

# Tecnología de Programación

*Martín L. Larrea*

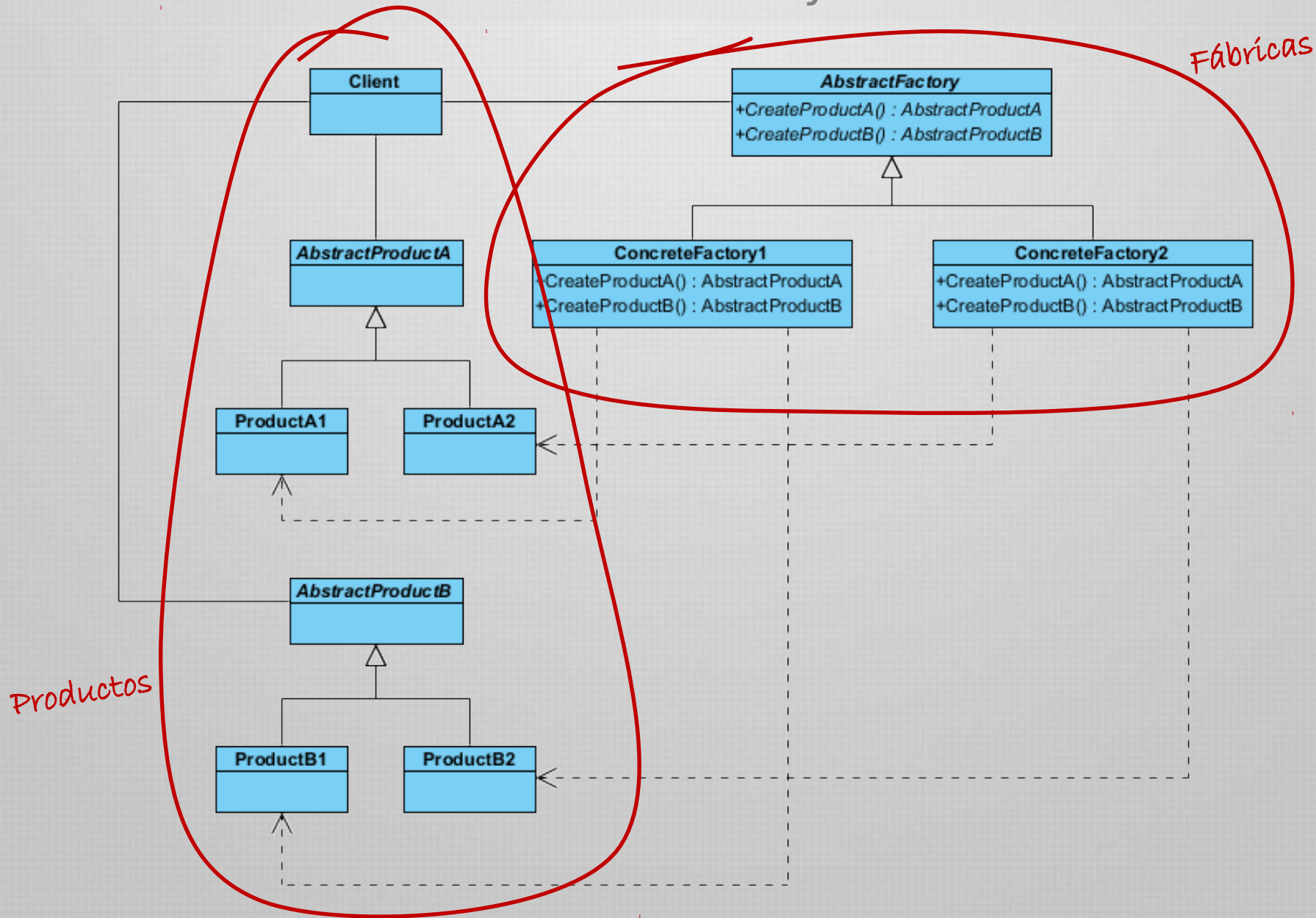
Departamento de Ciencias e Ingeniería de la Computación  
Universidad Nacional del Sur

# Patrones GoF

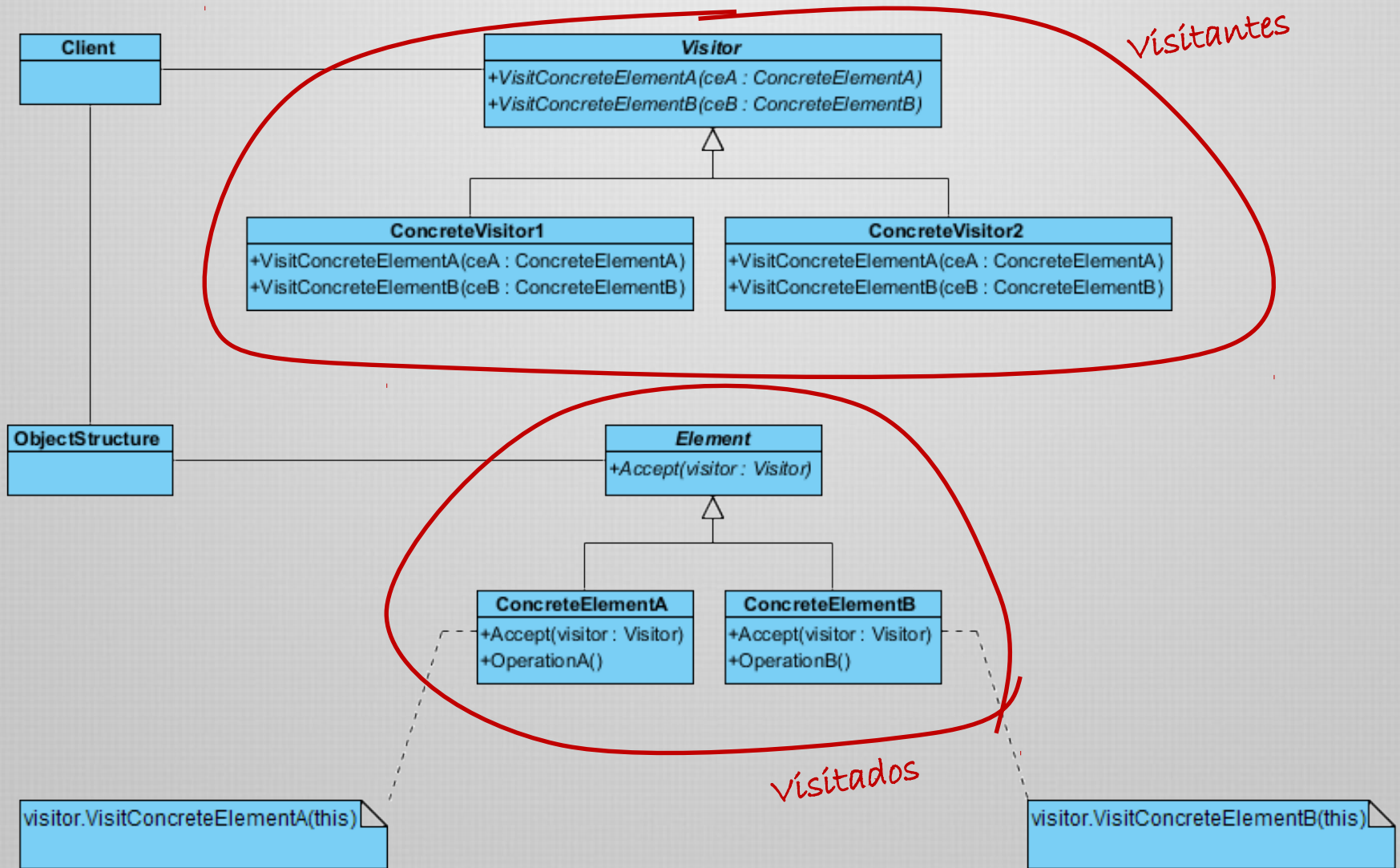


		PROPÓSITO		
		CREACIONAL	ESTRUCTURAL	COMPORTAMIENTO
SCOPE	CLASE	Factory Method	Adapter	Interpreter Template Method
	OBJETO	Abstract Factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Facade Proxy	Chain of Responsibility Command Iterator Mediator Memento Flyweight Observer State Strategy Visitor

