DERIVED PROJECT REPORT

ROCKET LAYOUTS

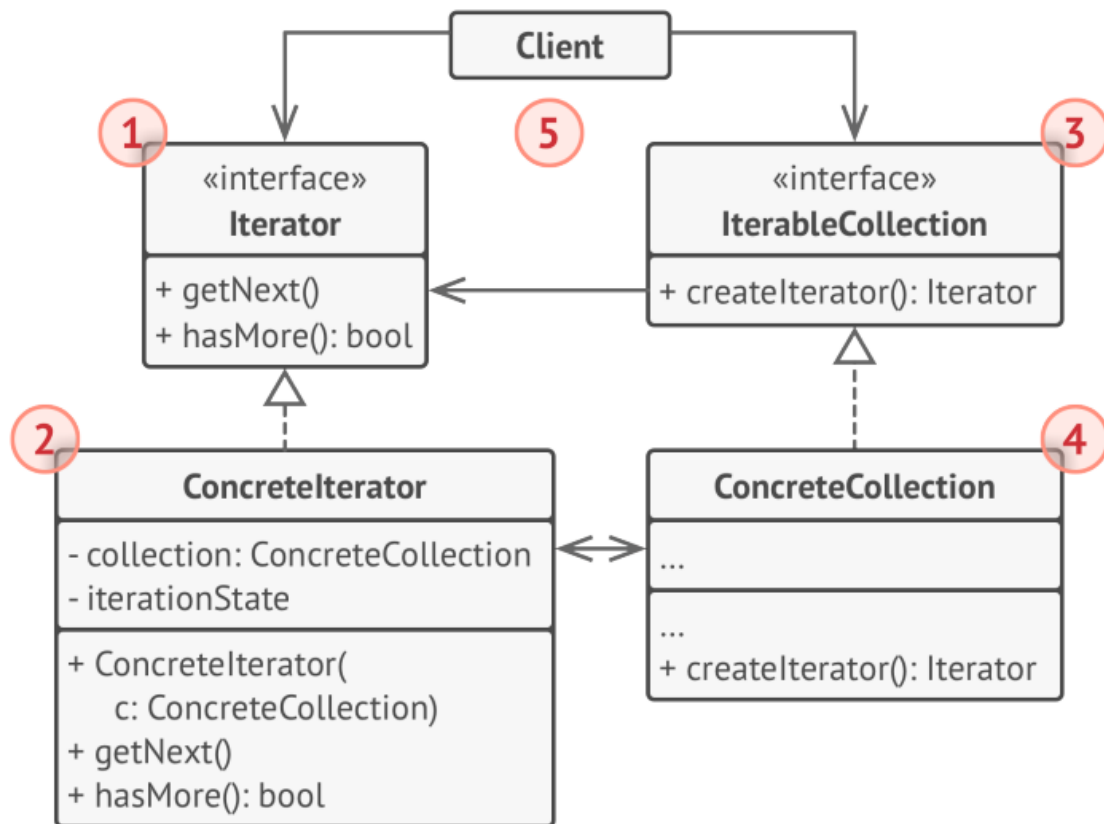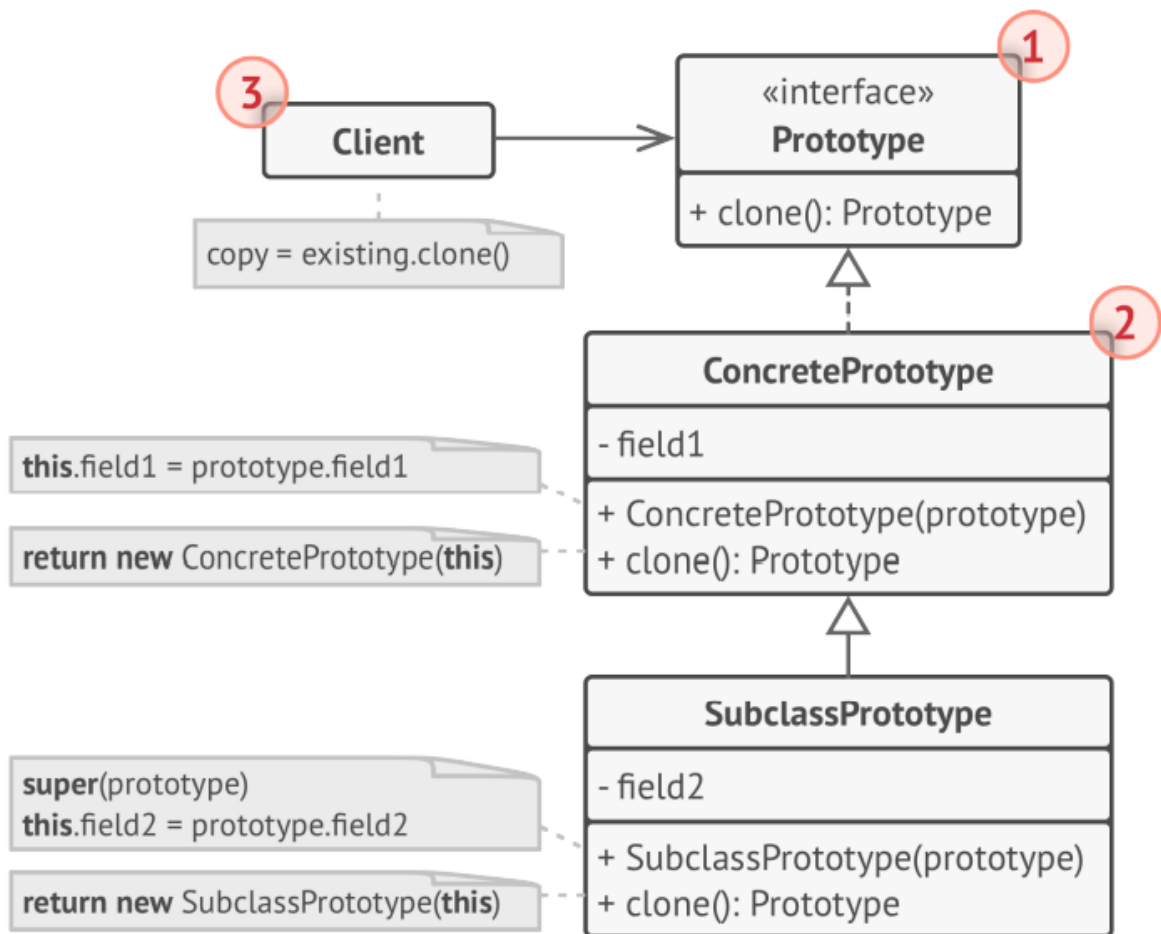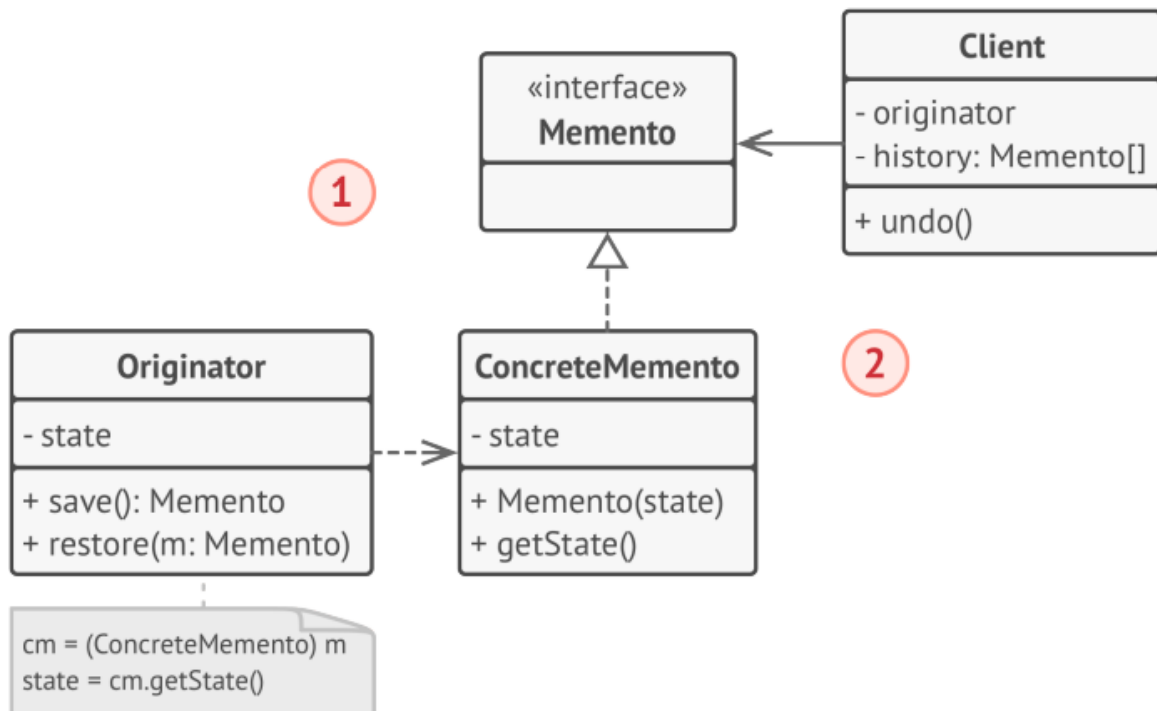| Falcon9		FalconHeavy
