



10 Pines

Diseño a la Gorra - Episodio 04



Hernán Wilkinson



hernan.wilkinson@10pines.com



@HernanWilkinson

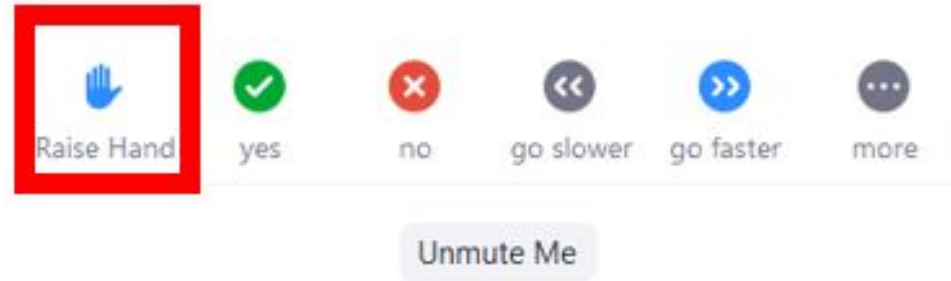


<https://alagorra.10pines.com>





Estar muteados a menos que sea necesario



No voy a poder leer el chat



La comunicación visual es importante. Usarla a discreción





Diseño ¡a la gorra!

¡Bienvenidos!

Durante esta serie de Webinars exploraremos *qué* significa **Diseñar Software con Objetos** y *cómo* lo podemos hacer cada vez mejor.

Trataremos muchos temas que irán desde cuestiones filosóficas como qué significa Diseñar en nuestra profesión y dónde está expresado ese Diseño, pasando por consejos y heurísticas para diseñar "mejor" y terminado con ejemplos concretos de cómo aplicar esas heurísticas en la *vida real*.

Los webinars son "*language agnostic*", o sea que no dependen de un lenguaje de programación en particular, aunque los ejemplos que usaremos estarán hechos principalmente en **Java**, **JavaScript**, **Ruby**, **Python** y mi querido **Smalltalk** cuando amerite 😊.

Te esperamos todos los Martes a las 19 Hrs GMT-3 a partir del Martes 11 de Agosto de 2020. Para poder participar tenes que registrarte [acá](#).

Todo el código y presentaciones estarán disponibles para que lo puedan usar y consultar en cualquier momento [acá](#).

¡Trae ganas de aprender y pasarla bien!

¿Por qué a la Gorra?

Al igual que cuando Diseñamos Software está bueno usar una **Metáfora** para entender qué estamos modelando, en este caso usamos una metáfora para explicar cómo *financiaremos*

Donaciones



\$100 - Casi una 🍷

Pagar

\$250 - Una buena 🍺

Pagar

\$500 - Menos que 🍔 + 🍷

Pagar



Donate



¿Querés donar un monto variado o en otra plataforma?



Dictado por:
Hernán Wilkinson

<https://alagorra.10pines.com>

¡En breve estaremos
anunciando nuevos cursos!



¿Qué vimos en el último episodio?



Metáfora





ModelCreator →
Asistente para Completar Formulario
(**FormCompletionAssistant**)

CompositeModelCreator →
Asistente para Completar Sección
(**FormSectionCompletionAssistant**)



