TEMA 3
DISSENY DE PATRONS

1. Patrons de creació.

- Abstract Factory
- Builder
- Factory Method
- Object Pool
- Prototype
- Singleton.

2. Patrons structurals.

- Adapter
- Bridge
- Composite
- Decorator
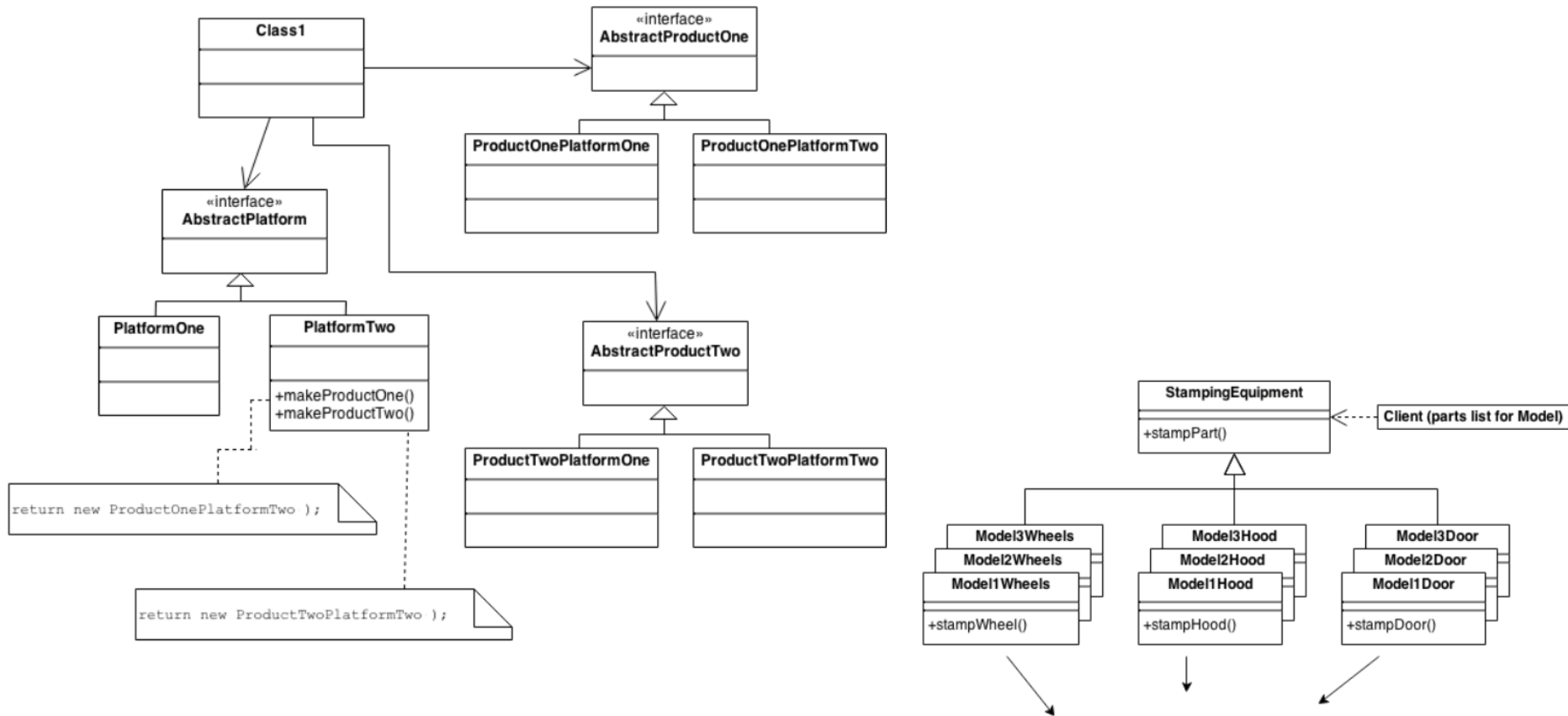- Facade
- Flyweight
- Private Class Data
- Proxy

3. Patrons Comportament.

- Chain of responsibility
- Command
- Interpreter
- Iterator
- Mediator
- Memento
- Null Object
- Observer
- State
- Strategy
- Template method
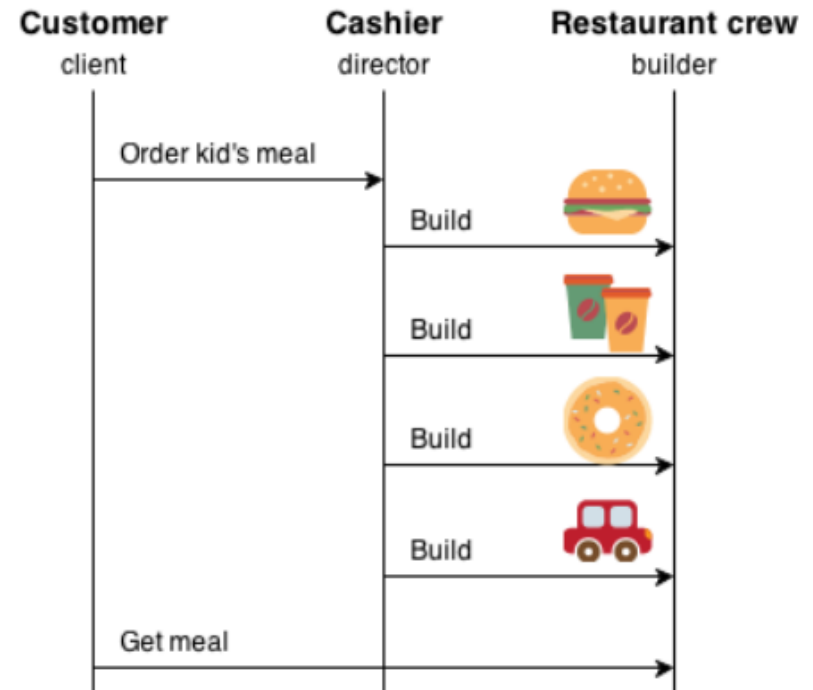- Visitor
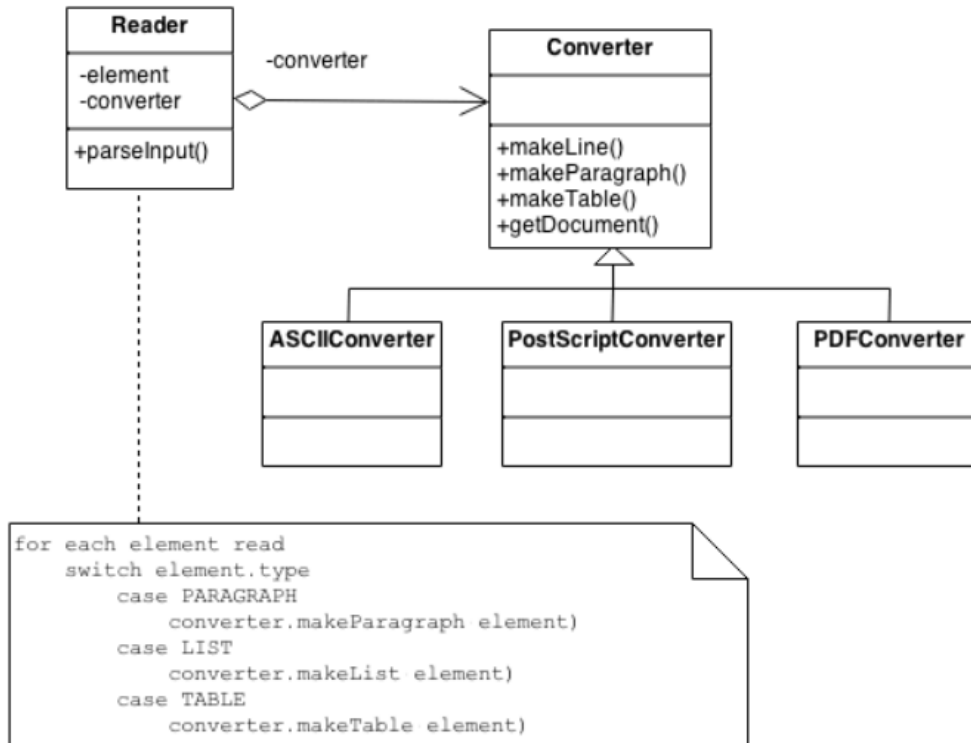
Patrons de disseny
Creació

