



strategy

<https://refactoring.guru/design-patterns/strategy>

# Tema 3: Disseny

Anna Puig

Enginyeria Informàtica

Facultat de Matemàtiques i Informàtica,

Universitat de Barcelona

Curs 2021/2022



UNIVERSITAT DE  
BARCELONA



# Temari

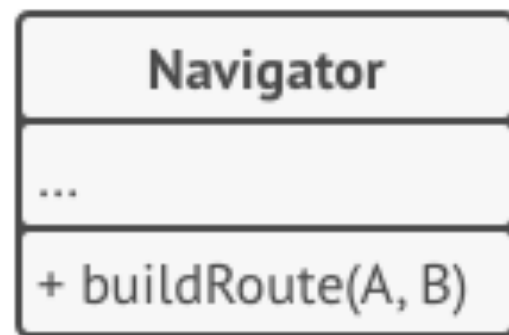
1	Introducció al procés de desenvolupament del software	
2	Anàlisi de requisits i especificació	
3	Disseny	
4	Del disseny a la implementació	
5	Ús de frameworks de testing	
		3.1 Introducció
		3.2 Patrons arquitectònics
		3.3 Criteris de Disseny: G.R.A.S.P.
		3.4 Principis de Disseny: S.O.L.I.D.
		3.5 <b>Patrons de disseny</b>

# 3.4. Patrons de disseny

Propòsit → Àmbit ↓	CREACIÓ	ESTRUCTURA	COMPORTAMENT
<b>CLASSE</b>	<ul style="list-style-type: none"> <li>• Factory method</li> </ul>	<ul style="list-style-type: none"> <li>• class Adapter</li> </ul>	<ul style="list-style-type: none"> <li>• Interpreter</li> <li>• Template method</li> </ul>
<b>OBJECTE</b>	<ul style="list-style-type: none"> <li>• Abstract Factory</li> <li>• Builder</li> <li>• Prototype</li> <li>• Singleton</li> <li>• Object pool</li> </ul>	<ul style="list-style-type: none"> <li>• Object Adapter</li> <li>• Bridge</li> <li>• Composite</li> <li>• Decorator</li> <li>• Facade</li> <li>• Flyweight</li> <li>• Proxy</li> </ul>	<ul style="list-style-type: none"> <li>• Chain of Responsibility</li> <li>• Command</li> <li>• Iterator</li> <li>• Mediator</li> <li>• Memento</li> <li>• Observer</li> <li>• State</li> <li>• <b>Strategy</b></li> <li>• Visitor</li> </ul>

# Patró Strategy: Problema

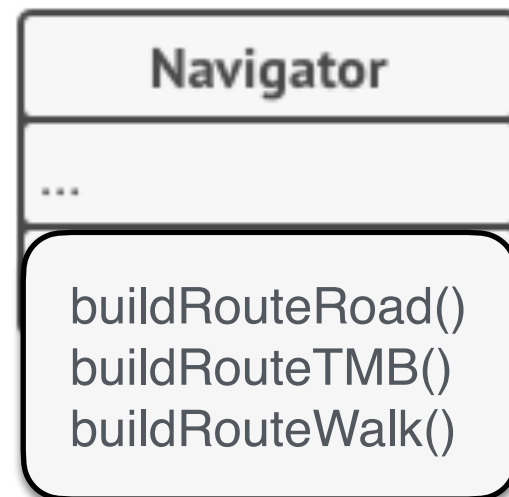
Es vol fer un algorisme que calculi la ruta entre un punt A i un punt B del mapa