Cargo □ 1 x VacuumMerlinEngine	**Payload**	Cargo Humans □ 1 x VacuumMerlinEngine
Falcon9Core 1 x [9 x Merlin Engines]	**Propulsion**	Falcon9Core Falcon9Core Falcon9Core 3 x [9 x Merlin Engines]
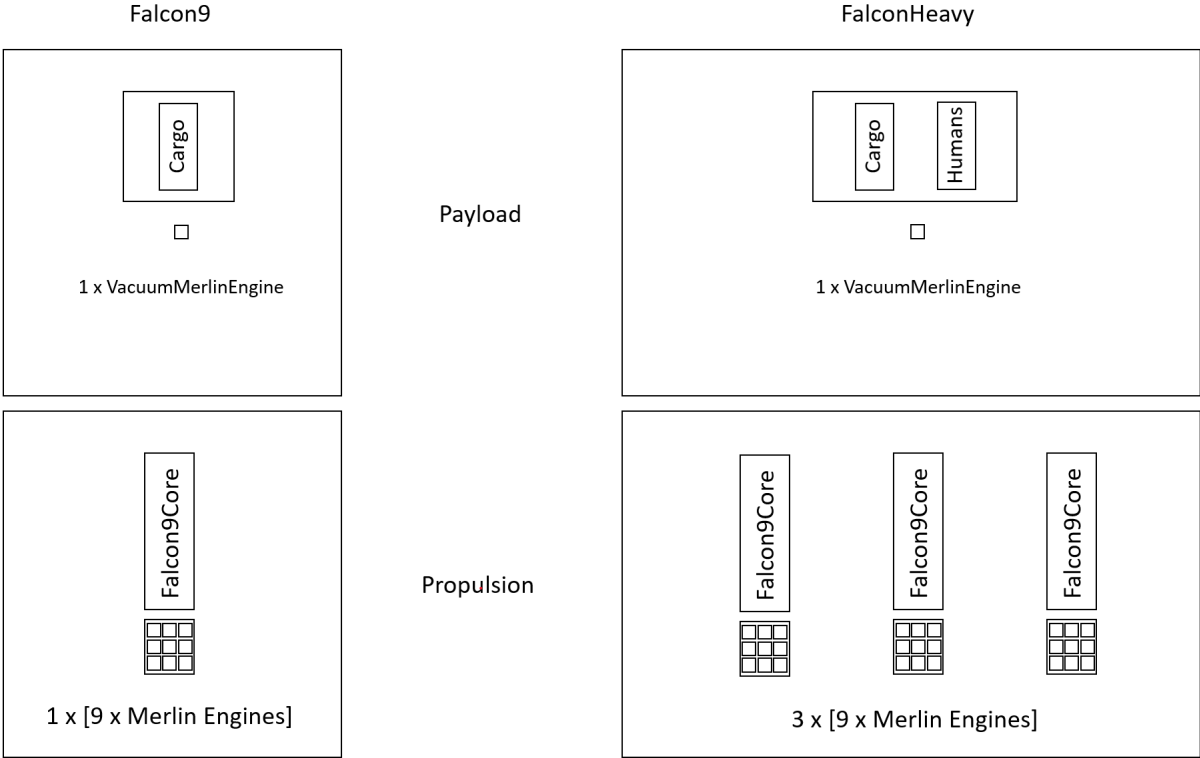
ITERATOR