H4: Usar **metáforas** para “*mapear*” el problema

Tip: No usar nombres de patrones en las clases
como *CompositeModelCreator*



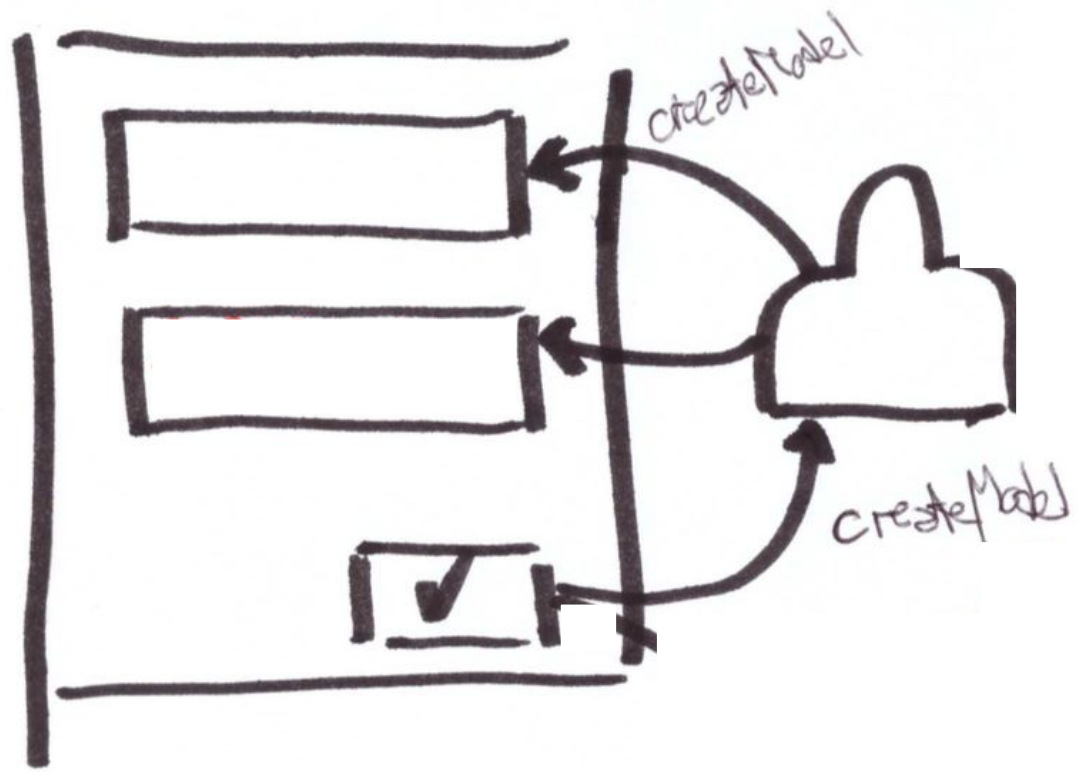