Creació – Abstract Factory
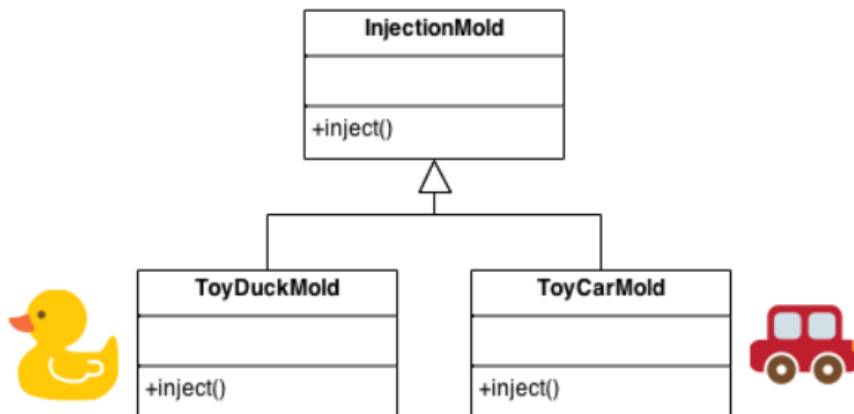

- Creates an instance of several families of classes

Source: https://sourcemaking.com/design_patterns/abstract_factory

Creació – Builder


```
for each element read
    switch element.type
        case PARAGRAPH
            converter.makeParagraph element)
        case LIST
            converter.makeList element)
        case TABLE
            converter.makeTable element)
```

- Separates object construction from its representation

Source: https://sourcemaking.com/design_patterns/abstract_factory

Creació – Factory Method

«interface»
Framework

+*makeProduct() : Product*

ApplicationOne

+makeProduct() : Product

ApplicationTwo

+makeProduct() : Product

```
return new ProductOne );
```

Product

ProductTwo   **ProductOne**

InjectionMold

+inject()

ToyDuckMold

+inject()

ToyCarMold

+inject()
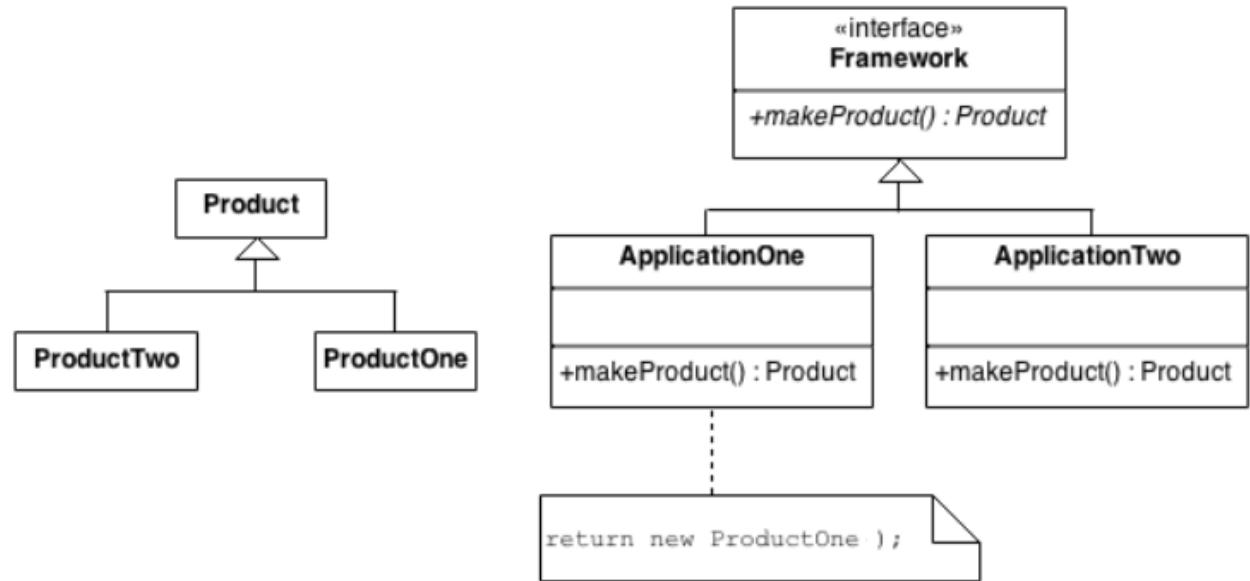
- Creates an instance of several derived classes

5

Source: https://sourcemaking.com/design_patterns/abstract_factory

Creació – Object Pool

ReusablePool.getInstance().acquareReusable()

Client

ReusablePool
-reusables
+static getInstance() : ReusablePool +acquireReusable() : Reusable +releaseReusable(in a : Reusable) +setMaxPoolSize(in size)

Create work space
for newly hired employee.

No   Office equipment   Yes
is in stock

Order new
equipment.

Take equipment
from warehouse.

Store
(new object creation)

Deliver office equipment
to a work space.

If employee is fired,
return all equipment
to warehouse.

