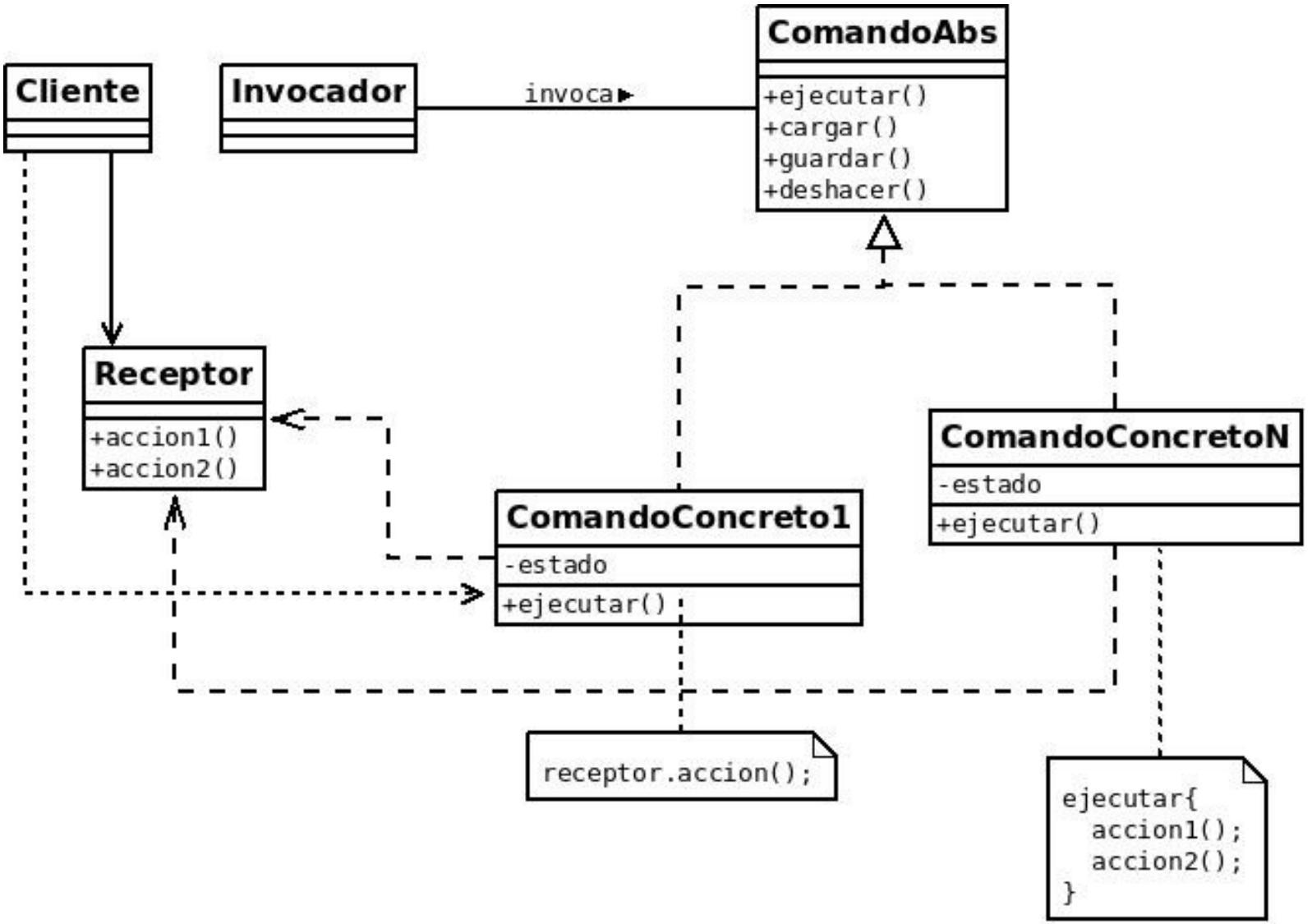
# PATRÓN COMMAND

Es un patrón de diseño de comportamiento que encapsula una petición en un objeto, permitiendo asi parametrizar las distintas peticiones, o bien llevar un registro de las mismas para poder deshacerlas.

# Command - motivación

A veces es necesario enviar peticiones a objetos sin saber nada acerca de la operación solicitada o de quién es el receptor de la petición.