- There is a Human and Cargo Iterator.
- That was used to traverse through the collections of Humans and Cargo.
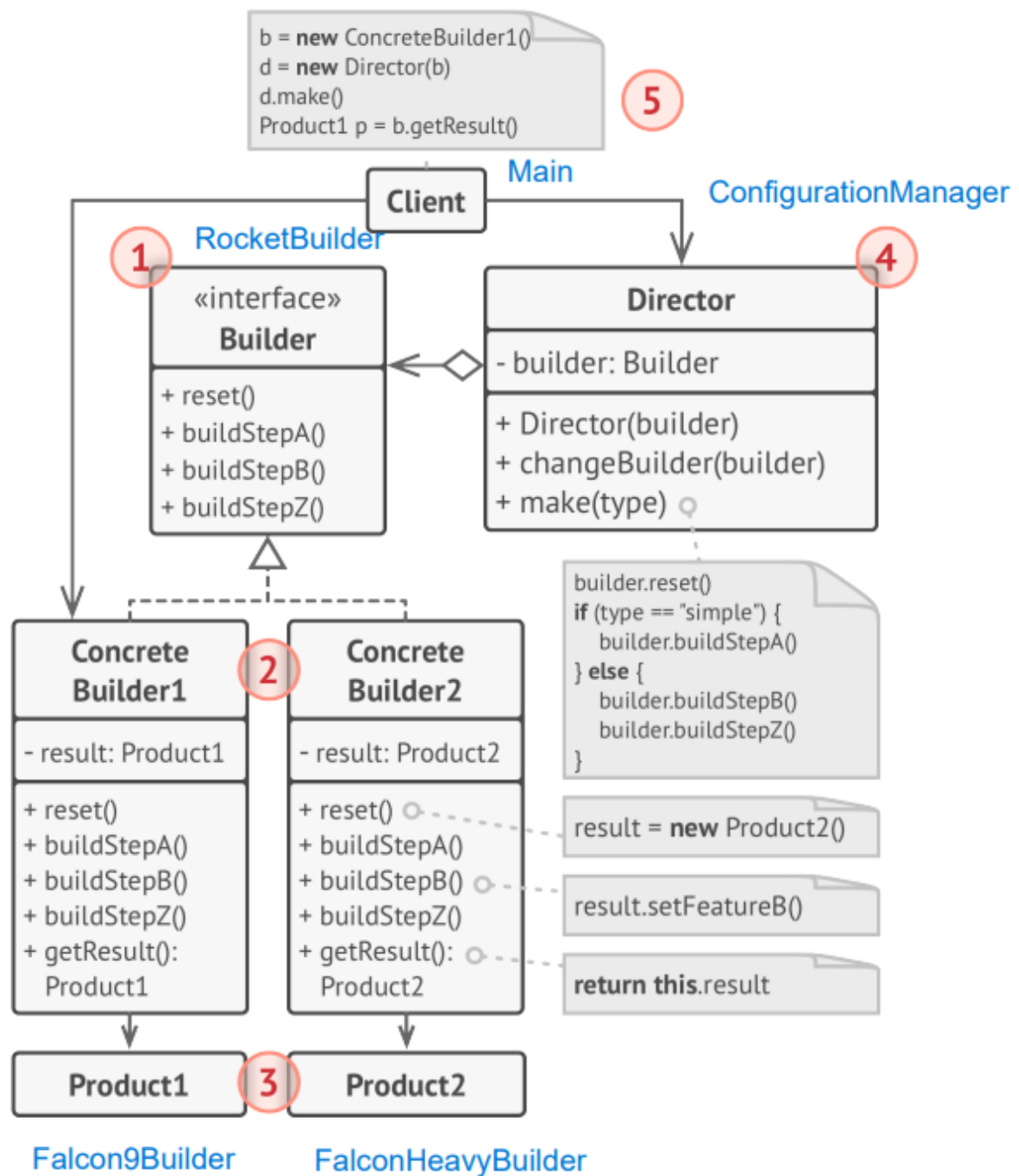
PROTOTYPE