# Abstract Factory

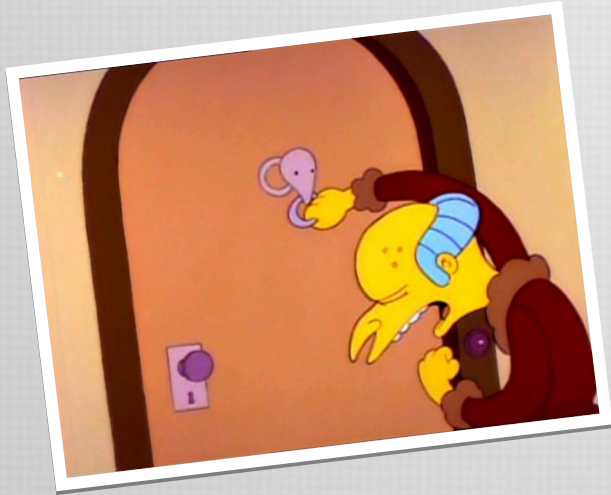


# Visitor





# Visitor



`casa.accept(burns)`

Le pedimos al visitado que  
acepte el visitante



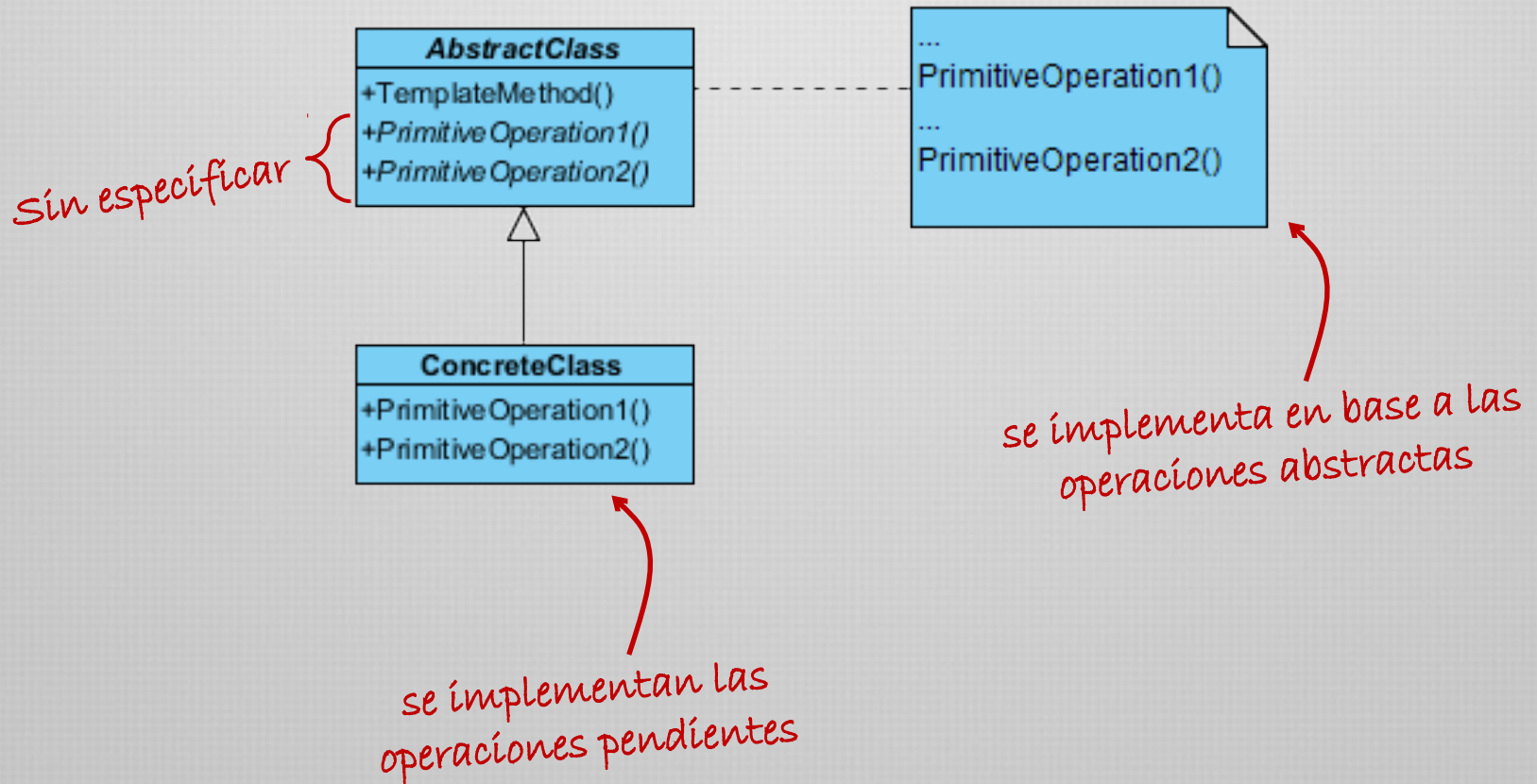
La casa dice:  
`burns.visit(this)`

al aceptar, el visitado le pide al  
visitador que "pase" a esta casa

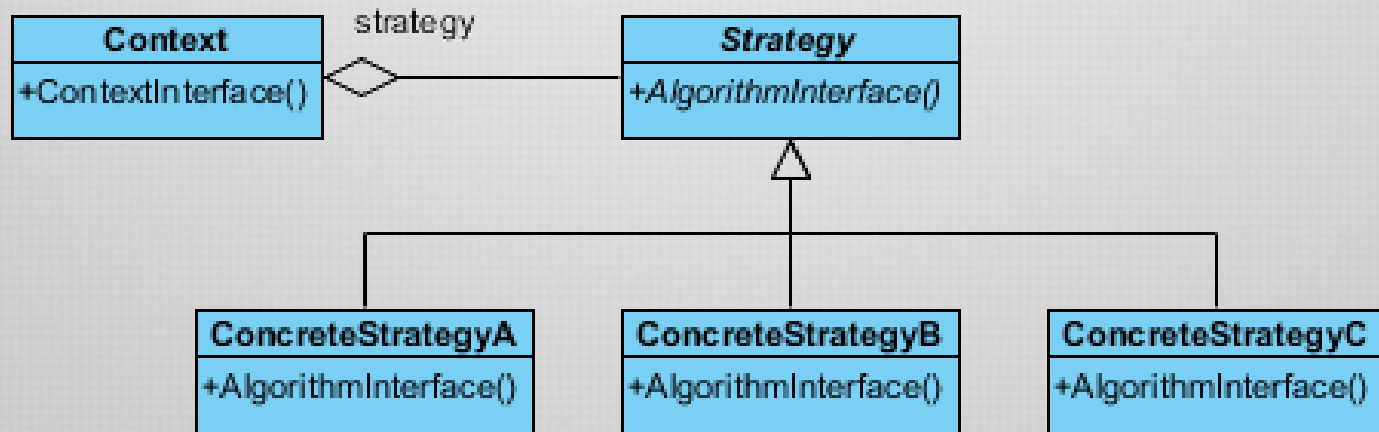


...ahora el visitante  