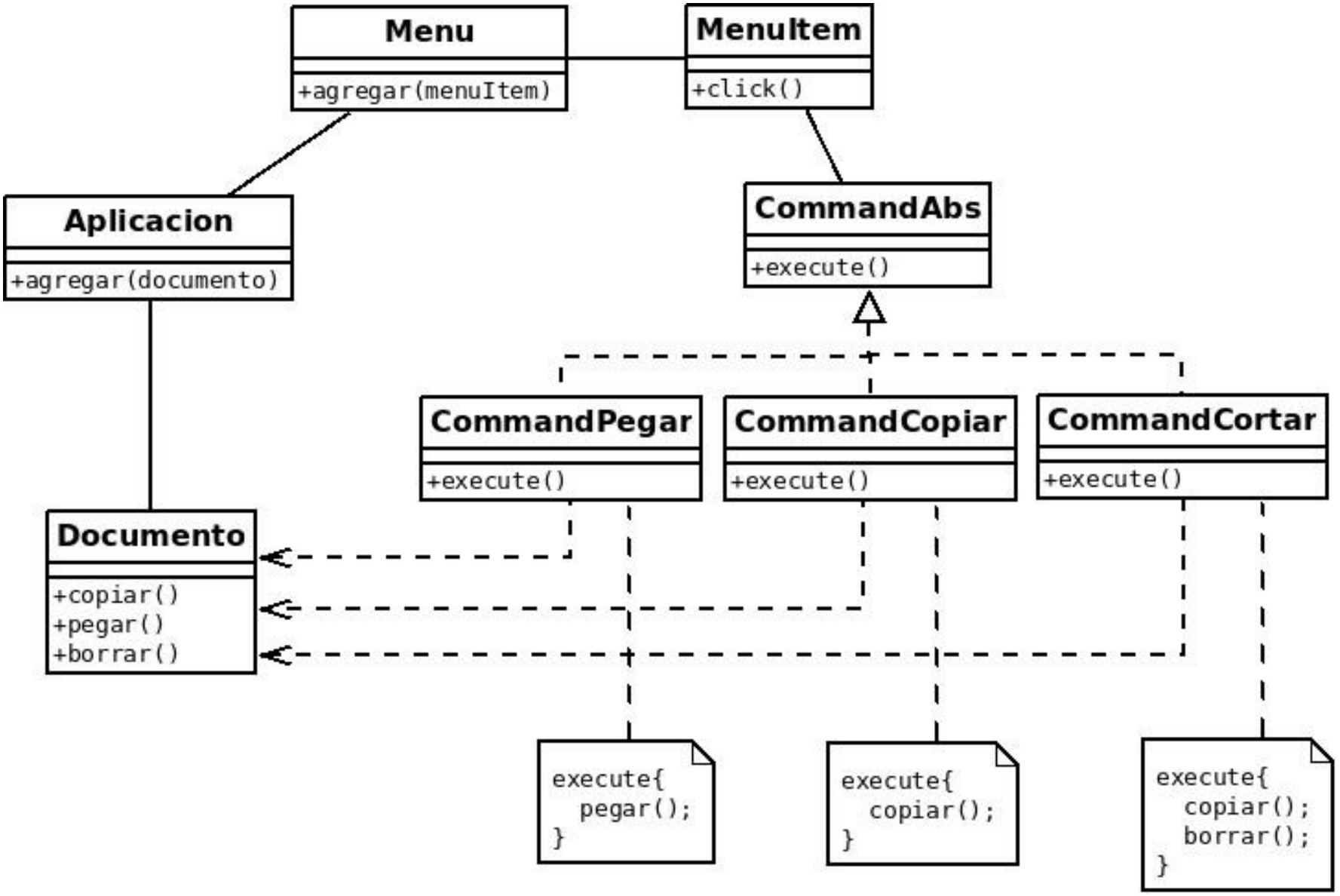
# Command - estructura



# Command – estructura cont.

- Command: declara una interfaz para ejecutar una operación.
- CommandConcrete: define un enlace entre un objeto receiver y una acción. Implementa execute invocando la/s correspondiente/s operación/es del receiver.
- Cliente: crea un objeto CommandConcrete y establece su receiver.
- Invoker: solicita a la orden que ejecute la petición.
- Receiver: sabe como llevar a cabo las operaciones asociadas a una petición.

# Command - ejemplo



# Command - beneficios

- Desacopla el objeto que invoca la operación de aquel que sabe como realizarla.
- Los Commands son objetos, por lo que pueden ser manipulados y extendidos como cualquier otro objeto.
- Se pueden ensamblar commands en un Command compuesto.
- Es facil añadir nuevos commands ya que no se cambian las clases existentes.