Warehouse
(Object Pool)

Work space
(context)

- Avoid expensive
  acquisition and release
  of resources by recycling
  objects that are no longer
  in use

6

clonerRegistry is populated
each Image derived class
registering an instance of itself

ImageHandler

+populateImages()

-cloneRegistry

-images

Image

+clone() : Image

images[next] =
clonerRegistry.get lookupKey).clone);

ImageOne

+clone() : ImageOne

ImageTwo

return new ImageOne this);

Cell

+split()

SingleCellOrganism

+split()
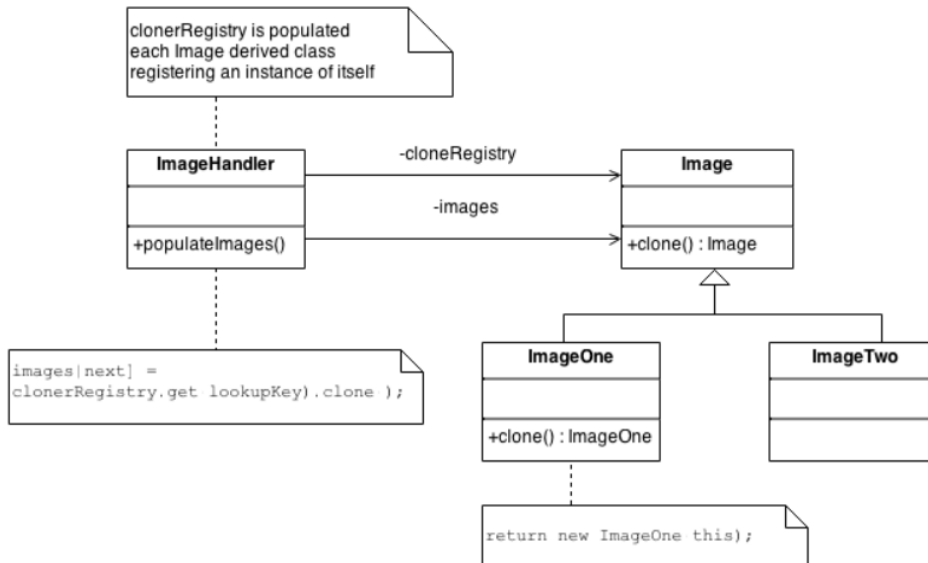
Clones

- A fully initialized instance to be copied or cloned

7

Source: https://sourcemaking.com/design_patterns/abstract_factory
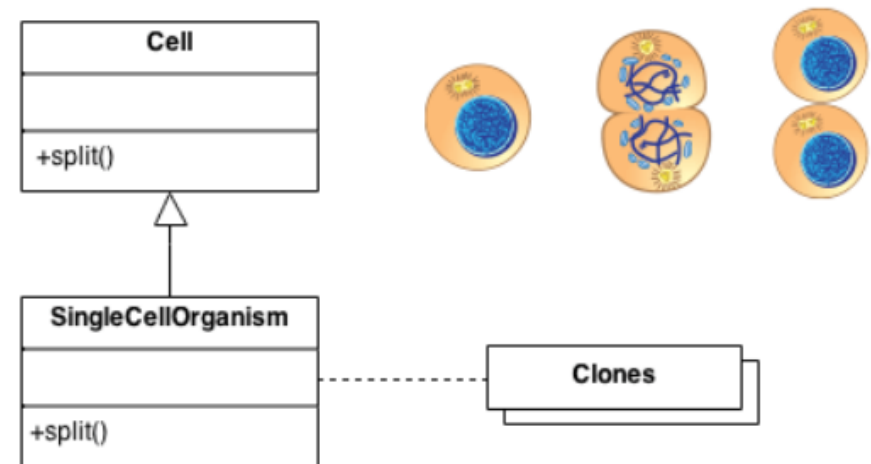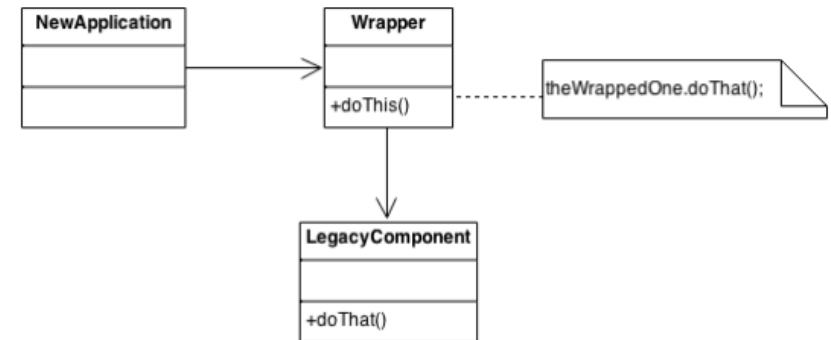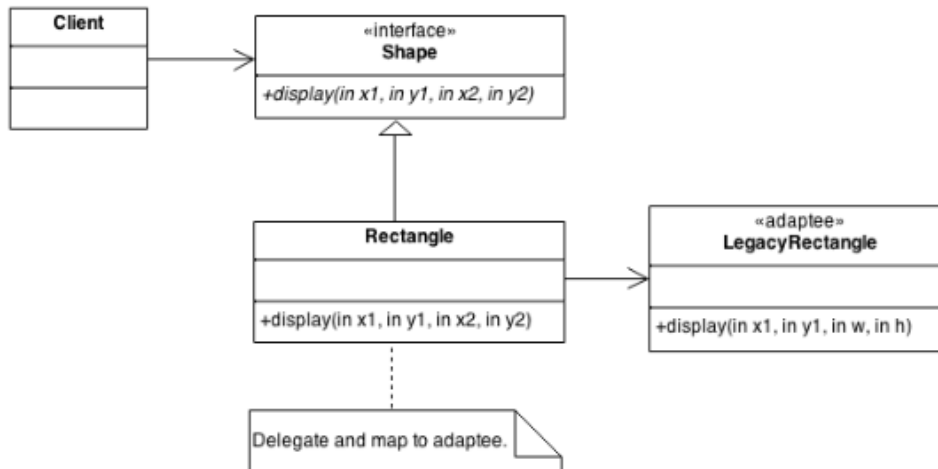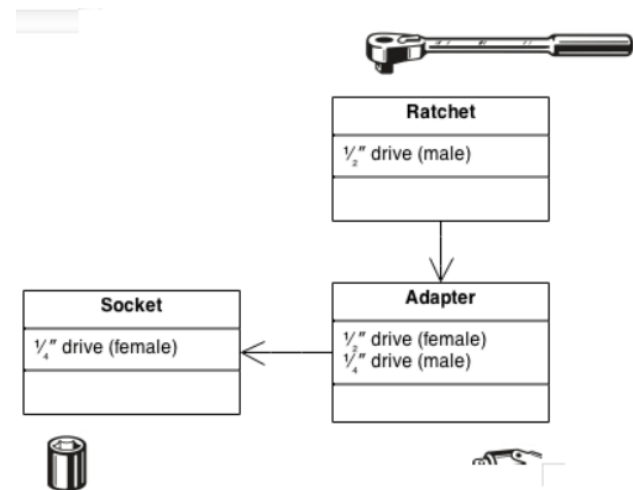
Creació – Singleton
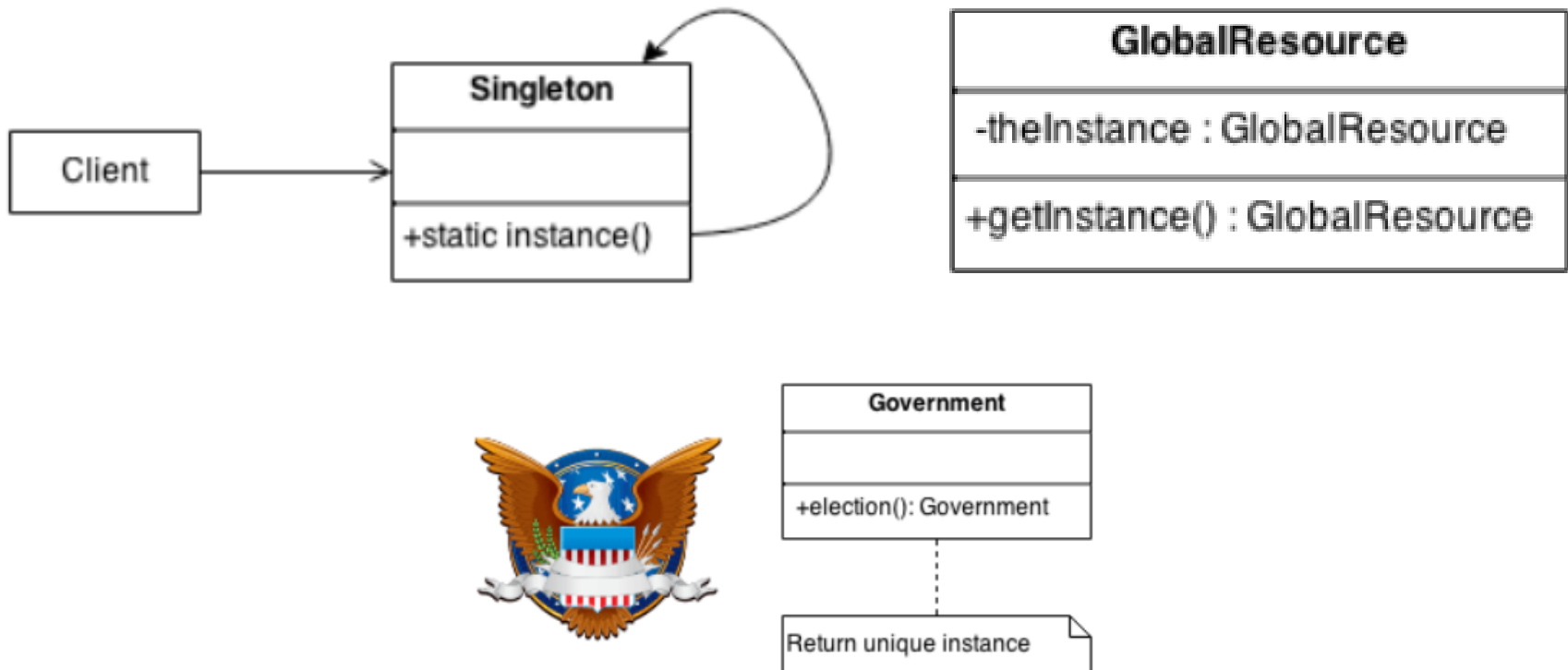
- A class of which only a single instance can exist