Los Objetos no viven aislados

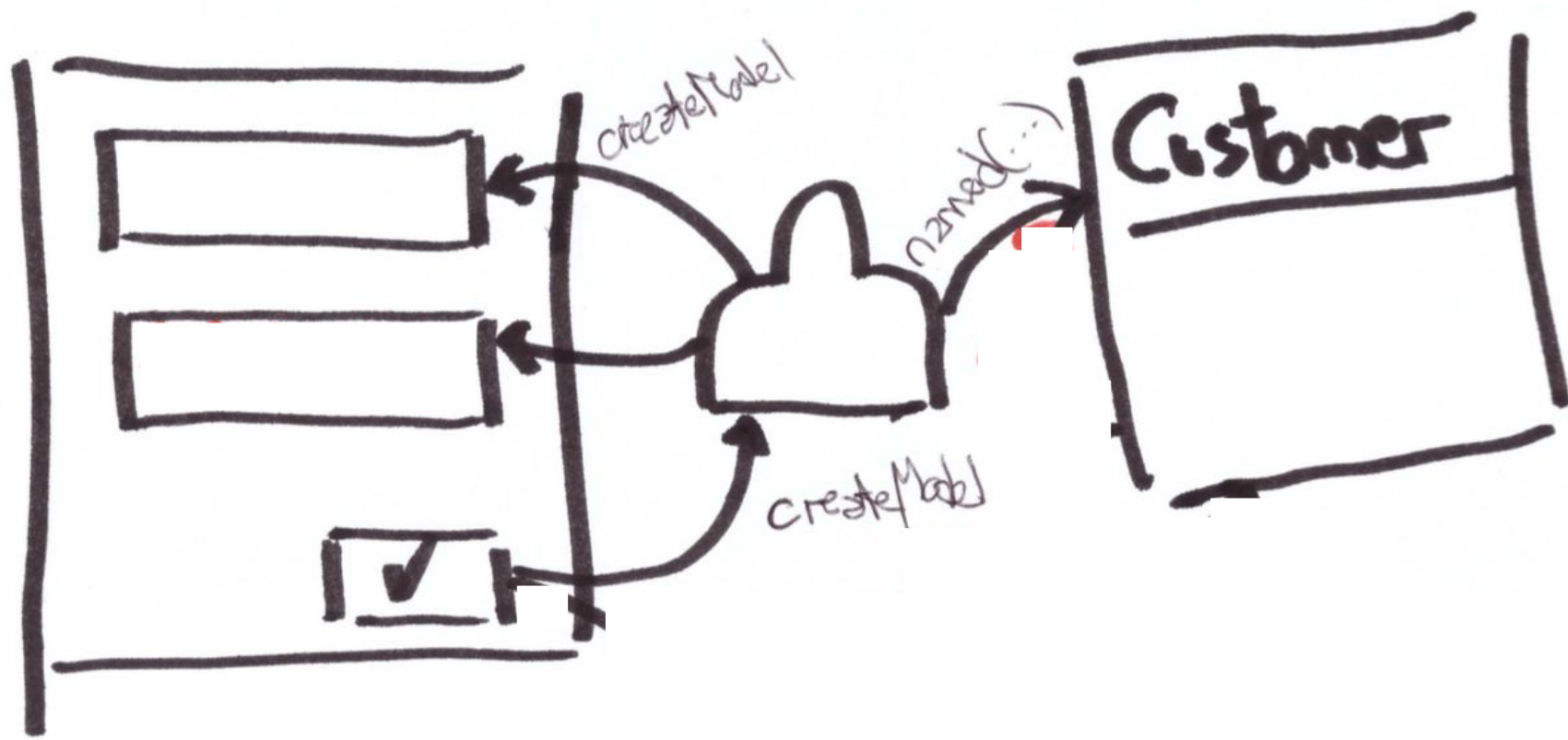


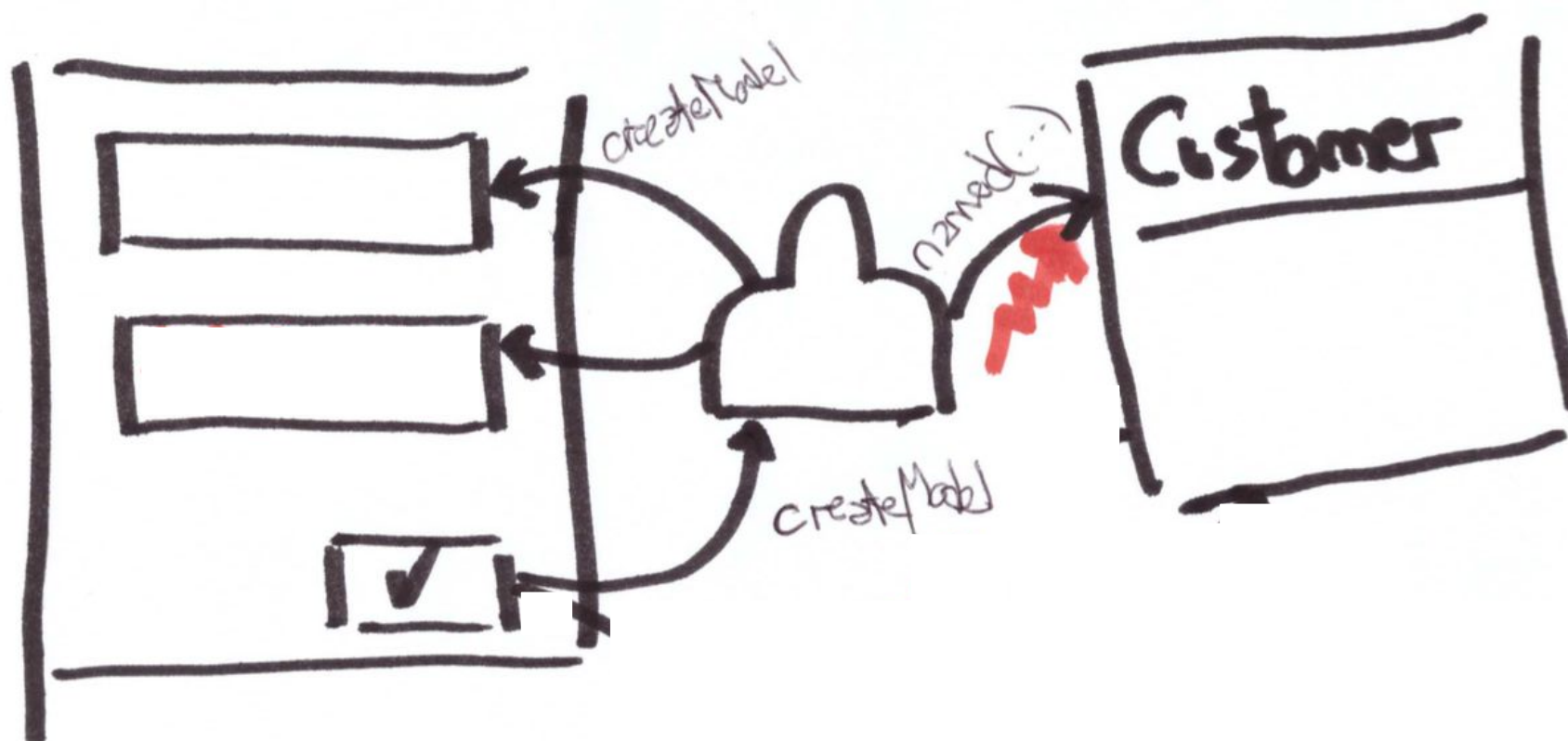
H5: Modelar el “conjunto” de Objetos