puede acceder a la  
casa y hace su tarea

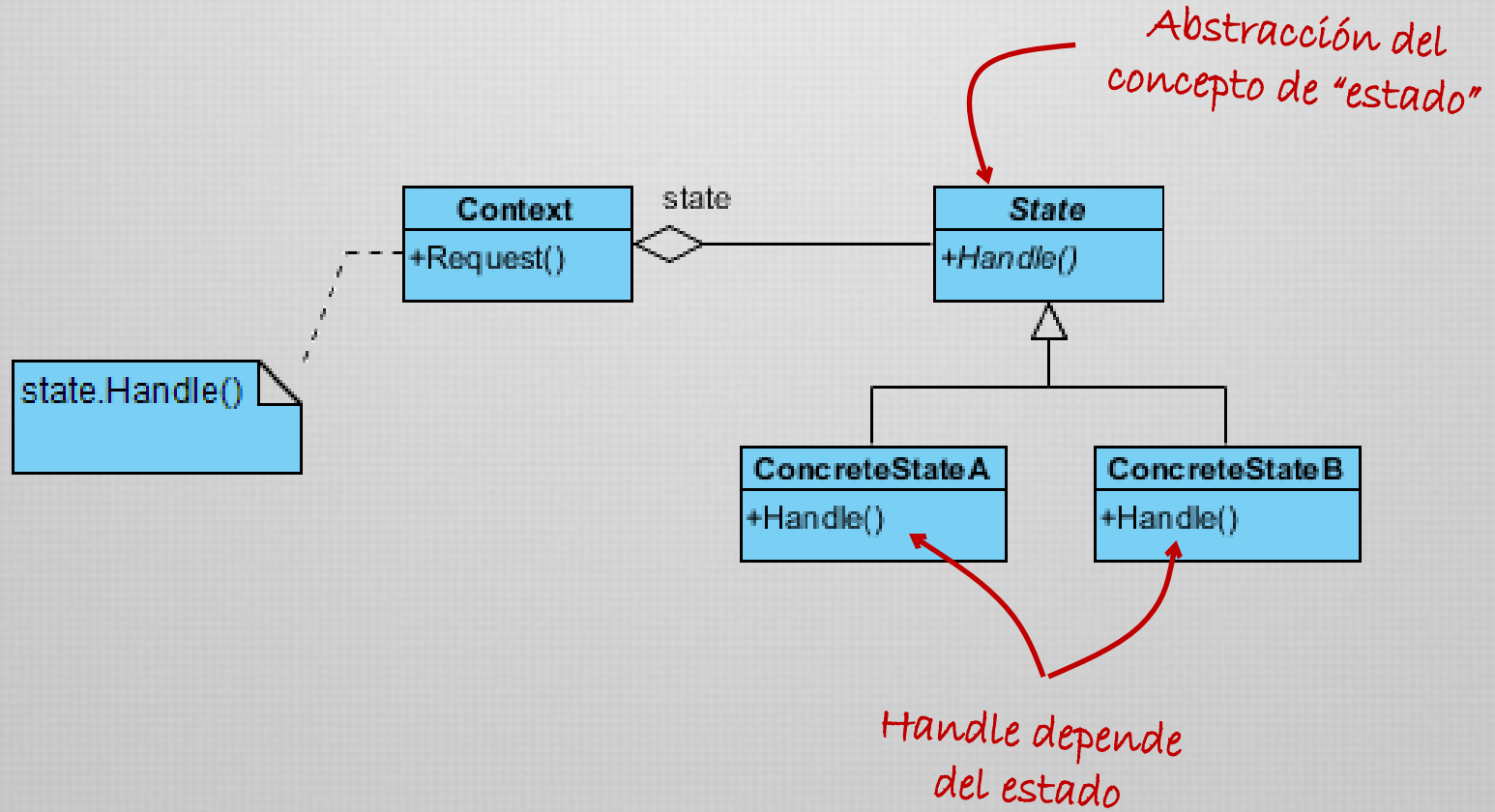
# Template Method



# Strategy



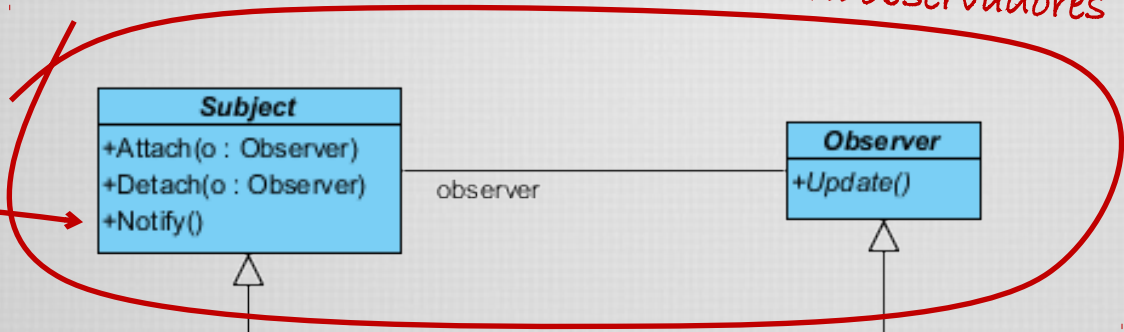
# State





# Observer

*Subject administra observadores*



*notifica a todos los observer (update)*

return subjectState

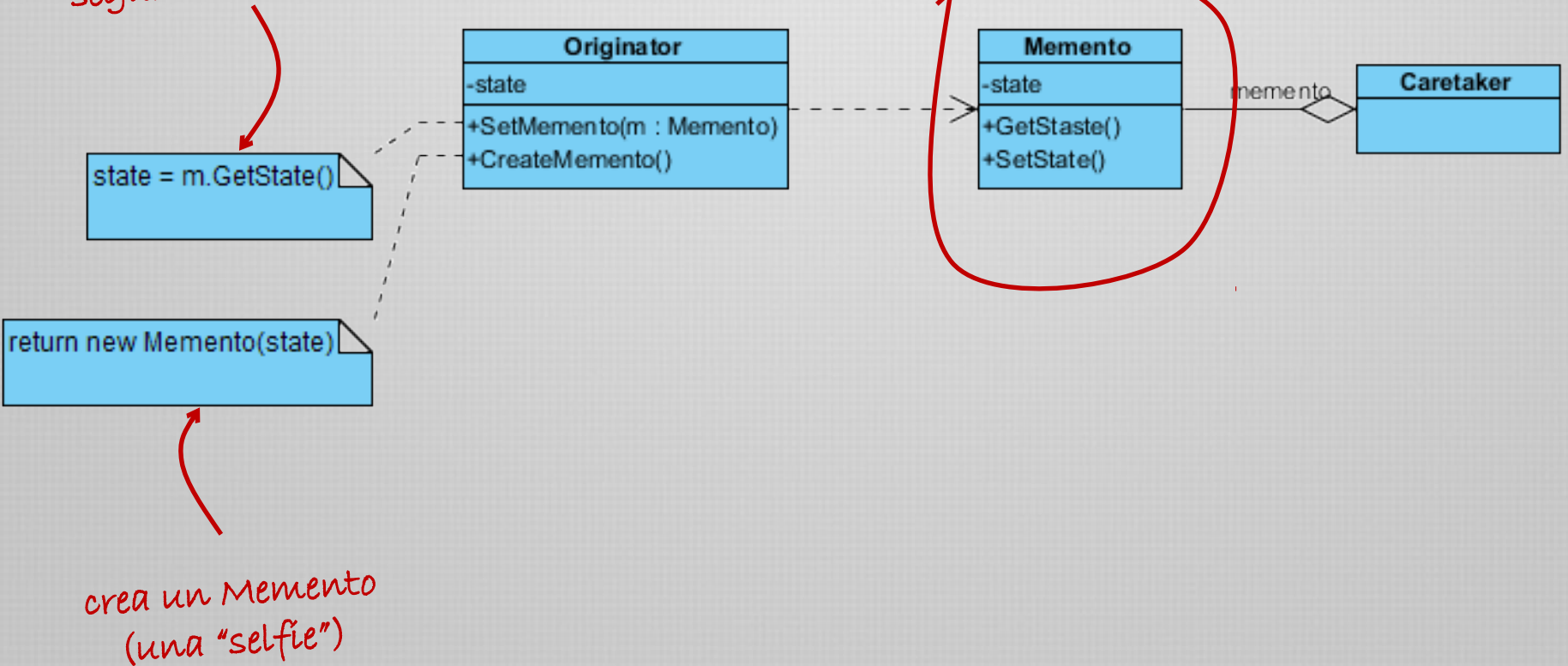
observerState = subject.GetState()