Patrons de disseny Structurals

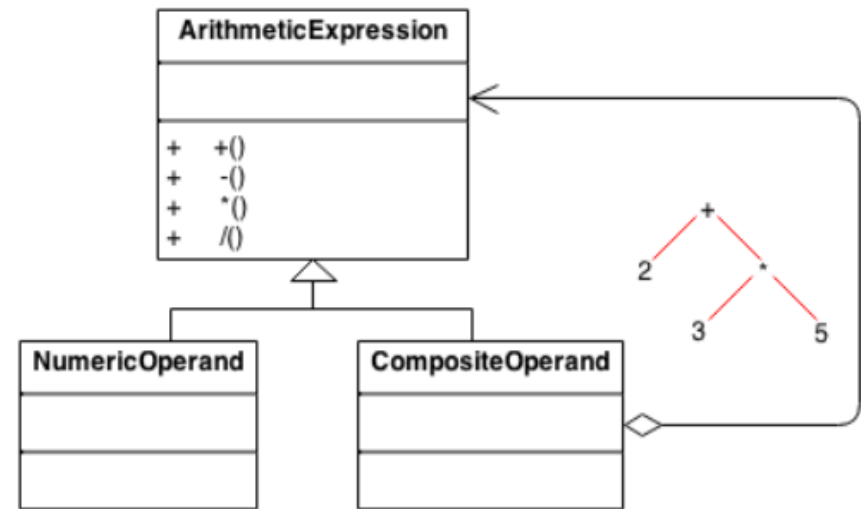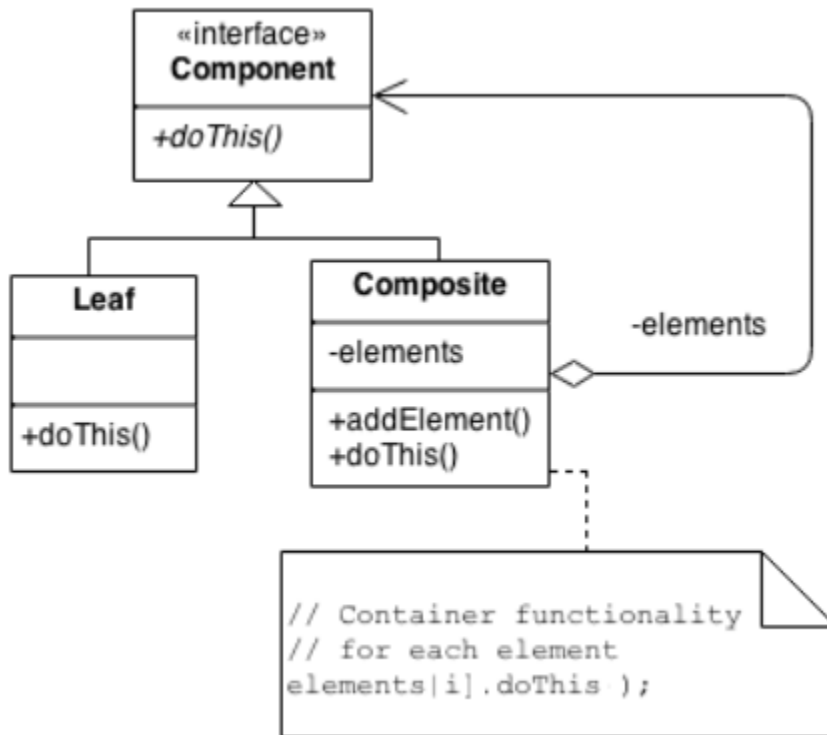# Structural – Adapter