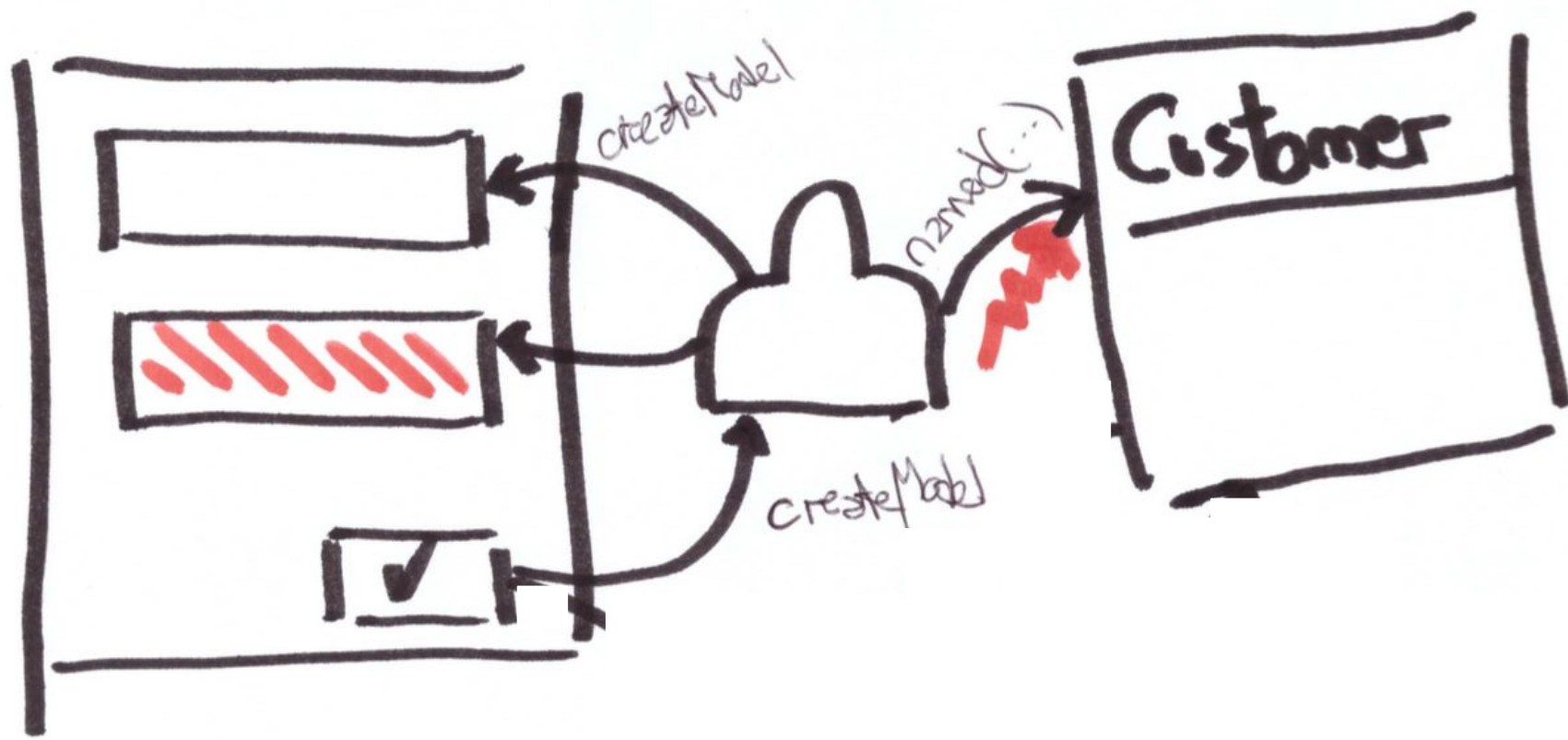
Tip: Se pueden tener distintas implementaciones del “conjunto”: Transient, persistent, client, etc