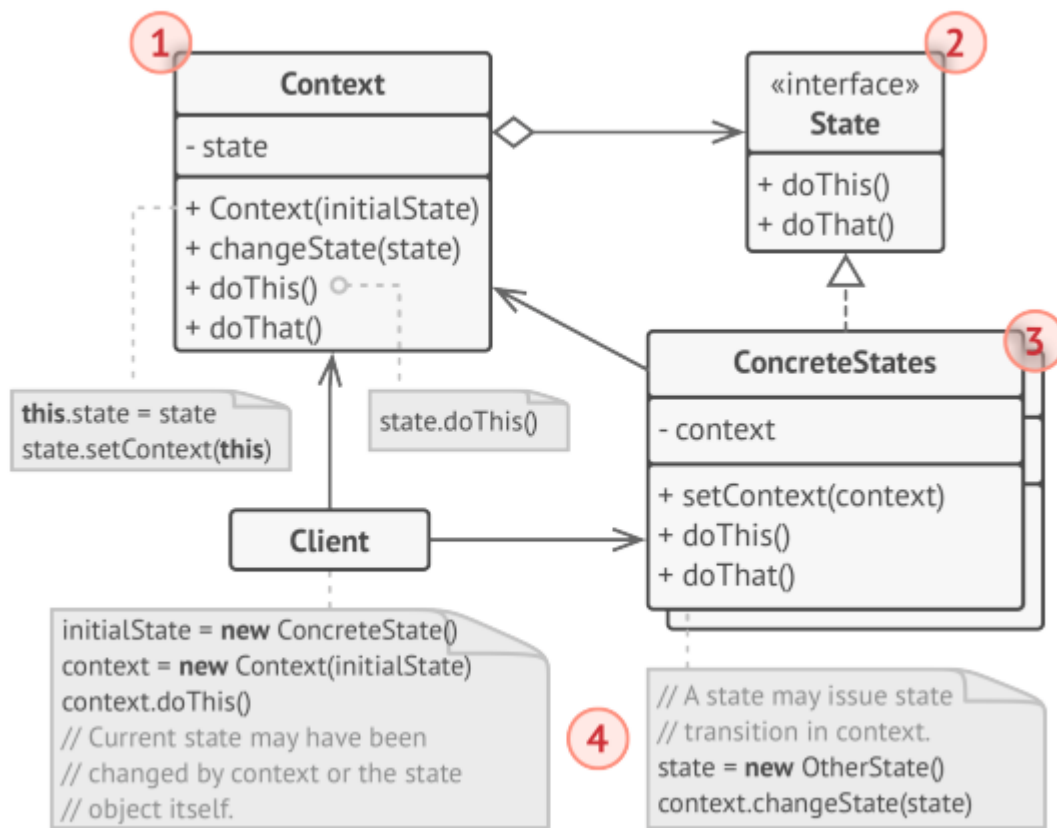- Used to copy Vacuum Merlin Engines.

MEMENTO

- Used to Save the state of the Falcon 9 and the Falcon heavy propulsion system during and before takeoff.

BUILBER


```
b = new ConcreteBuilder1()
d = new Director(b)
d.make()
Product1 p = b.getResult()
```
⑤

Main

Client

RocketBuilder ①

«interface»
Builder

+ reset()
+ buildStepA()
+ buildStepB()
+ buildStepZ()

ConfigurationManager

Director ④

- builder: Builder

+ Director(builder)
+ changeBuilder(builder)
+ make(type)

```
builder.reset()
if (type == "simple") {
    builder.buildStepA()
} else {
    builder.buildStepB()
    builder.buildStepZ()
}
```

Concrete Builder1 ②

- result: Product1

+ reset()
+ buildStepA()
+ buildStepB()
+ buildStepZ()
+ getResult():
  Product1

Concrete Builder2

- result: Product2

+ reset()
+ buildStepA()
+ buildStepB()
+ buildStepZ()
+ getResult():
  Product2

`result = new Product2()`

`result.setFeatureB()`

return this.result

Product1 ③   **Product2**

Falcon9Builder   **FalconHeavyBuilder**

- Used to assemble the `Payload` and `Propulsion` components of a rocket
  - Payload
    - combination of a CargoCollection and/or HumanCollection with a VacuumMerlinEngine
  - Propulsion
    - combination of either 1 or 3 Falcon9Cores with 9 MerlinEngines in each core
- Uses the decorator pattern to decorate the `Payload` component with a `Propulsion` component

STATE

Context (1)
- state
+ Context(initialState)
+ changeState(state)
+ doThis()
+ doThat()

this.state = state
state.setContext(**this**)

state.doThis()

State «interface» (2)
+ doThis()
+ doThat()

ConcreteStates (3)
- context
+ setContext(context)
+ doThis()
+ doThat()

Client

initialState = **new** ConcreteState()
context = **new** Context(initialState)
context.doThis()
// Current state may have been
// changed by context or the state
// object itself.