*observa el Subject concreto y actúa según sus objetivos*

# Memento

restaura su estado según un Memento

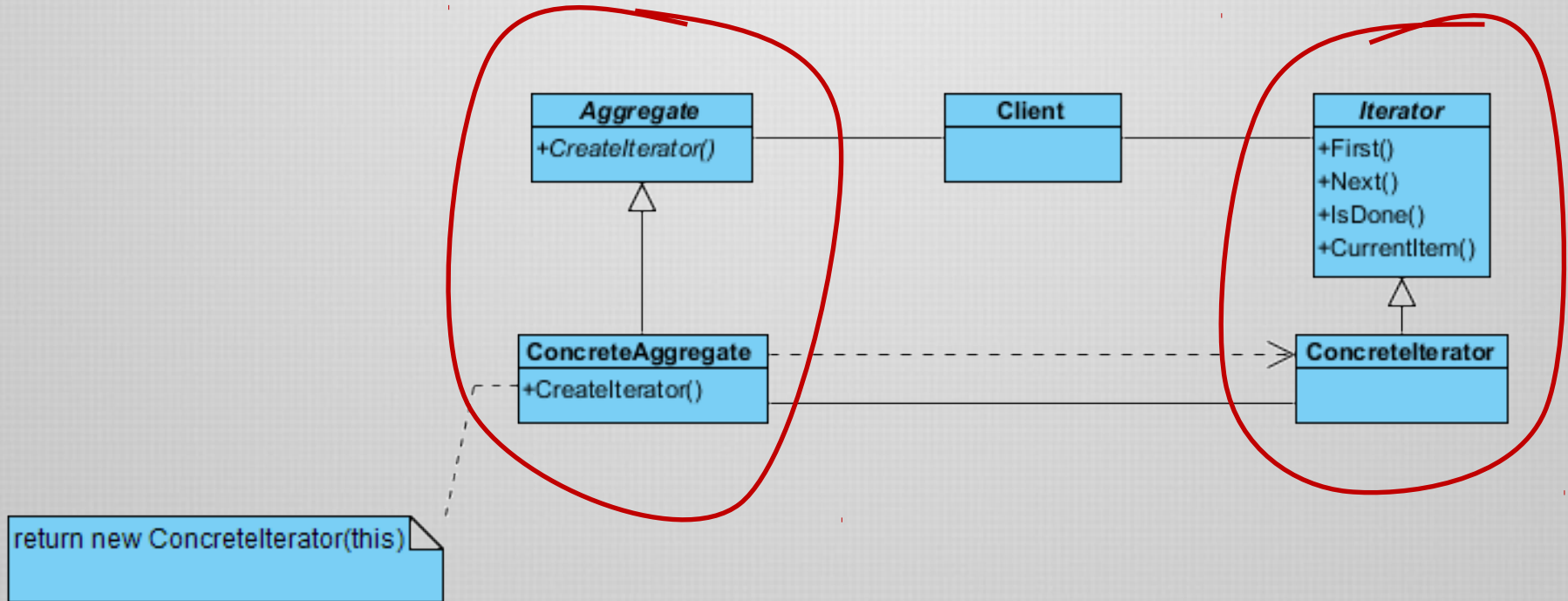
"Fotografía" del estado del Originator



# Iterator

*Estructura a recorrer*

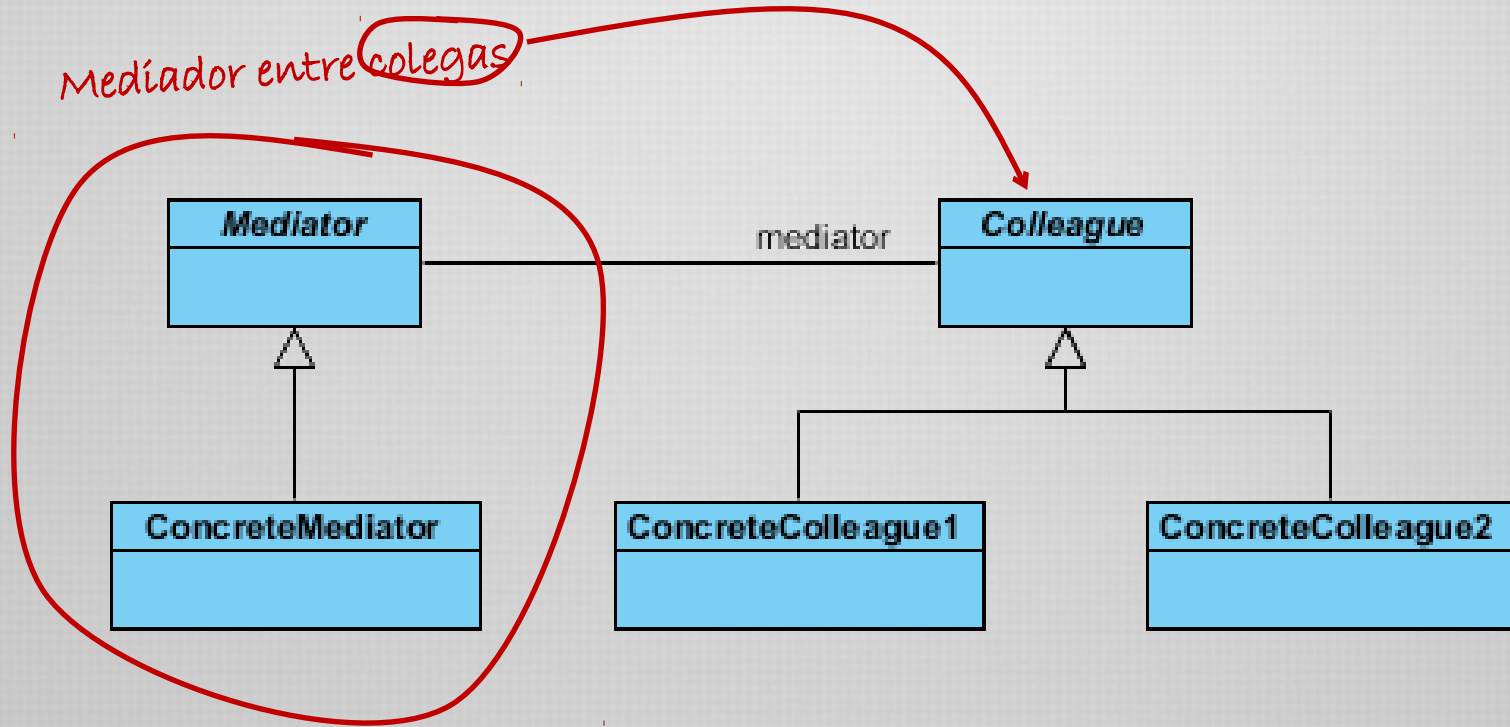
*Iterador*



*Es la estructura la que crea su iterador para ser recorrida*

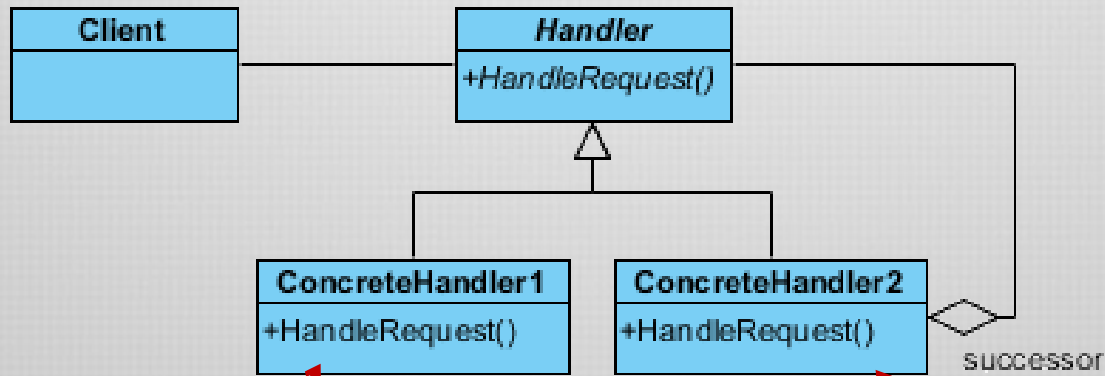
# Mediator

mediador entre colegas





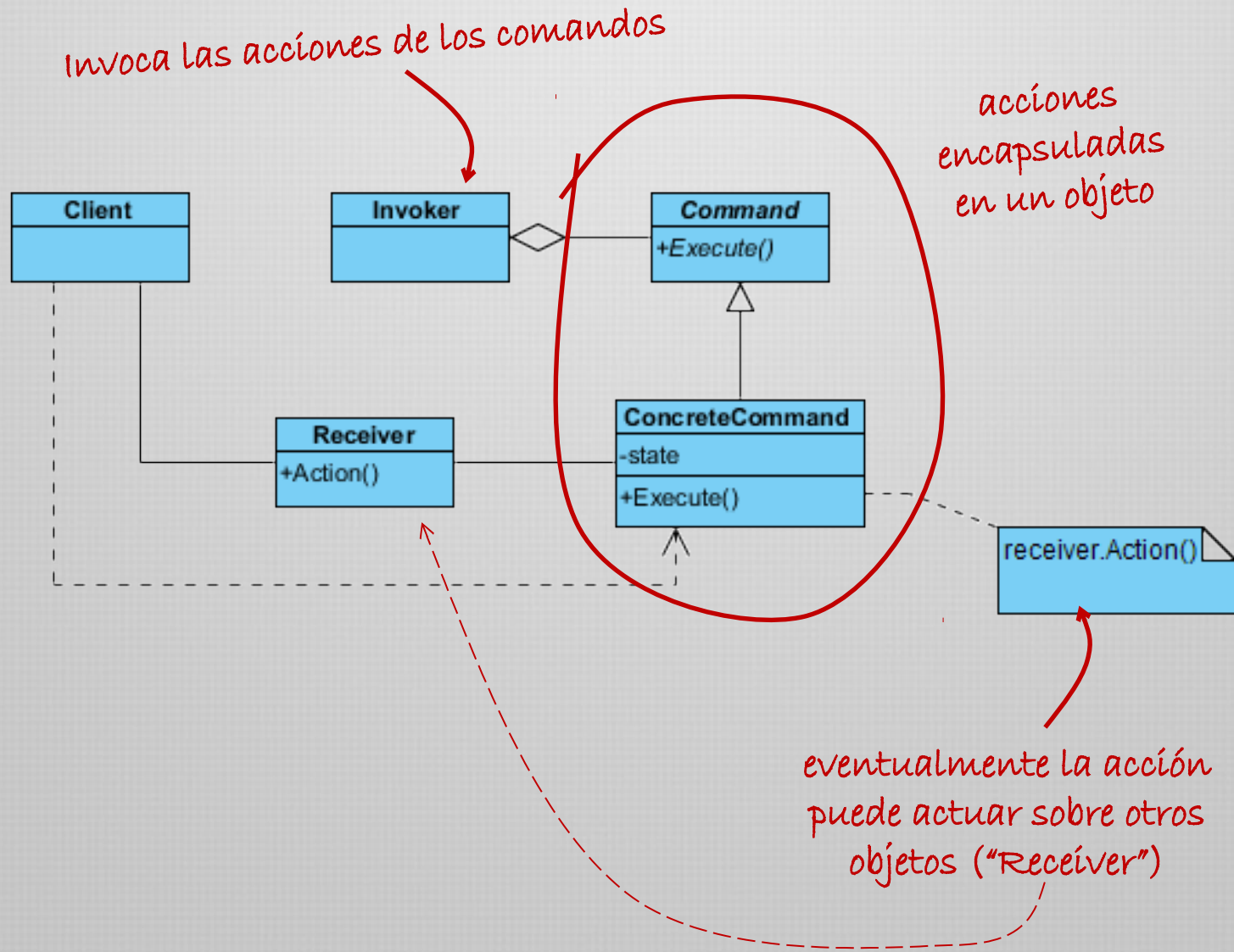
# Chain of Responsibility



*Realiza una  
tarea puntual*

*Realiza una tarea  
puntual  
y ordena al sucesor a  
realizar su propia tarea*

# Command

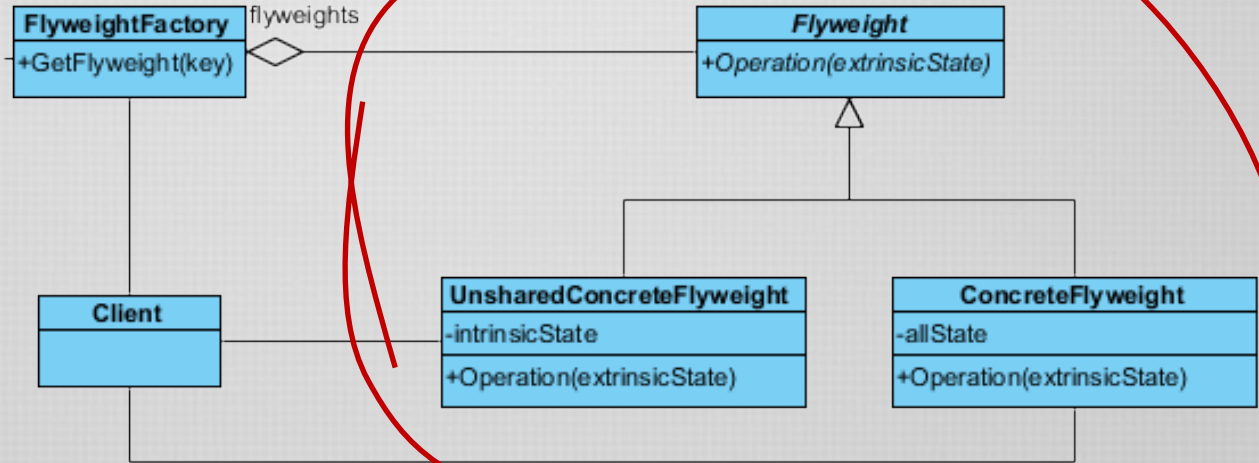


# Flyweight

fábrica de objetos

objetos administrados para compartir

if flyweights[key] exists  
return existing flyweight  
else  
create new flyweight  
add to pool of flyweights  
return new flyweight