- Match interfaces of different classes

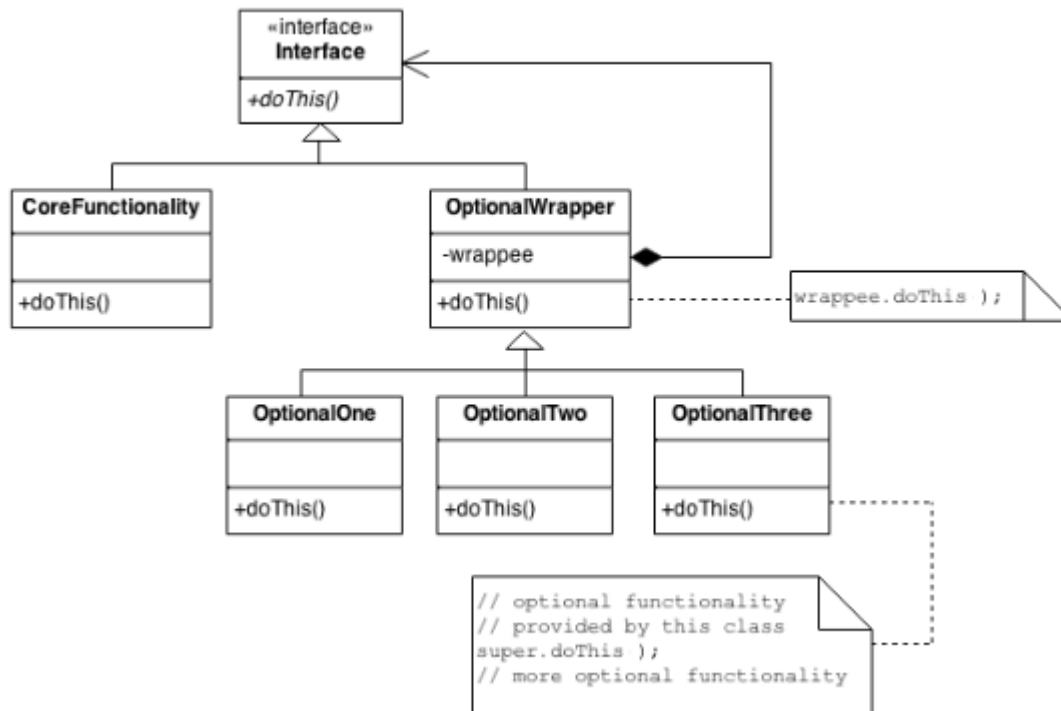Source: https://sourcemaking.com/design_patterns/abstract_factory

Structural – Bridge

ThreadScheduler

PreemptiveThreadScheduler   TimeSlicedThreadScheduler

UnixPTS   WindowsPTS   UnixTSTS   WindowsTSTS

ThreadShceduler

PreemptiveThreadScheduler   TimeSlicedThreadScheduler

UnixPTS   WindowsPTS   UnixTSTS   WindowsTSTS

JVM_PTS   JVM_TSTS

ThreadScheduler

PreemptiveThreadScheduler

TimeSlicedThreadScheduler

ThreadScheduler_Implementation

UnixPTS   WindowsPTS

JVM_PTS

Bridge

Switch
+ON()
+OFF()

SwitchImplementation
+ON()
+OFF()

- Separates an object's interface from its implementation

Source: https://sourcemaking.com/design_patterns/abstract_factory
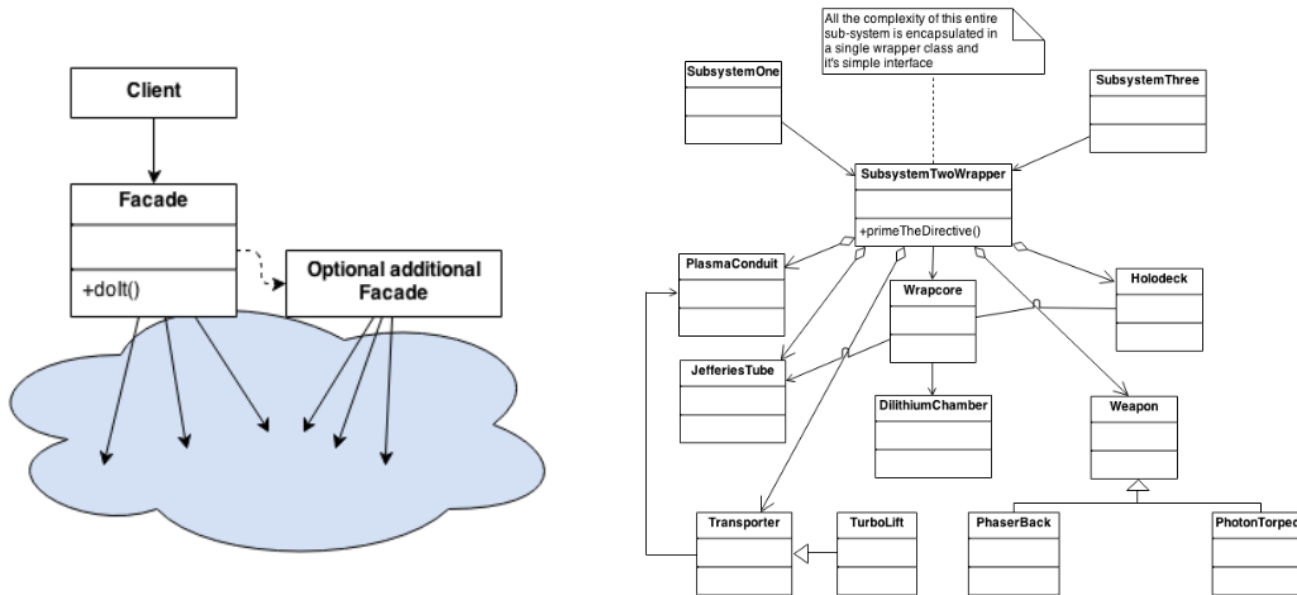
Structural – Composite