(4)
// A state may issue state
// transition in context.
state = **new** OtherState()
context.changeState(state)

- **F9STATE**
  1. The first stage has a single Falcon 9 core with 9 Merlin engines to get the second stage and the payload almost in orbit.
  2. The second stage has a single Vacuum Merlin Engine to provide the last kick to get the payload in the desired orbit.
- **FHSTATE**
  1. First stage has three Falcon 9 cores with a total of 27 Merlin engines to get to the second stage and the payload almost in orbit
  2. The second stage has a single Vacuum Merlin Engine to provide the last kick to get the payload in the desired orbit
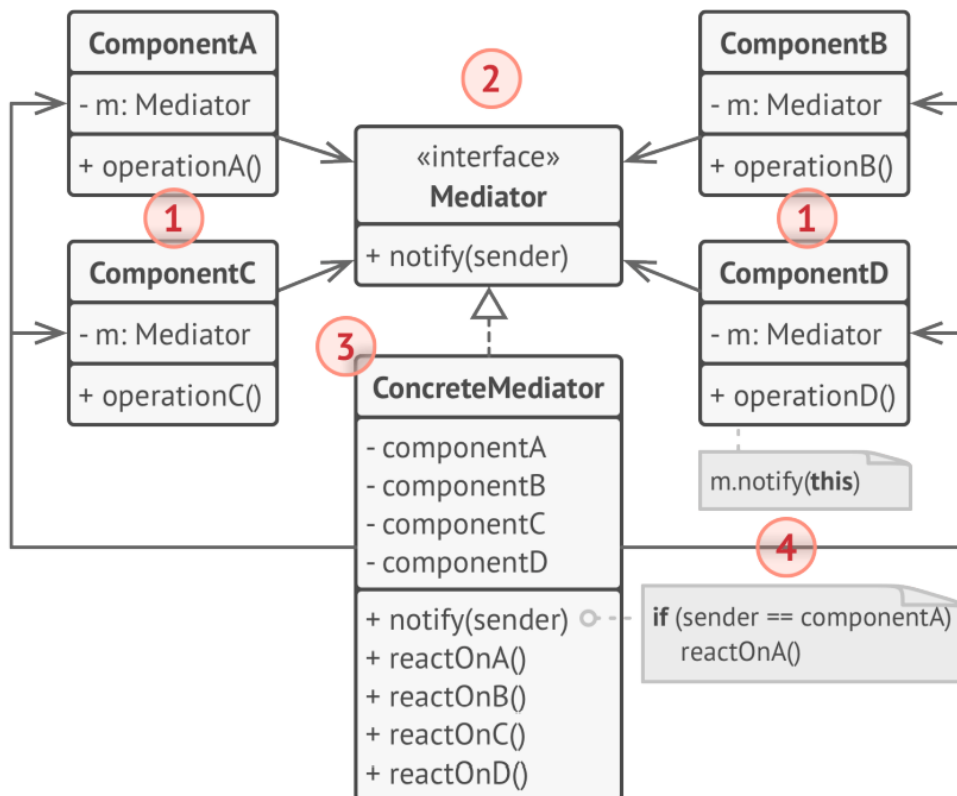
TEMPLATE METHOD
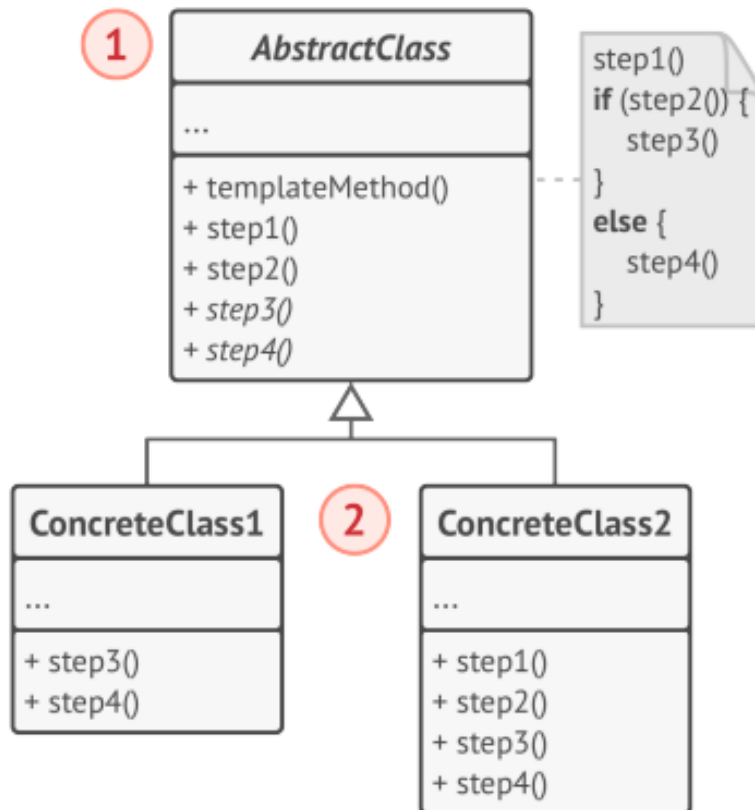
- Create an skeleton for classes , in the context of the project creates a skeleton for
    1. GroundMissionControl
    2. Core
    3. HumanCollection
    4. CargoCollection
    5. HumanIterator
    6. CargoIterator
    7. Satellite
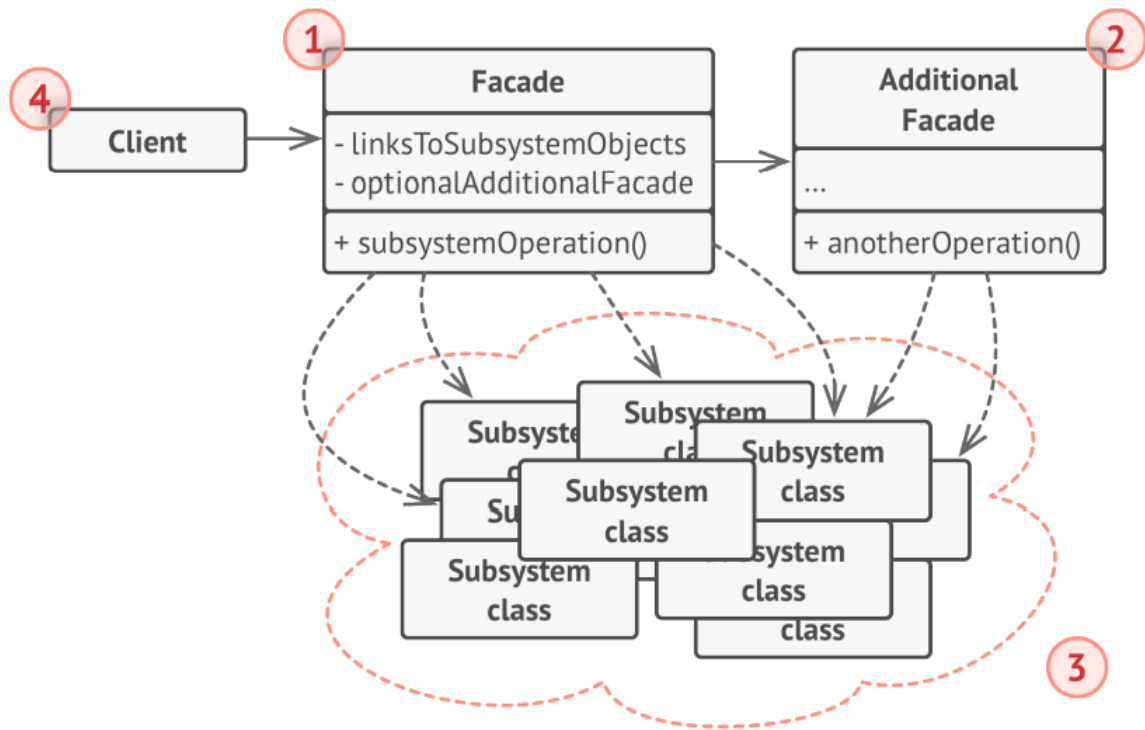   8. SatelliteTransmission
    9. SpaceCraft