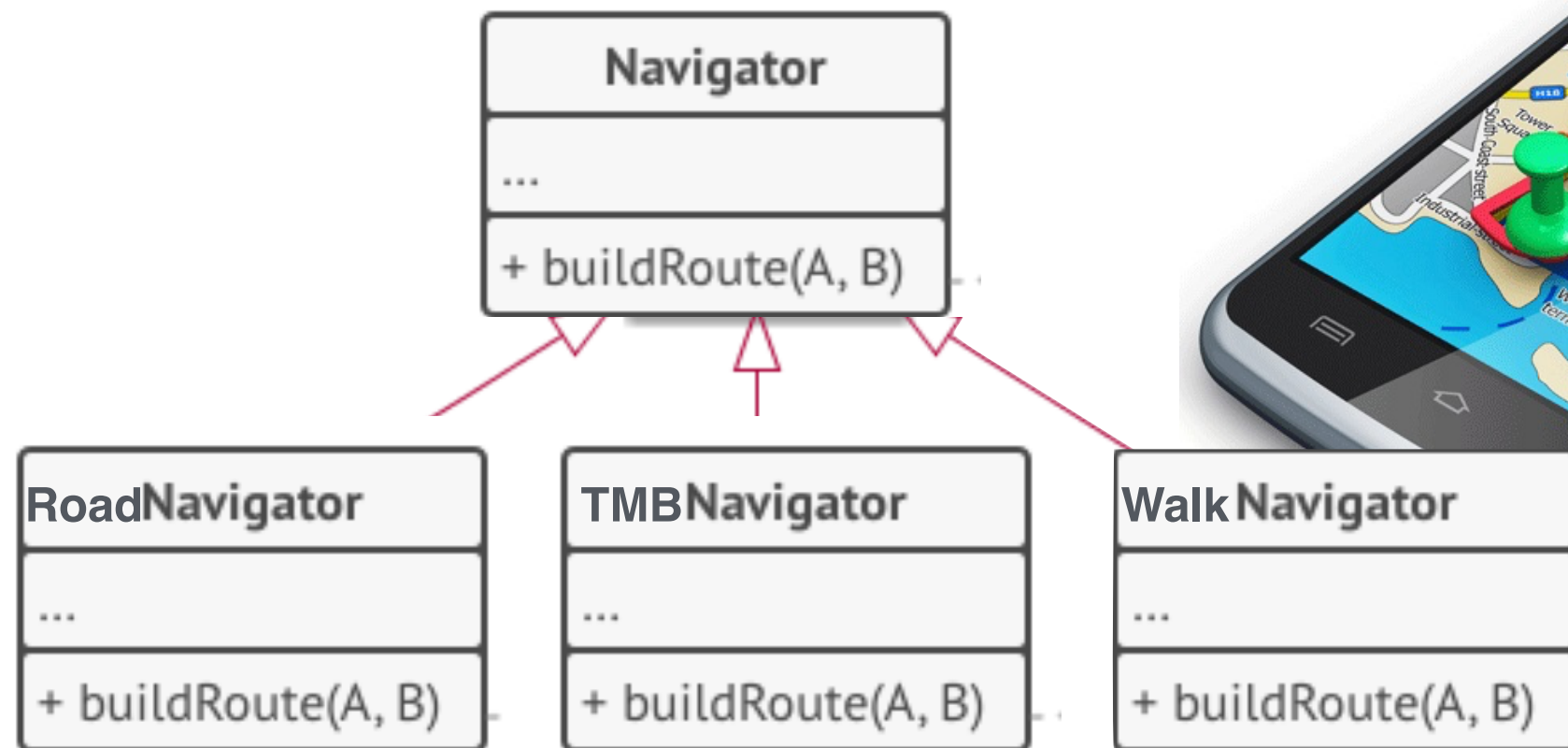
# Patró Strategy: Problema

Es vol fer un algorisme que calculi la ruta entre un punt A i un punt B del mapa



# Patró Strategy: Problema

Es vol fer un algorisme que calculi la ruta entre un punt A i un punt B del mapa



# Patró Strategy

Nom del patró: **Strategy**

Context:

Comportament de diferents lògiques d'un objecte

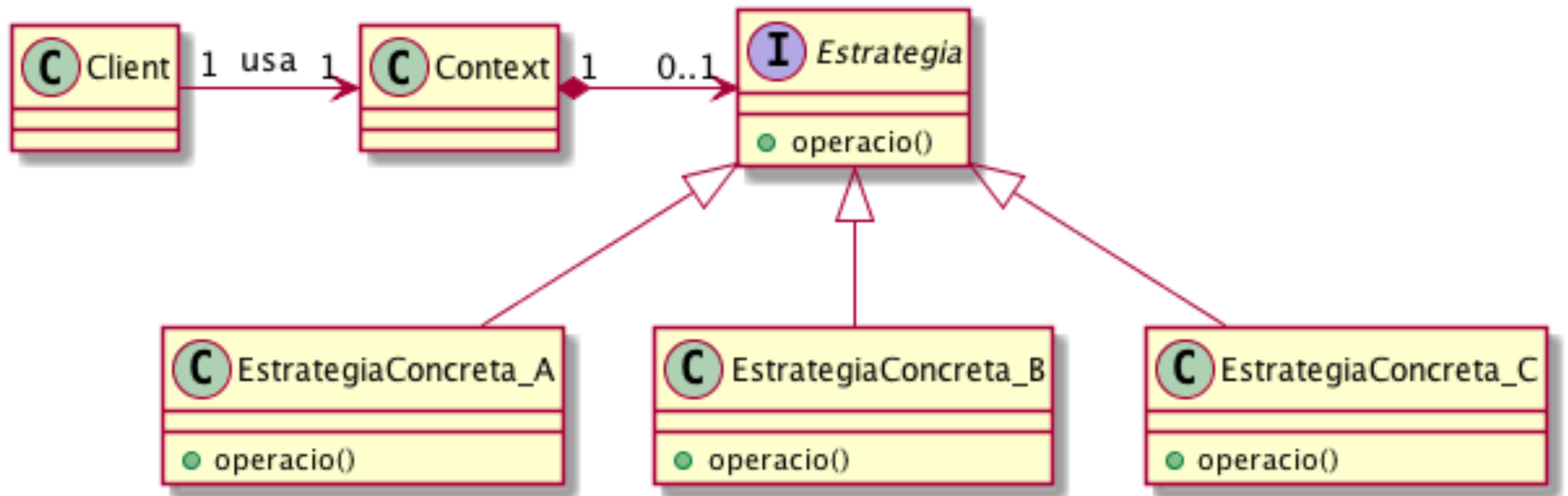
Problema:

- Un mateix problema es pot solucionar amb diferents estratègies o algorismes
- L'elecció d'una de les estratègies pot venir donada pel tipus d'objecte que l'utilitza

Solució:

- S'utilitza una interfície comuna per a poder encapsular tots els tipus d'algorismes
- S'utilitzen herències per modelar els clients que utilitzen les diferents estratègies