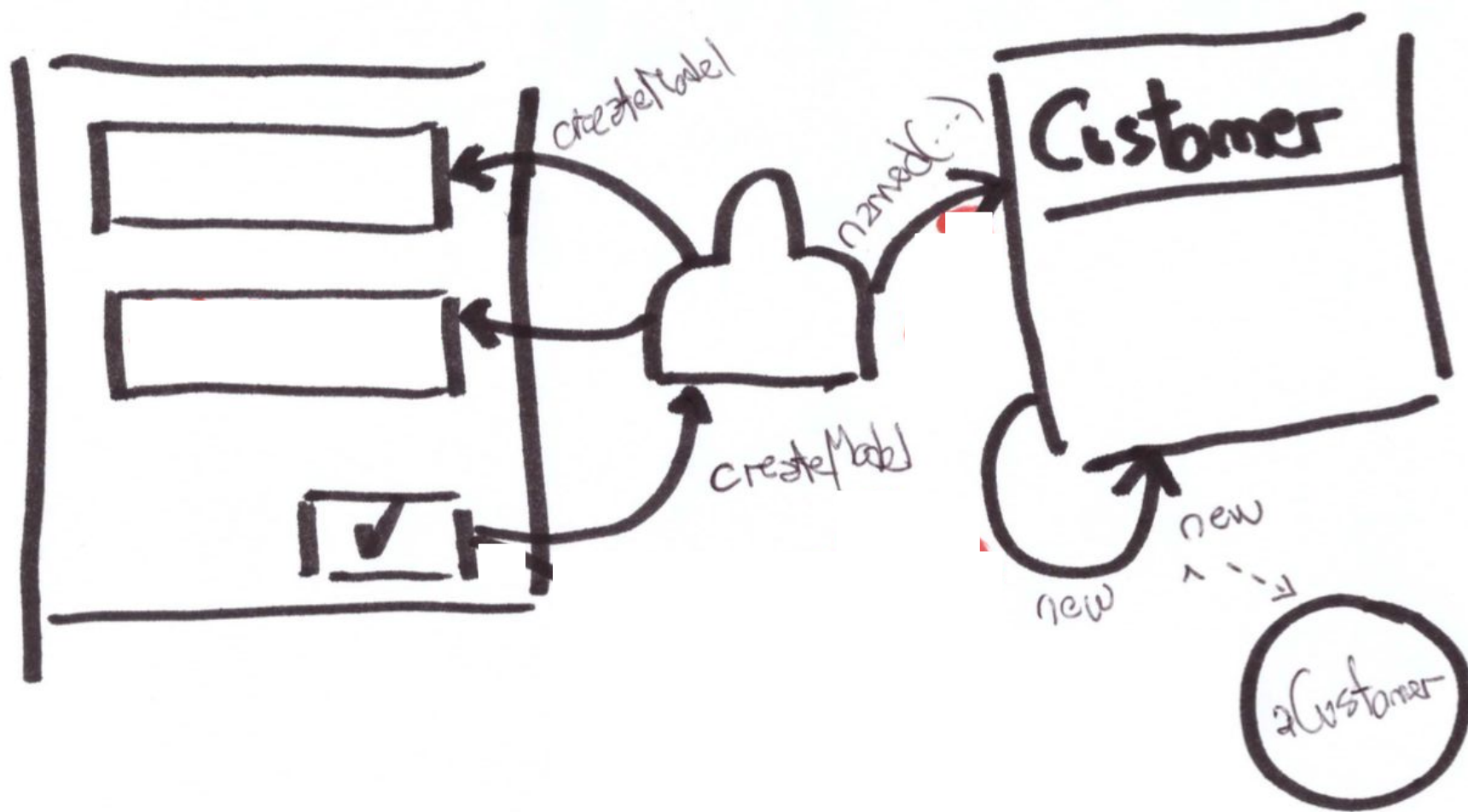