OBSERVER
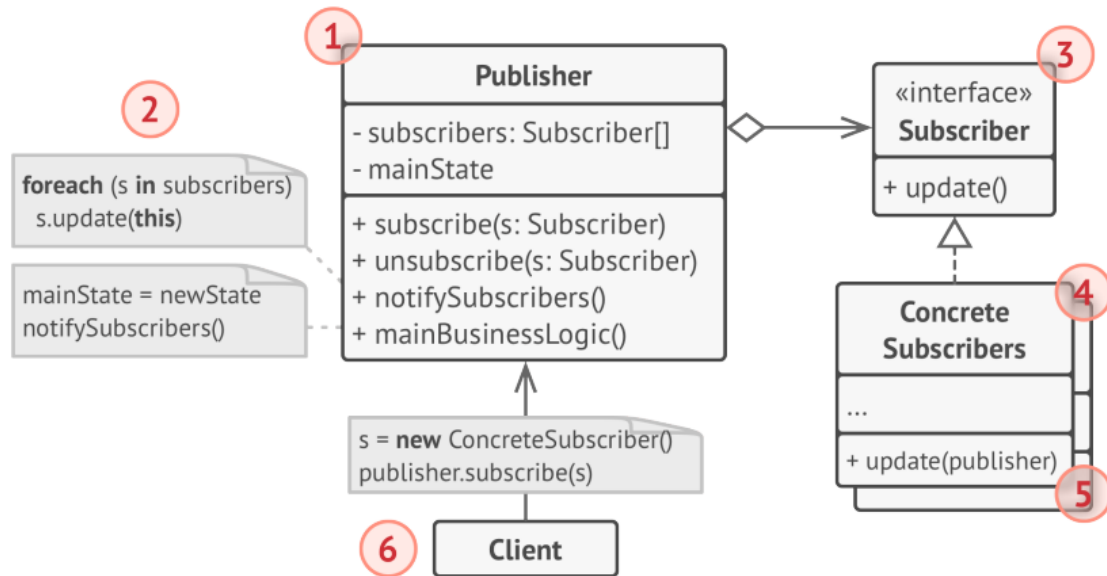
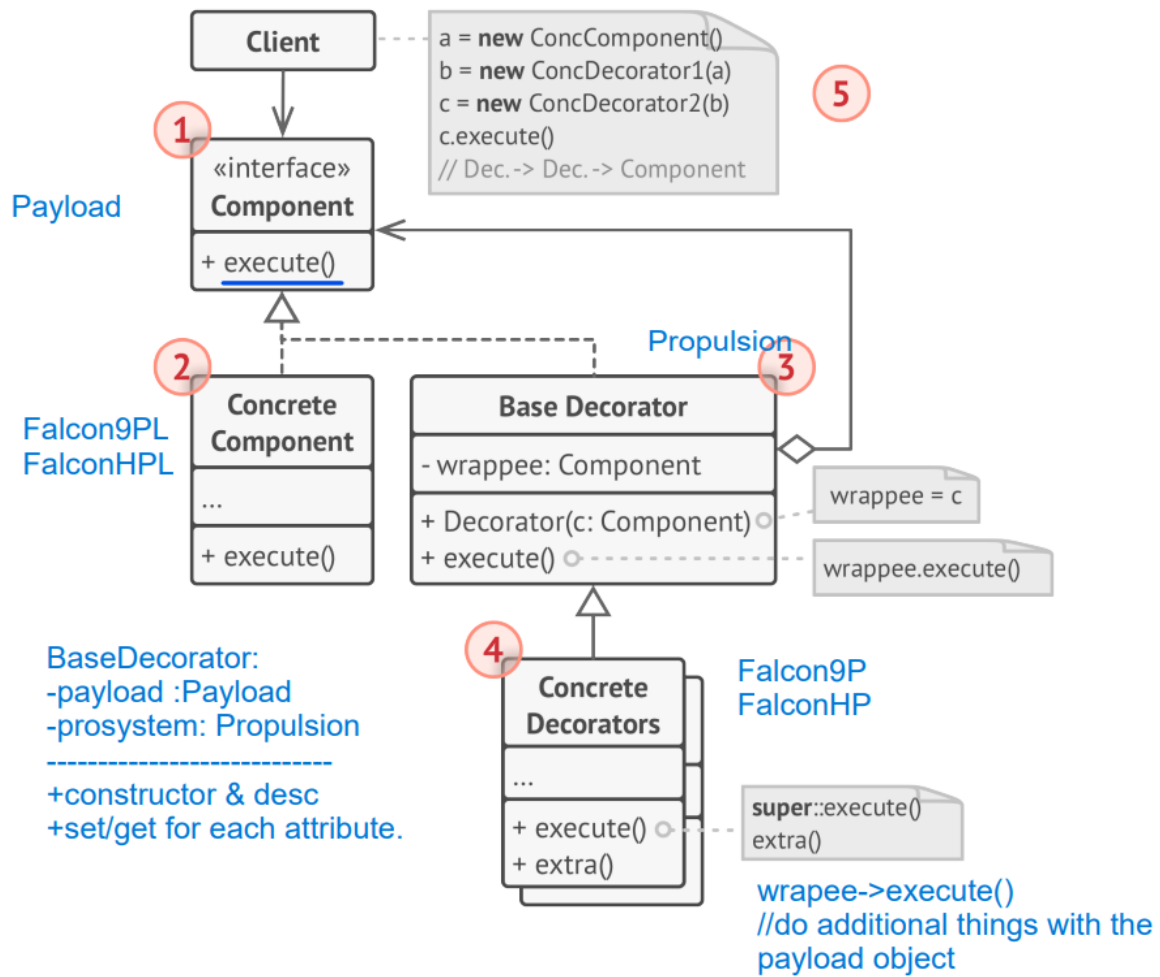- Communication to ground from the StarLink satellites