LetraTexto
cod_ASCII
cod_UNKCODE
color
subrayado
bold
italic
posicionTexto

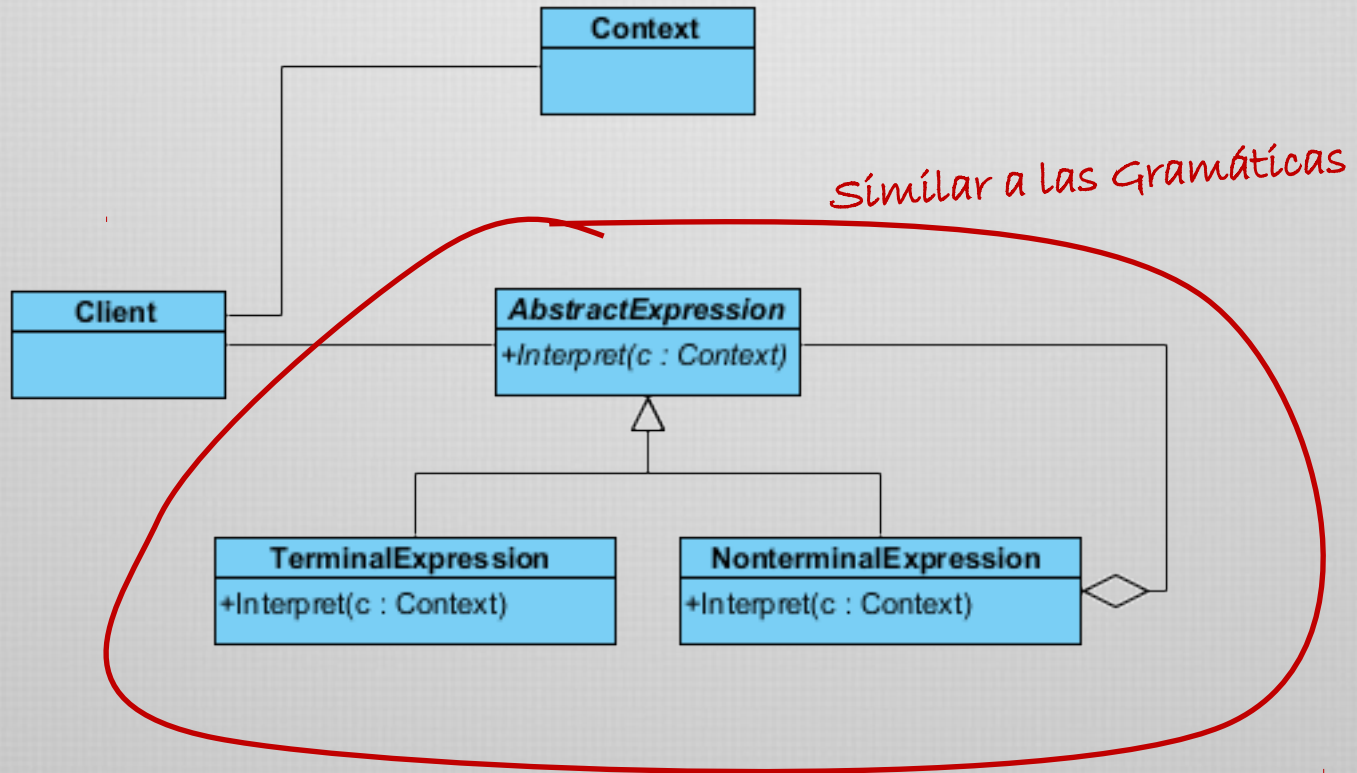
=

Letra
cod_ASCII
cod_UNKCODE

+

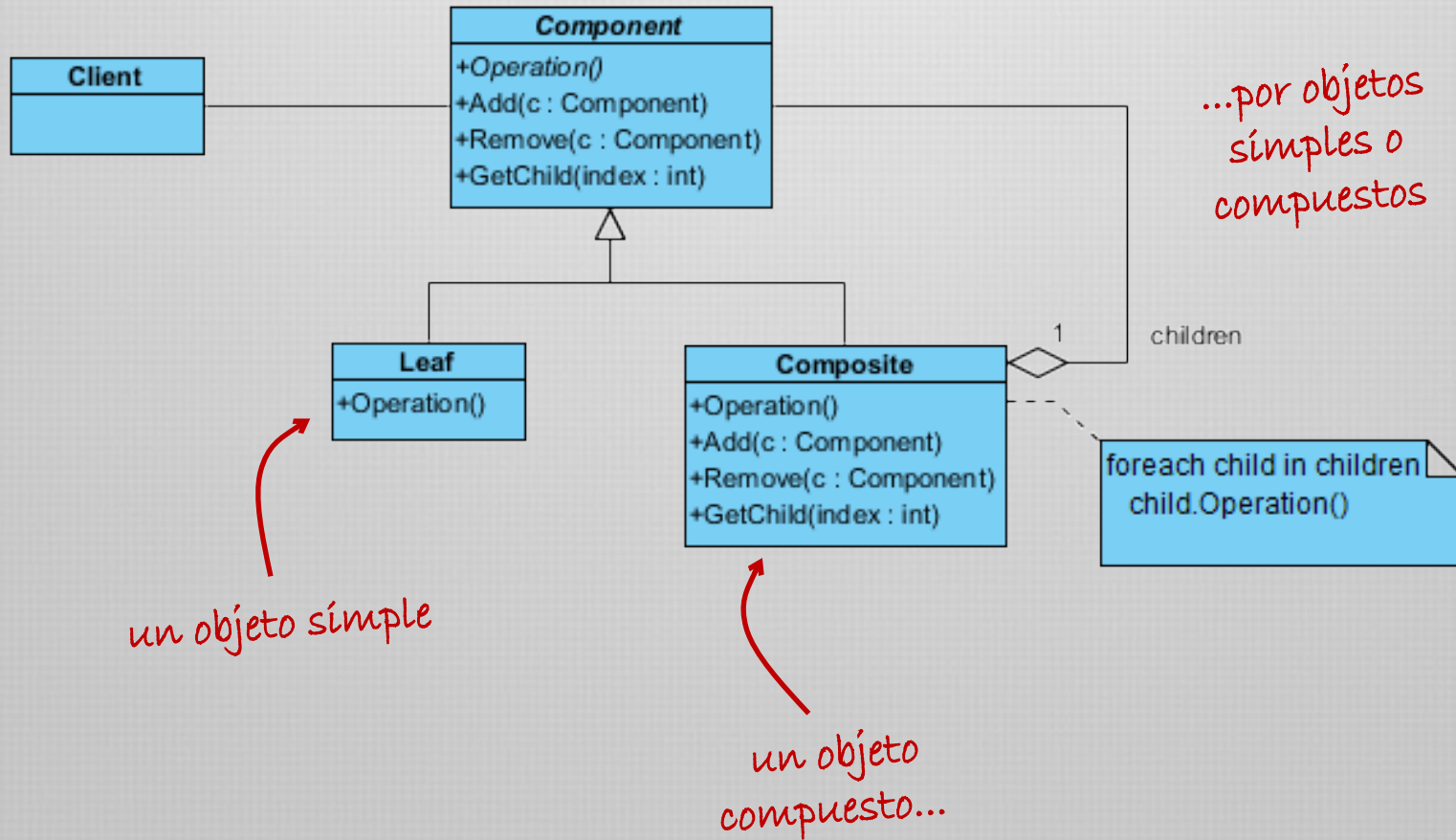
LetraT
color
subrayado
bold
italic
posicionTexto