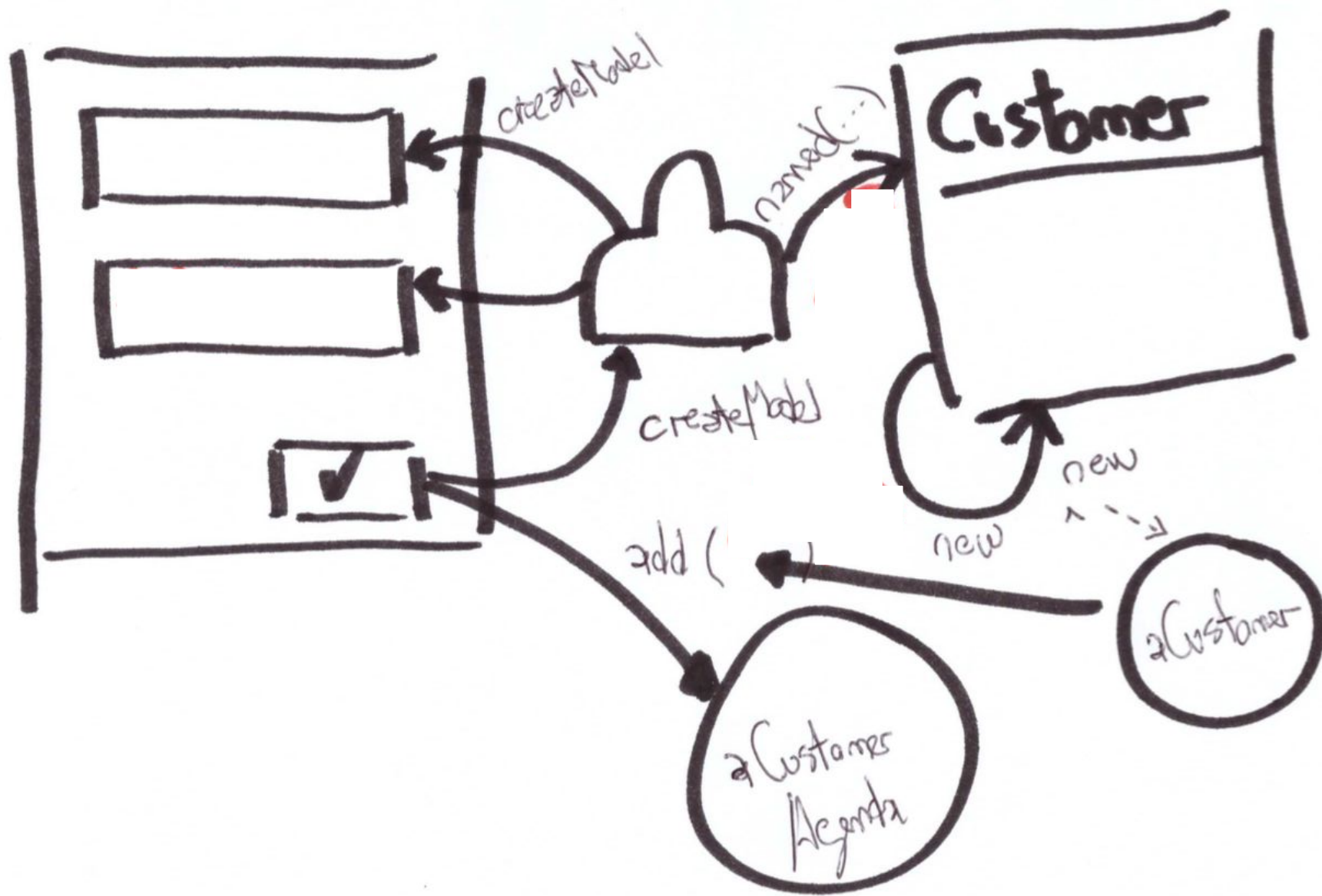