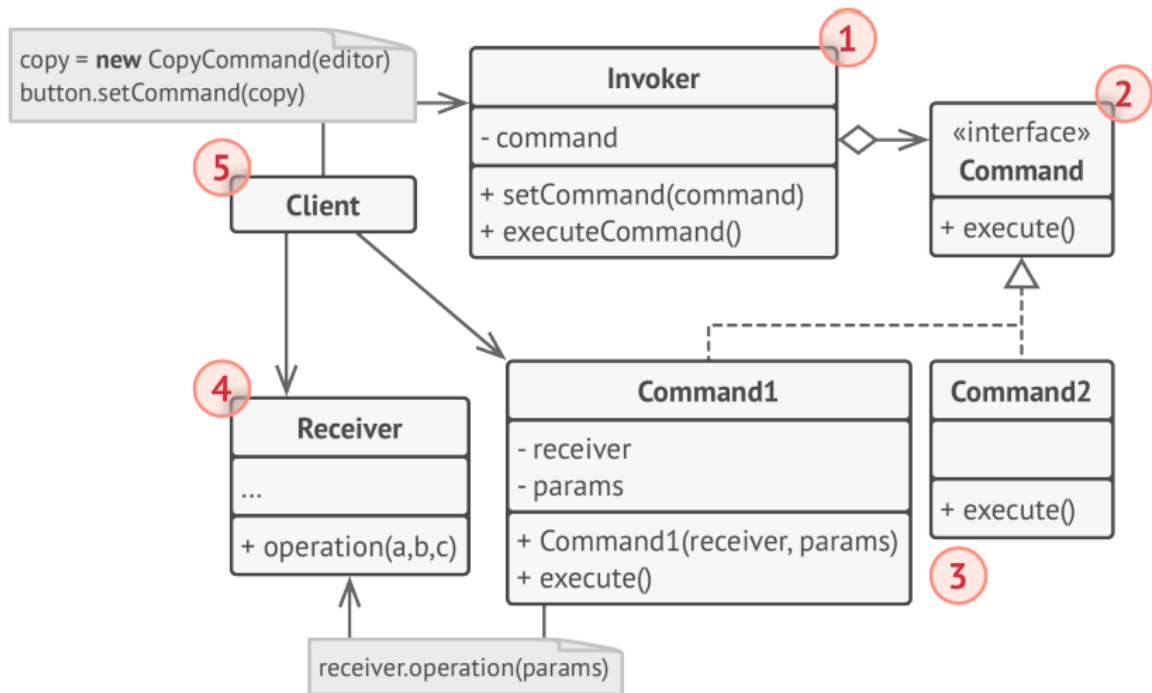# MEDIATOR

- Communication between StarLink satellites

FACADE

- Create an Interface for the use to interact with pre built functions to simulate certain scenarios
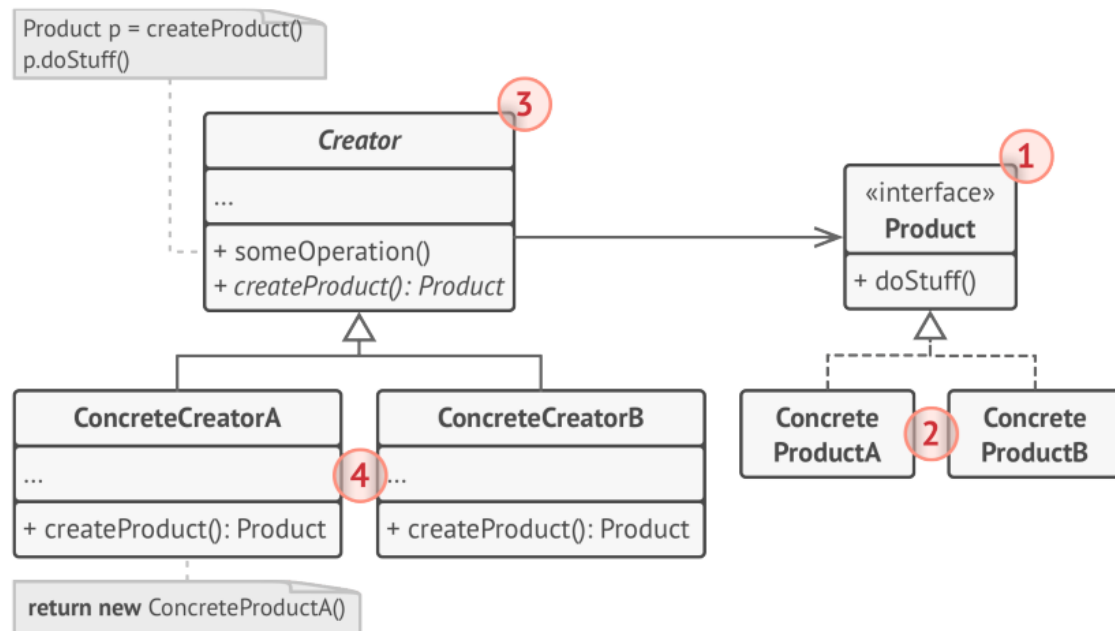
DECORATOR

Client

a = **new** ConcComponent()
b = **new** ConcDecorator1(a)
c = **new** ConcDecorator2(b)
c.execute()
// Dec. -> Dec. -> Component

⑤

①

Payload

«interface»
Component

+ execute()

②

Falcon9PL
FalconHPL

**Concrete
Component**

...

+ execute()

Propulsion

③

Base Decorator

- wrappee: Component

+ Decorator(c: Component)
+ execute()

wrappee = c

wrappee.execute()

BaseDecorator:
-payload :Payload
-prosystem: Propulsion

+constructor & desc
+set/get for each attribute.

④

**Concrete
Decorators**

...

+ execute()
+ extra()

Falcon9P
FalconHP

super::execute()
extra()

wrapee->execute()
//do additional things with the
payload object

- Decorates a Payload object with a Propulsion object to extend the functionality

COMMAND


```
copy = new CopyCommand(editor)
button.setCommand(copy)
```

Invoker (1)
- command

+ setCommand(command)
+ executeCommand()

«interface» **Command** (2)

+ execute()

Client (5)

Receiver (4)

...

+ operation(a,b,c)

Command1
- receiver
- params

+ Command1(receiver, params)
+ execute() (3)

Command2

+ execute()

```
receiver.operation(params)
```

- Enables the functionality to launch a Falcon9 or a FalconHeavy Rocket

FACTORY METHOD

VALGRIND OUTPUT

LINK

https://docs.google.com/document/d/1v-f4AZf23z0smsvgDYD6nxY3iSlJjrSBPuFzSnbEhcA/edit?usp=sharing