# Interpreter

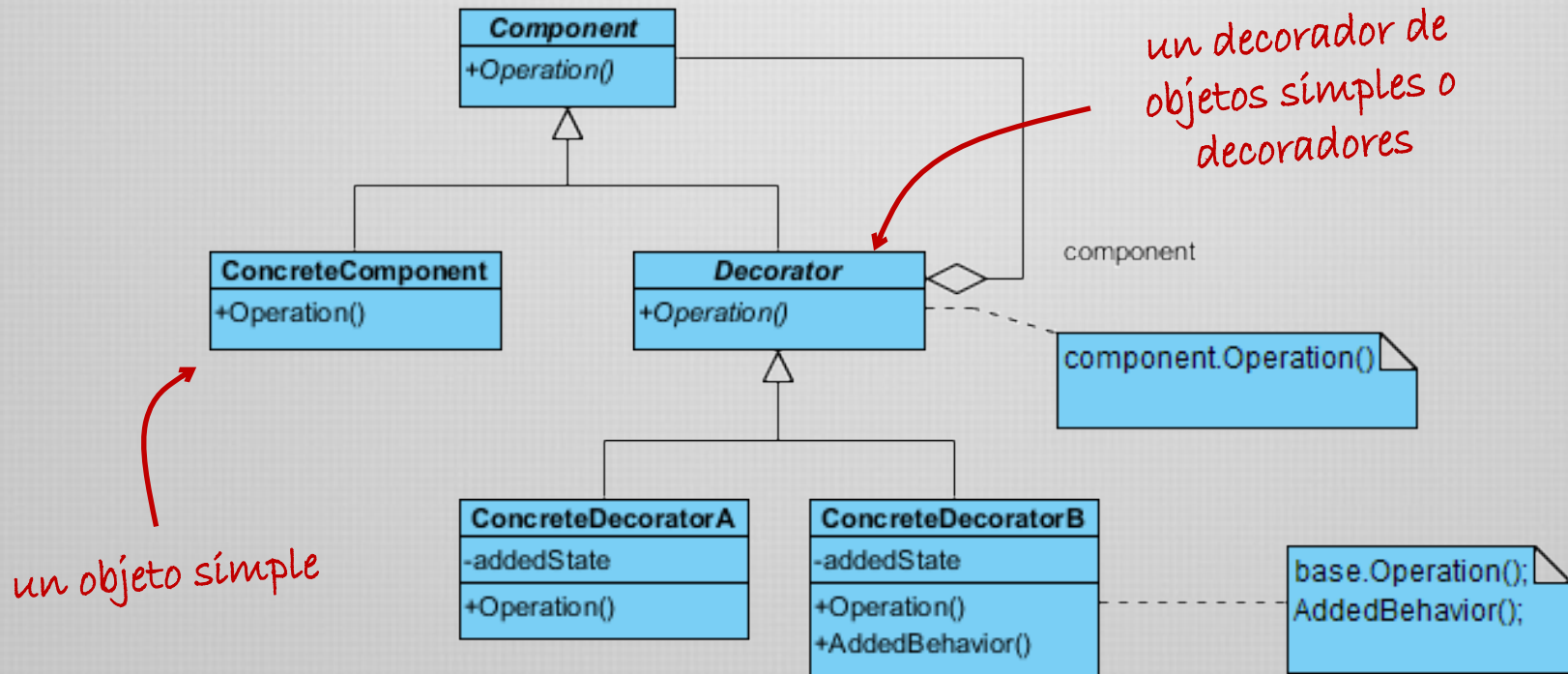




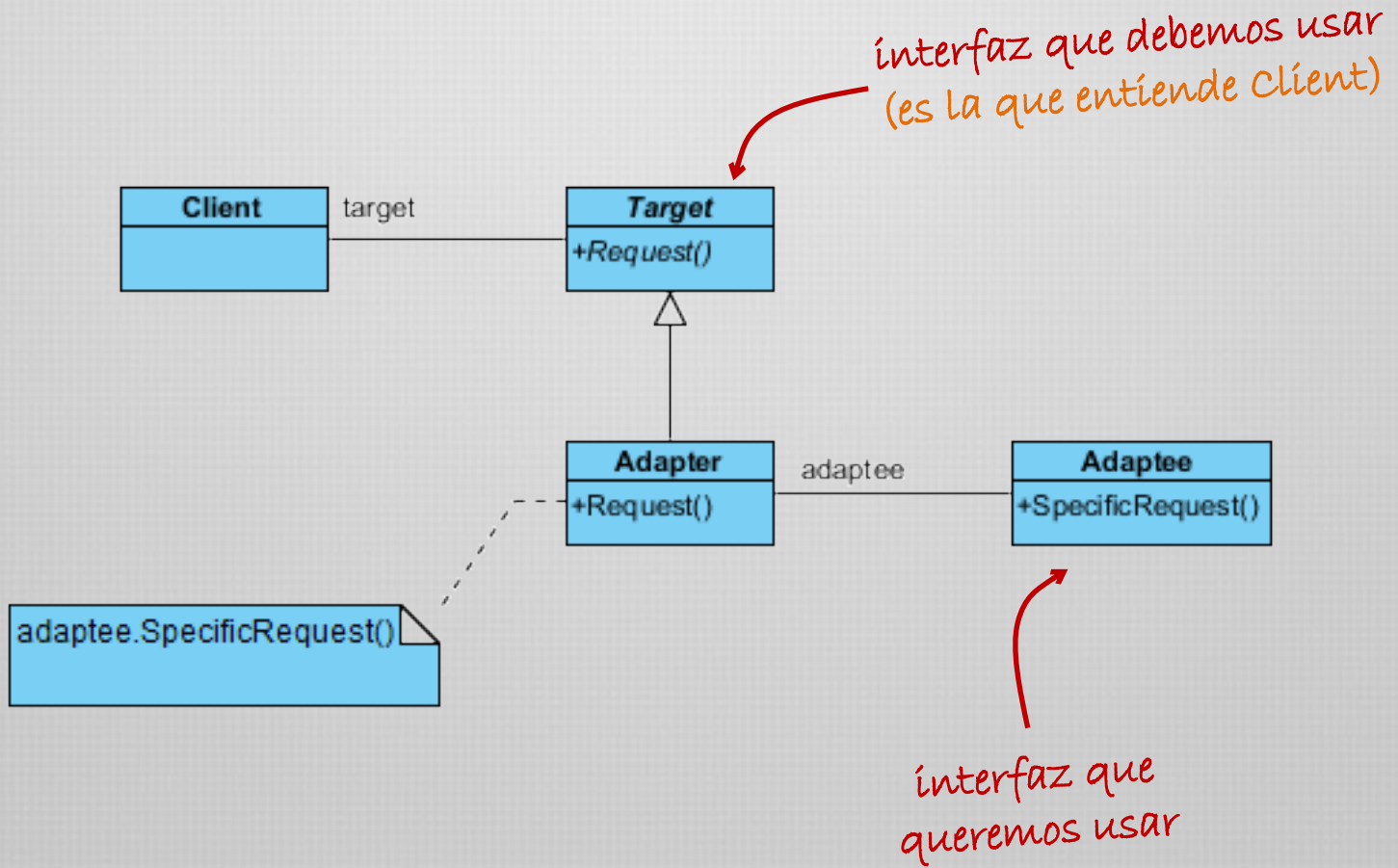
# Composite



# Decorator

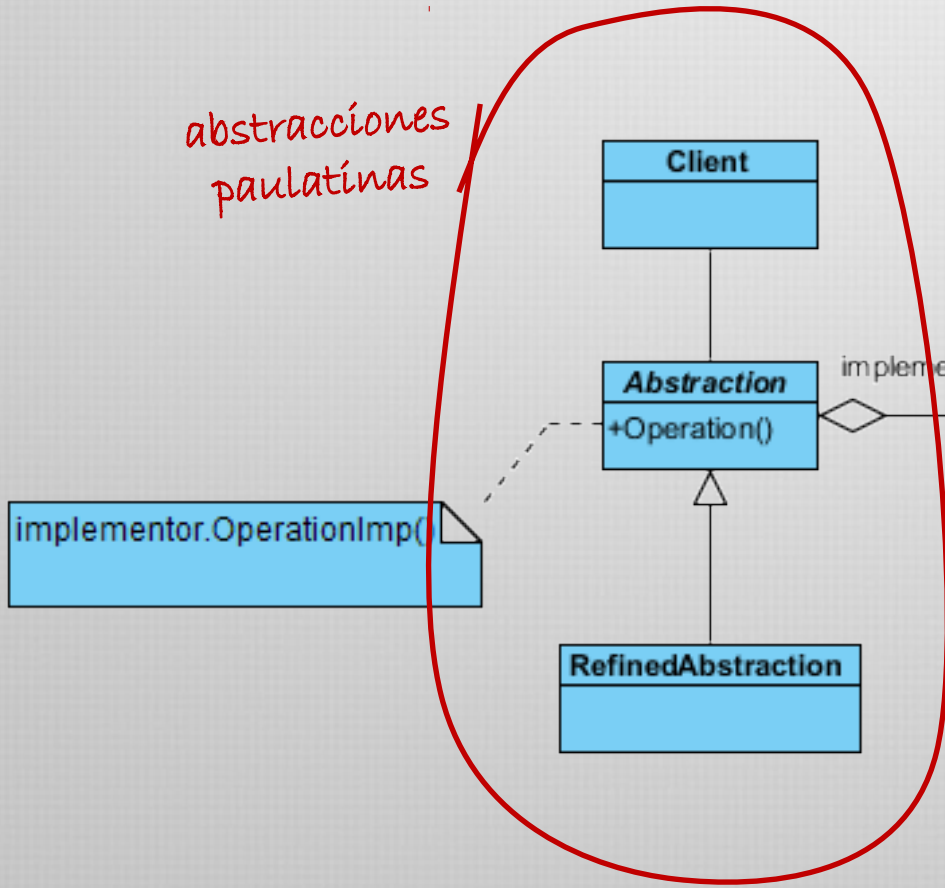