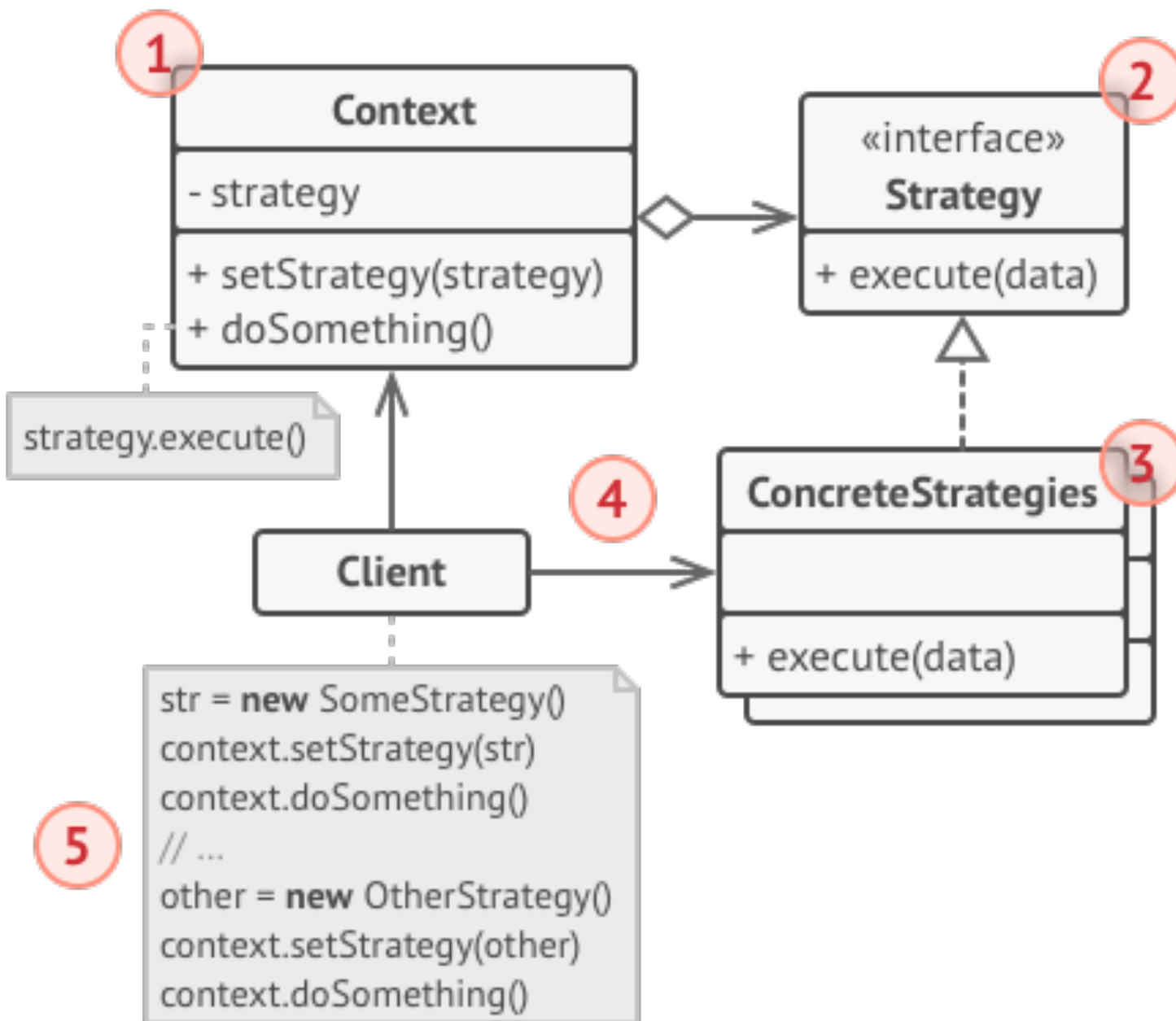
# Patró Strategy



- ❑ La classe **Strategy** declara la interfície comuna a tots els algorismes
- ❑ El **Client** es configura amb un context i una estratègia concreta i manté la referència corresponent
- ❑ El **Client** pot definir una interfície per a que la classe **Strategy** accedeixi a les seves dades (i li passarà a l'estratègia la seva referència) o bé li passa totes les dades necessàries en les operacions



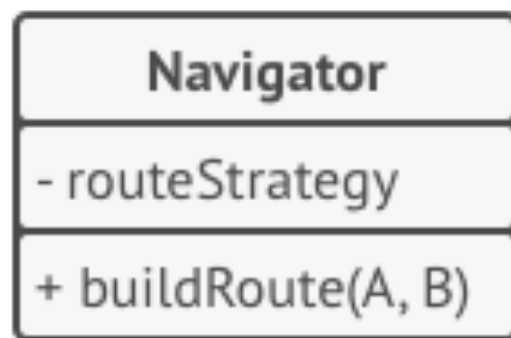
# Patró Strategy



1. El **Context** no coneix res de les estratègies concretes, només les sap executar
2. La classe **Strategy** declara la interfície comuna a tots els algorismes
3. Les implementacions concretes de les estratègies estan a les classes  
**ConcreteStrategies**
4. El **Client** crea l'estratègia concreta que vol fer servir
5. El **Client** passa al **Context** l'estratègia (`setStrategy`) i delega en el **Context** que l'executi.