- A tree structure of simple and composite objects

Structural – Decorator
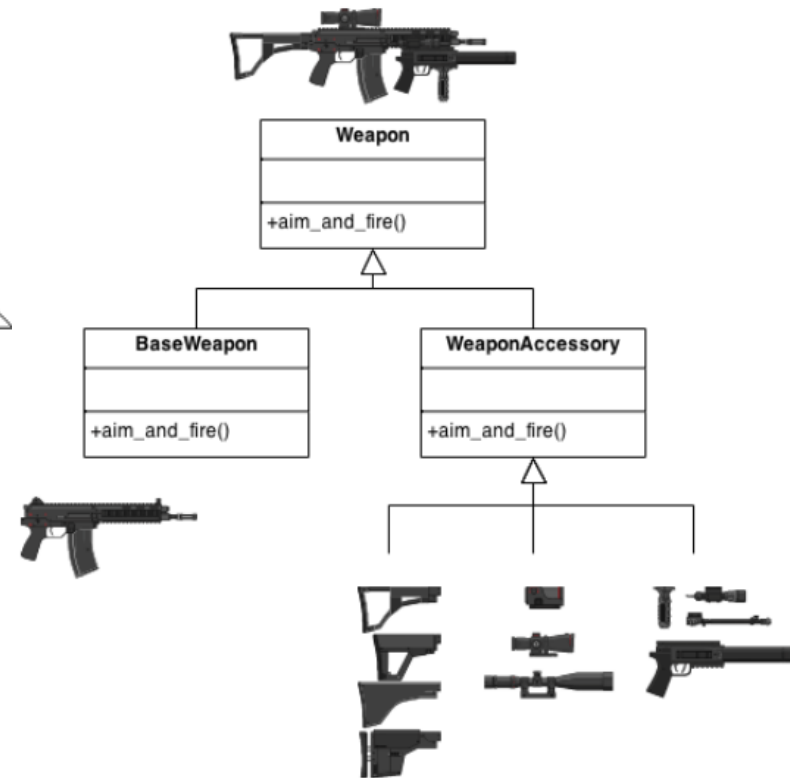
- Add responsibilities to objects dynamically
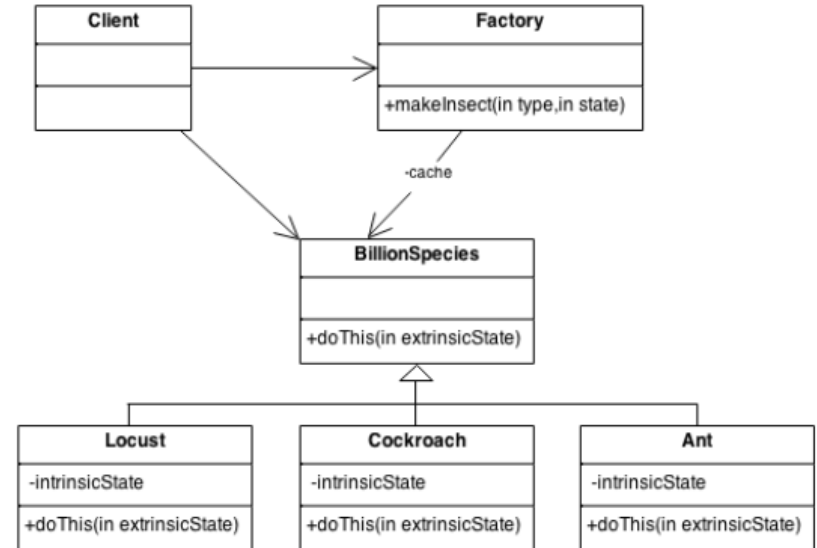
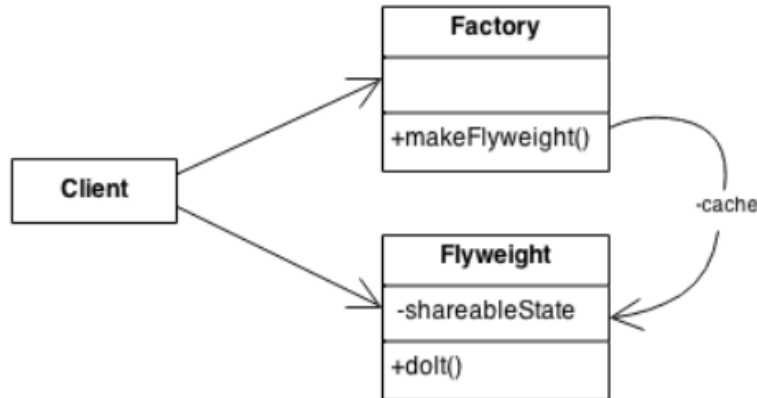Source: https://sourcemaking.com/design_patterns/abstract_factory

- A single class that represents an entire subsystem

Customer service Facade

Source: https://sourcemaking.com/design_patterns/abstract_factory

Structural – Flyweight

Factory
+makeFlyweight()

Client

Flyweight
-shareableState
+doIt()

-cache

Client

Factory
+makeInsect(in type,in state)

-cache

BillionSpecies
+doThis(in extrinsicState)

Locust
-intrinsicState
+doThis(in extrinsicState)

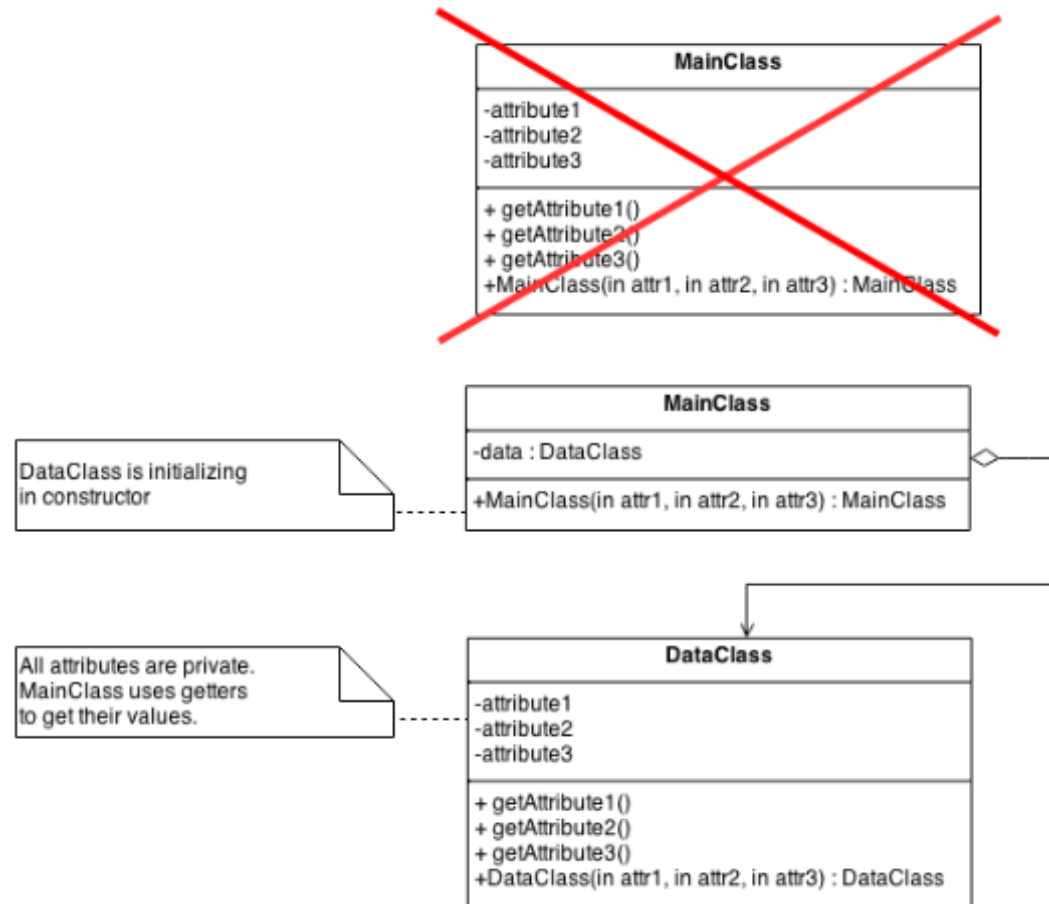| Cockroach |
|---|
| -intrinsicState |
| +doThis(in extrinsicState) |

Ant
-intrinsicState
+doThis(in extrinsicState)

Browser loads images
just once and then
reuses them from pool:

- A fine-grained instance used for efficient sharing

Source: https://sourcemaking.com/design_patterns/abstract_factory

Structural – Private Class Data

- Restricts accessor/mutator access

Source: https://sourcemaking.com/design_patterns/abstract_factory

Structural – Proxy
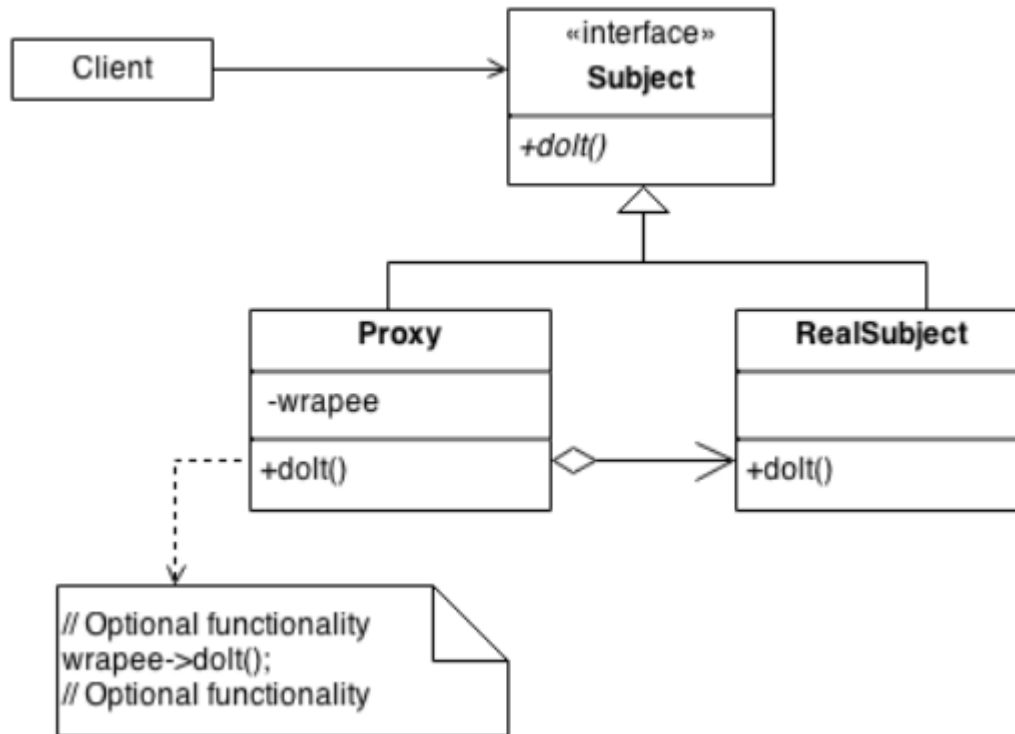
- An object representing another object

Source: https://sourcemaking.com/design_patterns/abstract_factory
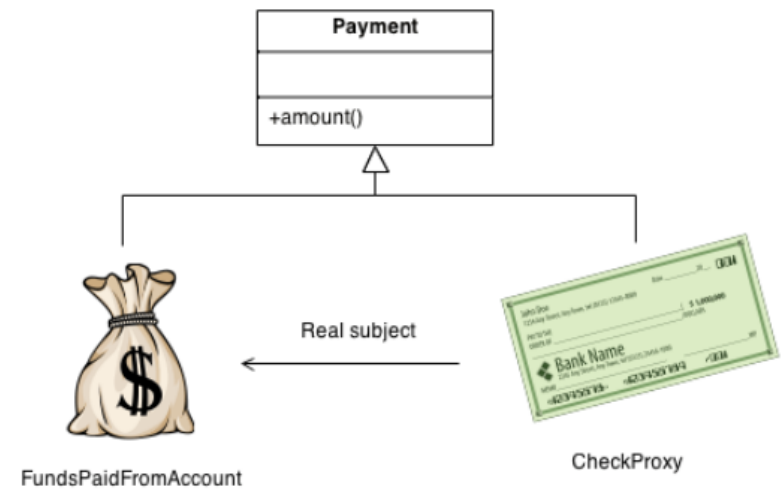
Patrons de disseny Comportament

Comportament – Chain of responsibility


```
Handler
-super
-nextHandler
+handle()        ----- nextHandler.handle );
```

```
HandlerOne      HandlerTwo
                              If  I can handle request)
                                // handle it
                              else
                                super.handle );
```

- A way of passing a request between a chain of objects

19

Comportament – Chain of responsibility


```
Handler
-super
-nextHandler
+handle()
```
nextHandler.handle);

```
HandlerOne
```

```
HandlerTwo
```
```
If  I can handle request)
  // handle it
else
  super.handle );
```

- A way of passing a request between a chain of objects

Source: https://sourcemaking.com/design_patterns/abstract_factory

Comportament – Command


```
Client
```

```
«interface»
CallbackInterface

+execute()
```

```
Receiver

+doThis()
+doThat()
```

```
targetObject = receiverObject;
targetMethod = methodPointer;
```

```
CallbackOne
```

-targetObject

```
CallbackTwo

-targetMethod

+constructor(receiverObject, methodPointer)
+execute()
```

```
// use Java reflection or
// C++ or Delphi pointer
// to member function
targetObject.targetMethod );
```

```
CallbackInterface token =
new CallbackTwo newReceiver ), "doThis");
// the token object is passed to another
// object and that object calls
token.execute );</p>
```

Customer	Waiter	Order	Cook
client	director	command	receiver

Order()

PlaceOrder()

Order
Chicken salad
Pasta
Ice tea
Cheese cake
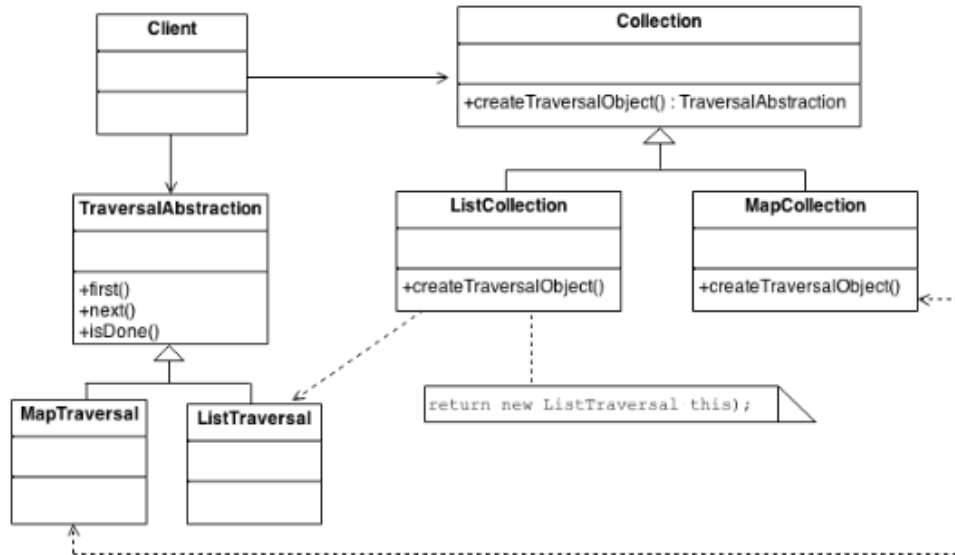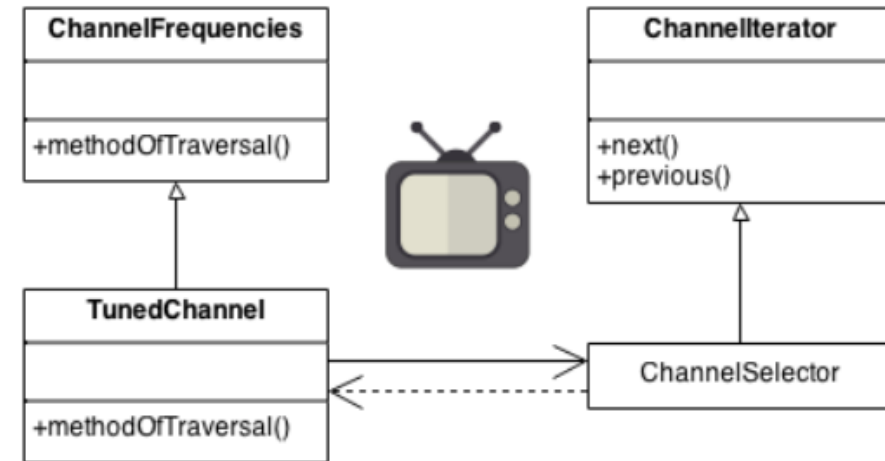
Cook()