# Adapter

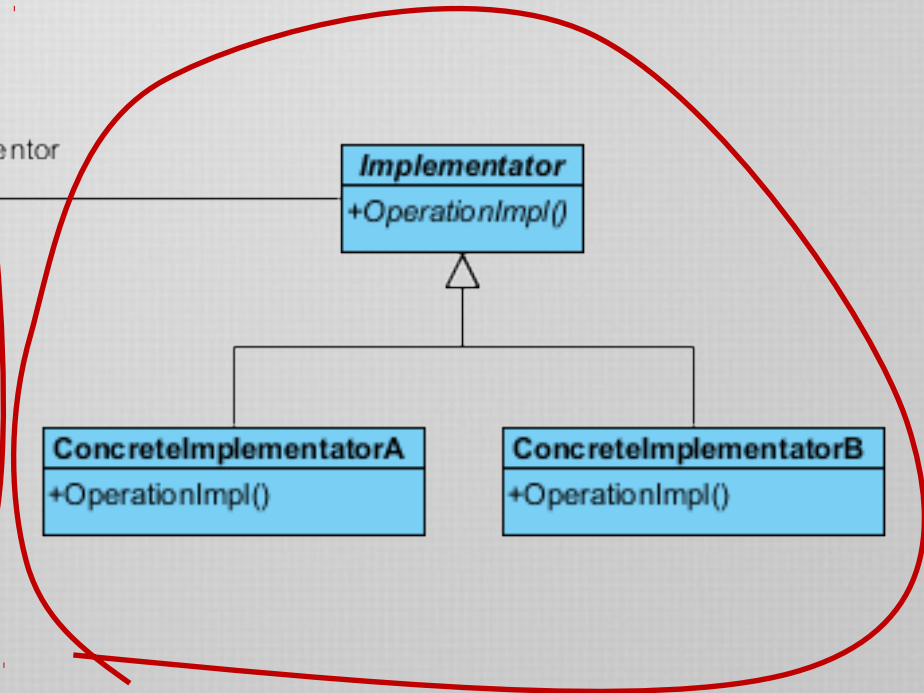


# Bridge

abstracciones paulatinas



implementaciones paulatinas

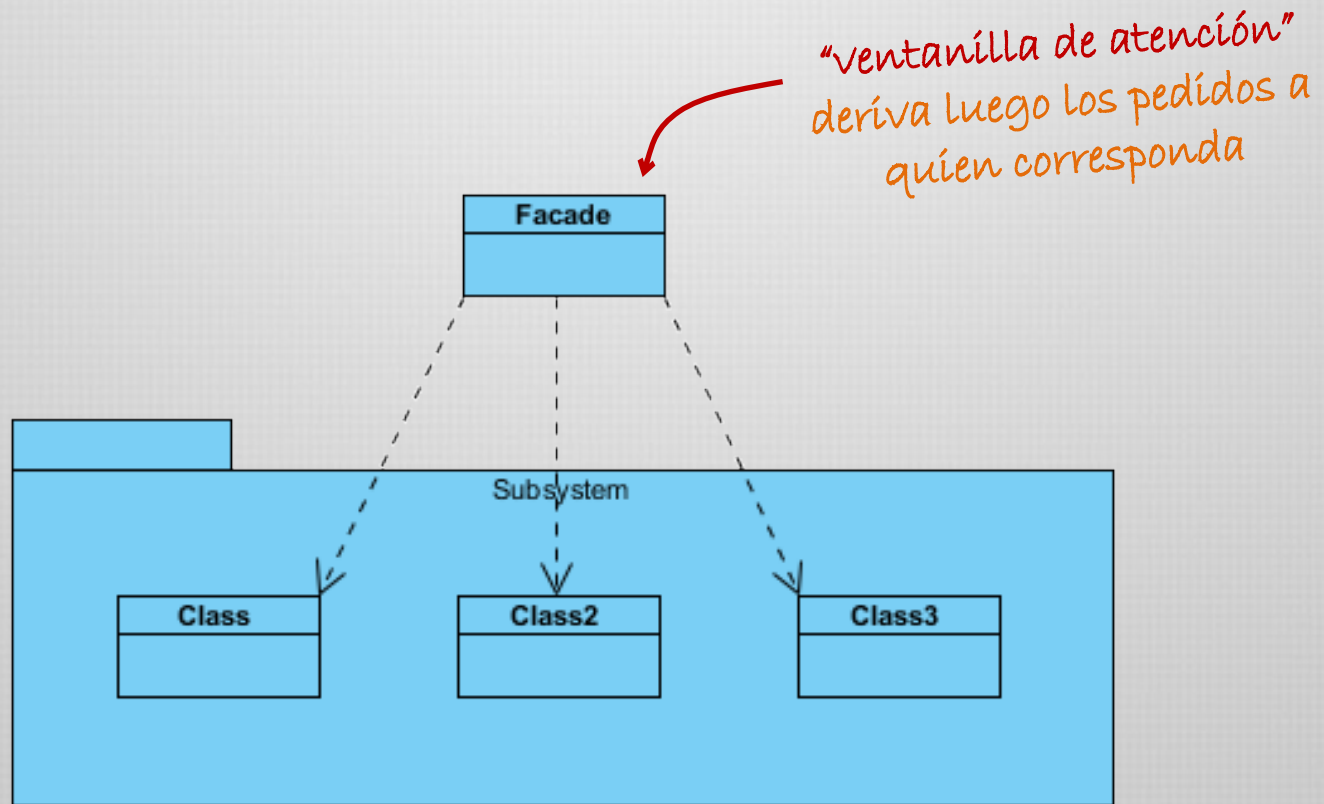


éstas abstracciones...