# Patró Strategy: Solució

Es vol fer un algorisme que calculi la ruta entre un punt A i un punt B del mapa



route = routeStrategy.buildRoute(A, B)



# Patró Strategy

Nom del patró: **Strategy**

## **Consideracions:**

- S'utilitza si es vol modificar l'estratègia a utilitzar en temps d'execució
- S'utilitza quan es tenen classes molt similars que només varien en la forma de comportar-se

## **Pros:**

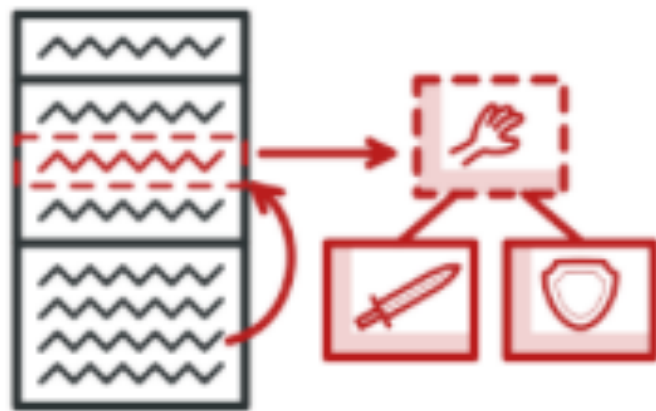
- S'aïllen els detalls de la implementació de la solució
- Es poden usar diferents estratègies i canviar-les en temps d'execució
- Open-Closed Principle ✓

## **Cons:**

- Els Clients han de tenir clar en què es diferencien les diferents estratègies
- En el cas de tenir pocs algorismes que rarament canvien, no cal complicar més el codi usant aquest patró