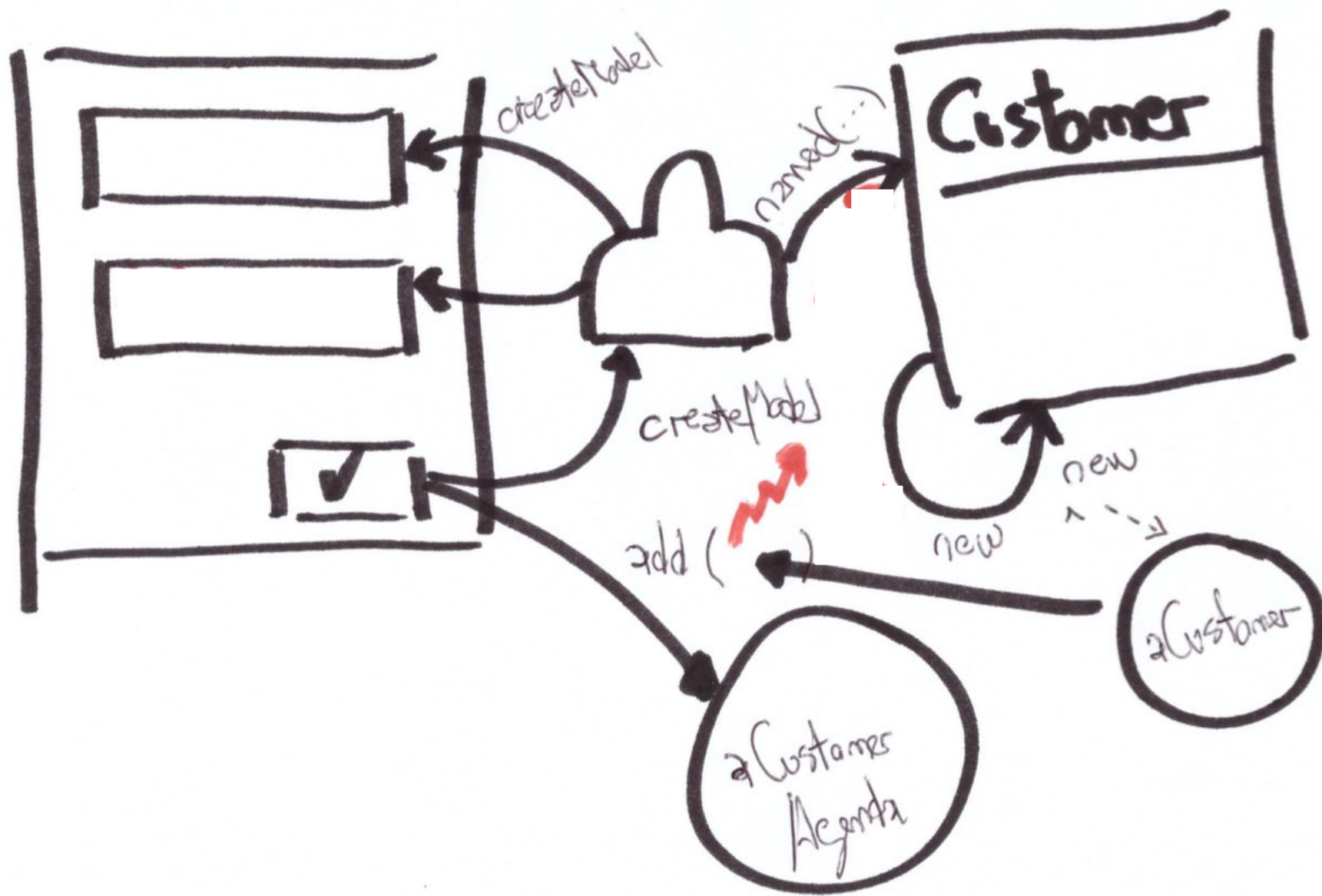


