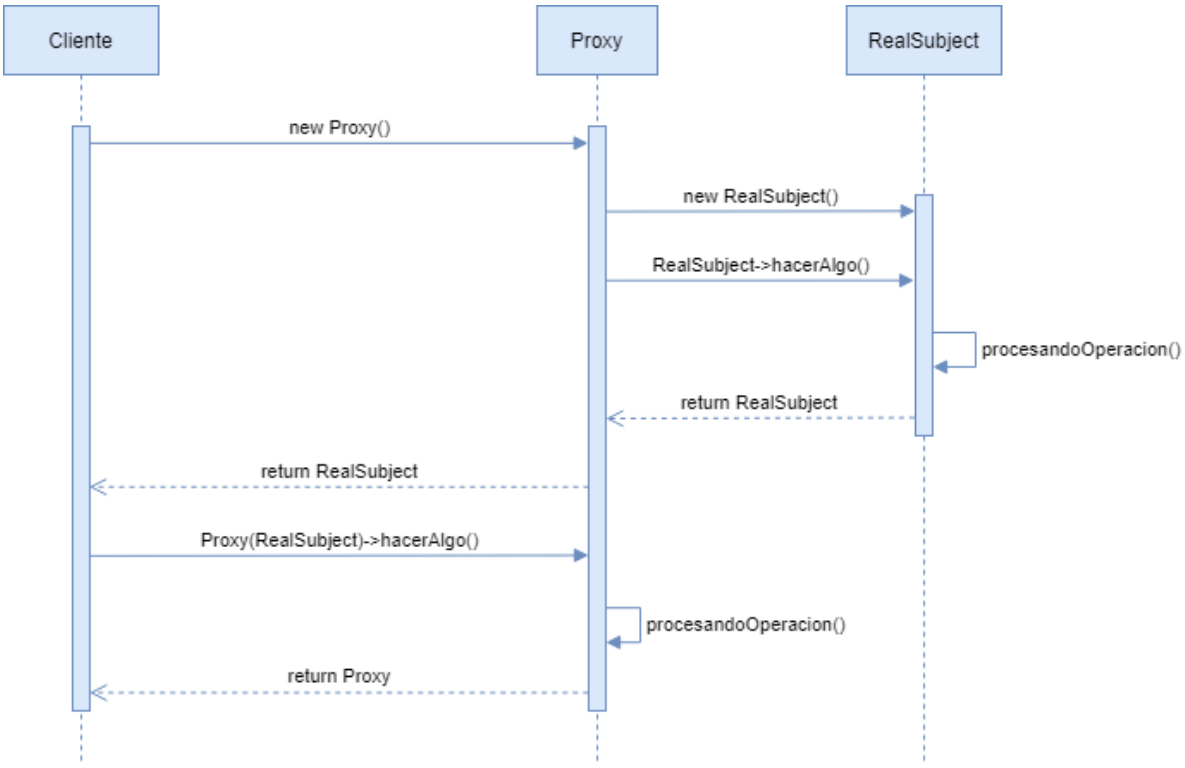

Figure 2: UML Diagram Proxy Pattern

Figure 2 explains the behavior of the proxy pattern using a diagram of sequence.

- The client class requests an Object from the Factory component.
- Component Factory creates a proxy that encapsulates the Object.
- Client class runs the proxy created by the Factory component.
- The Proxy class performs one or more actions prior to the execution of the Object.
- The Proxy class delegates execution to the Object component.
- The Proxy class performs one or more actions after the execution of the Object.
- The proxy class returns a result.