## 3.5. Patrons de disseny: Patró Strategy

- Patrons relacionats
  - **Template Method**



### Strategy

Diferents mètodes posats cadascun d'ells en una classe diferent



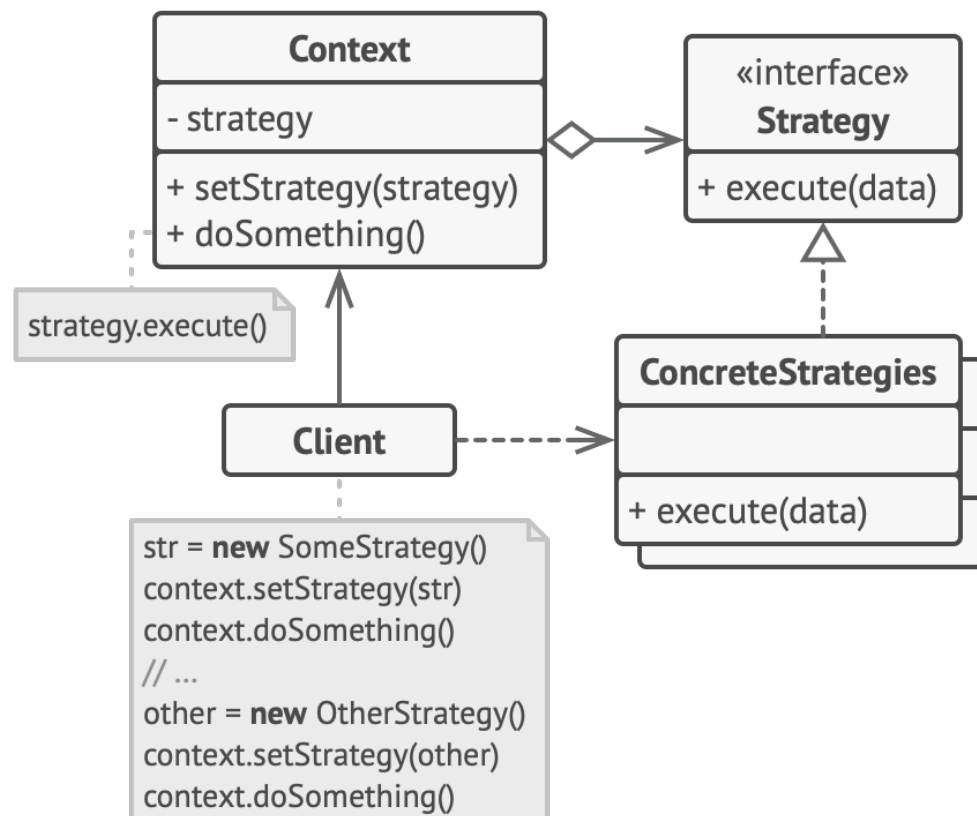
### Template Method

Esquelet d'un algorisme on <sup>12</sup> hi ha alguna part que es canvia a les classes filles



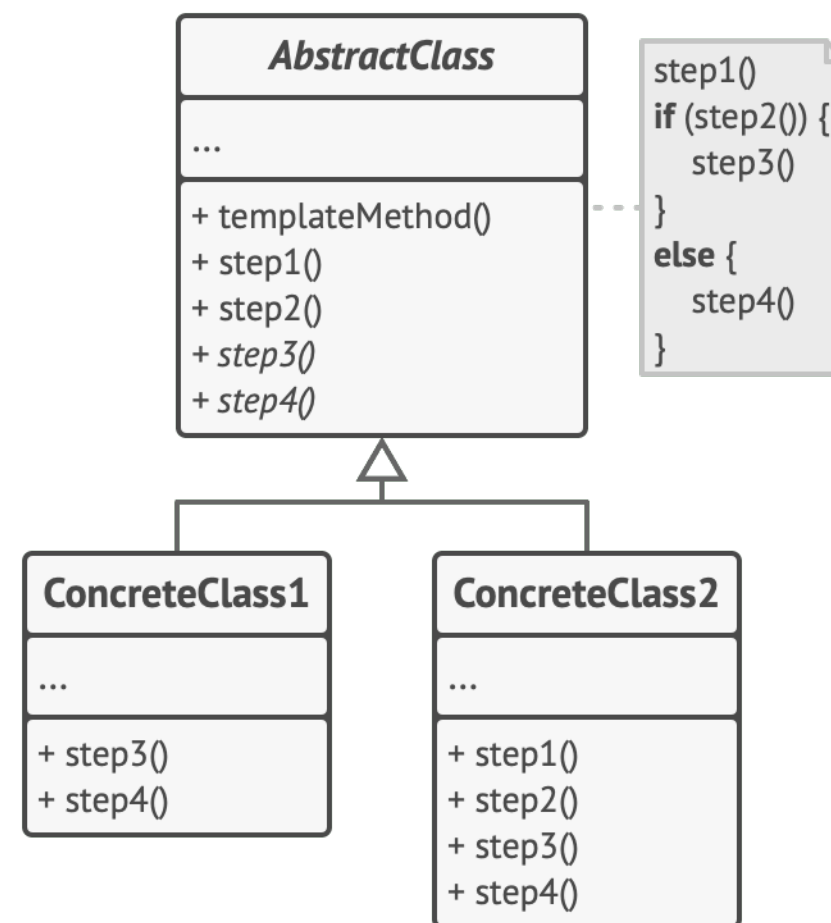
## 3.5. Patrons de disseny: Patró Strategy

- Patrons relacionats



### Strategy

Diferents mètodes posats cadascun d'ells en una classe diferent



### Template Method

Esquelet d'un algorithme on hi ha alguna part que es canvia a les classes filles