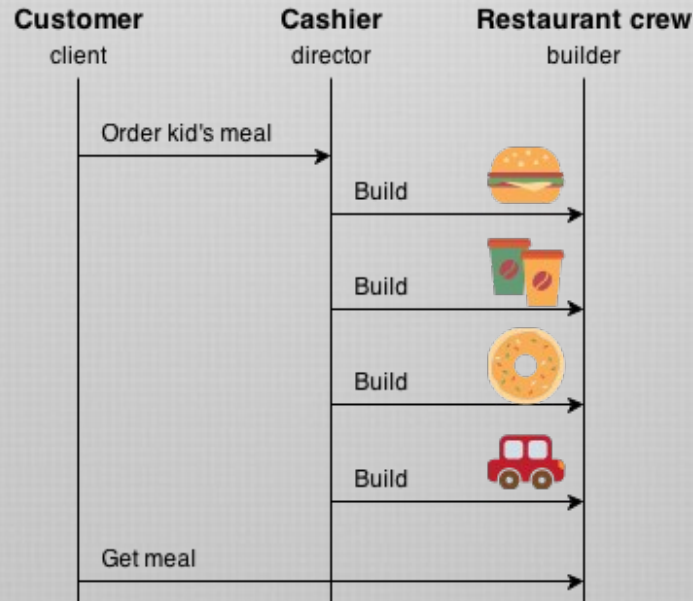
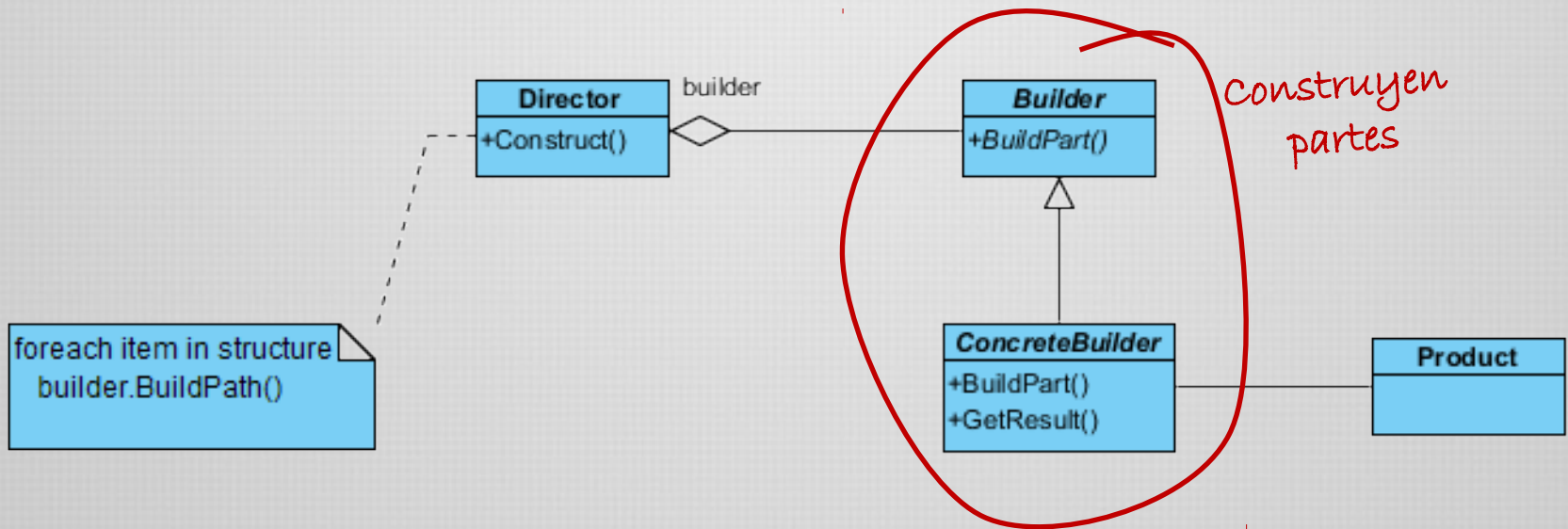
... se implementan de éstas formas posibles



# Facade

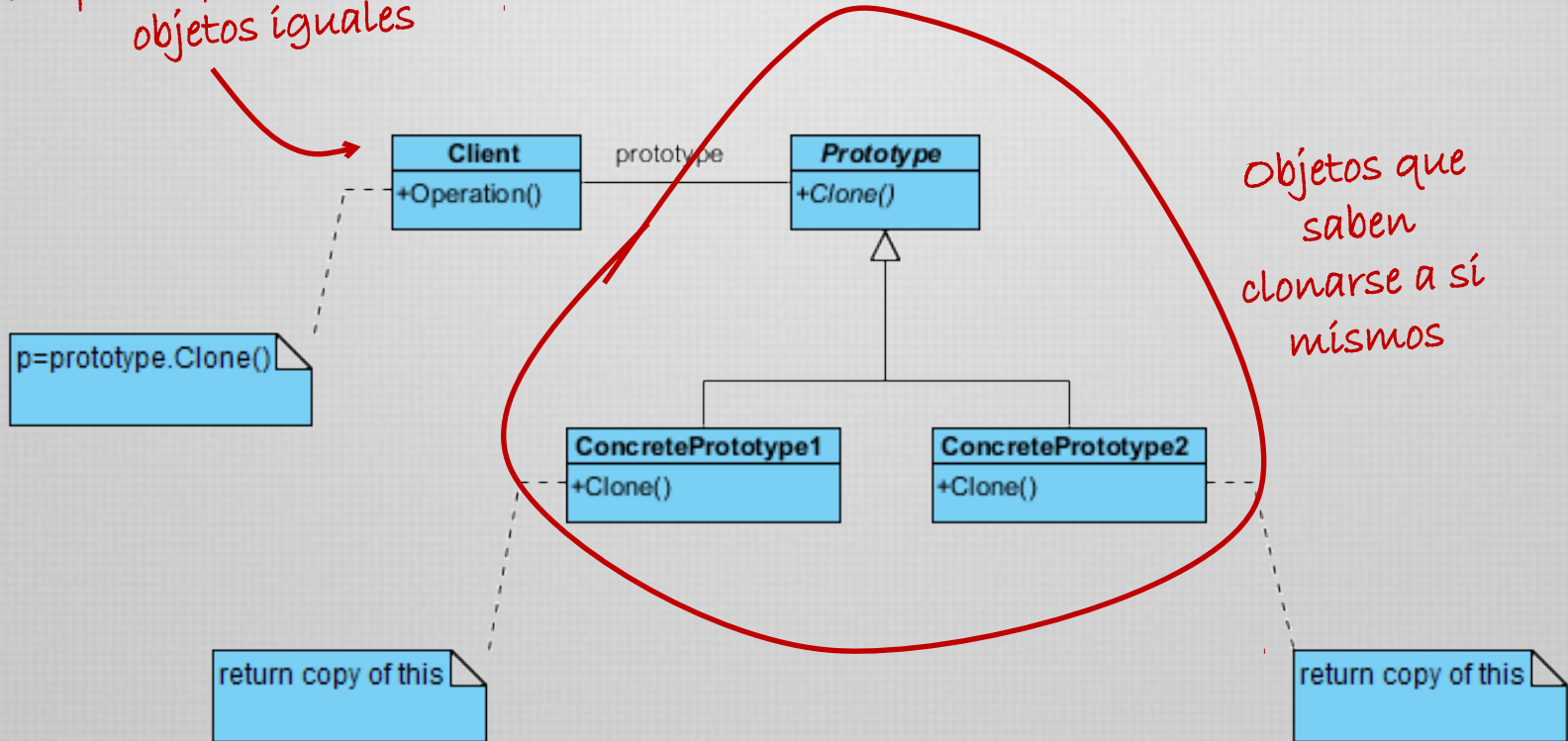


# Builder



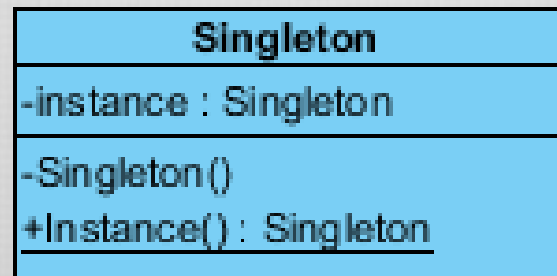
# Prototype

usa un prototipo para obtener varios objetos iguales



# Singleton

*el único objeto está  
confinado aquí*



# Factory Method