- Encapsulate a command request as an object

21
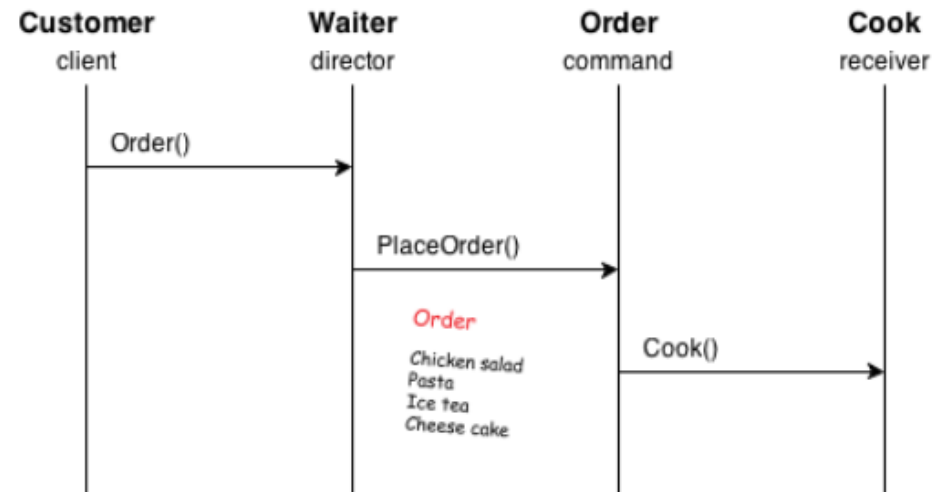
Comportament – Iterator

- Sequentially access the elements of a collection

Source: https://sourcemaking.com/design_patterns/abstract_factory

Comportament – Observer
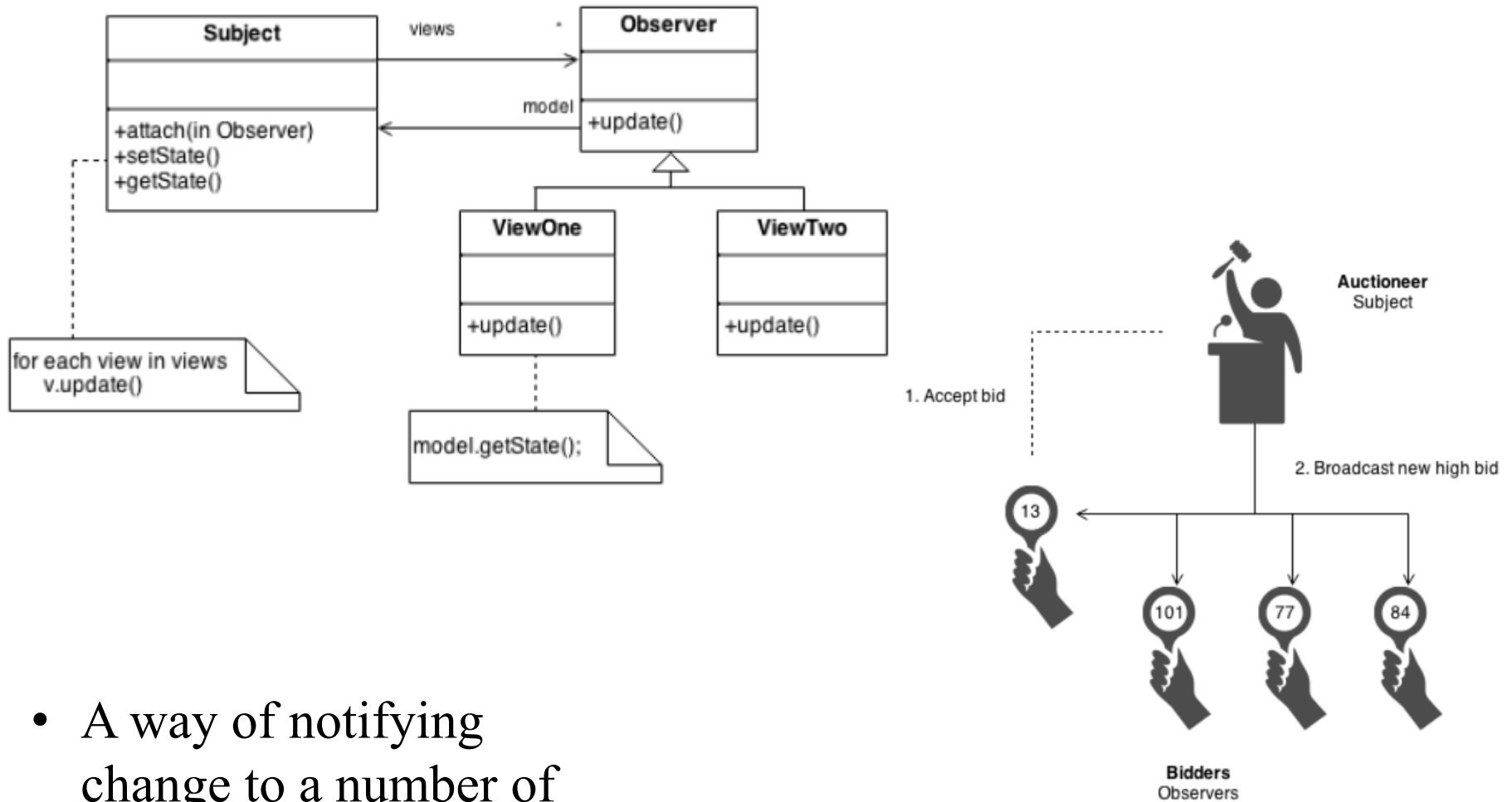
- A way of notifying change to a number of classes

Source: https://sourcemaking.com/design_patterns/abstract_factory