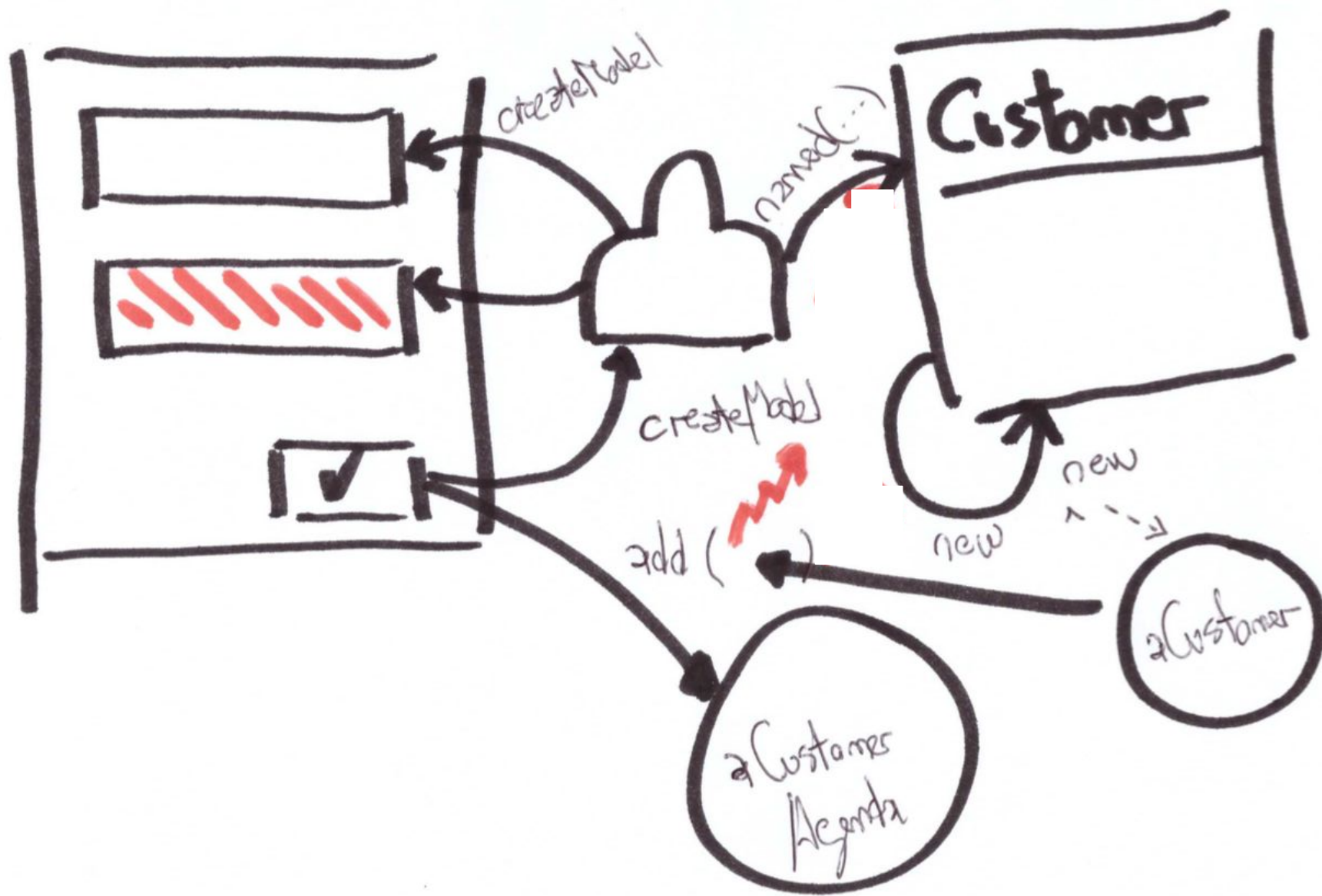


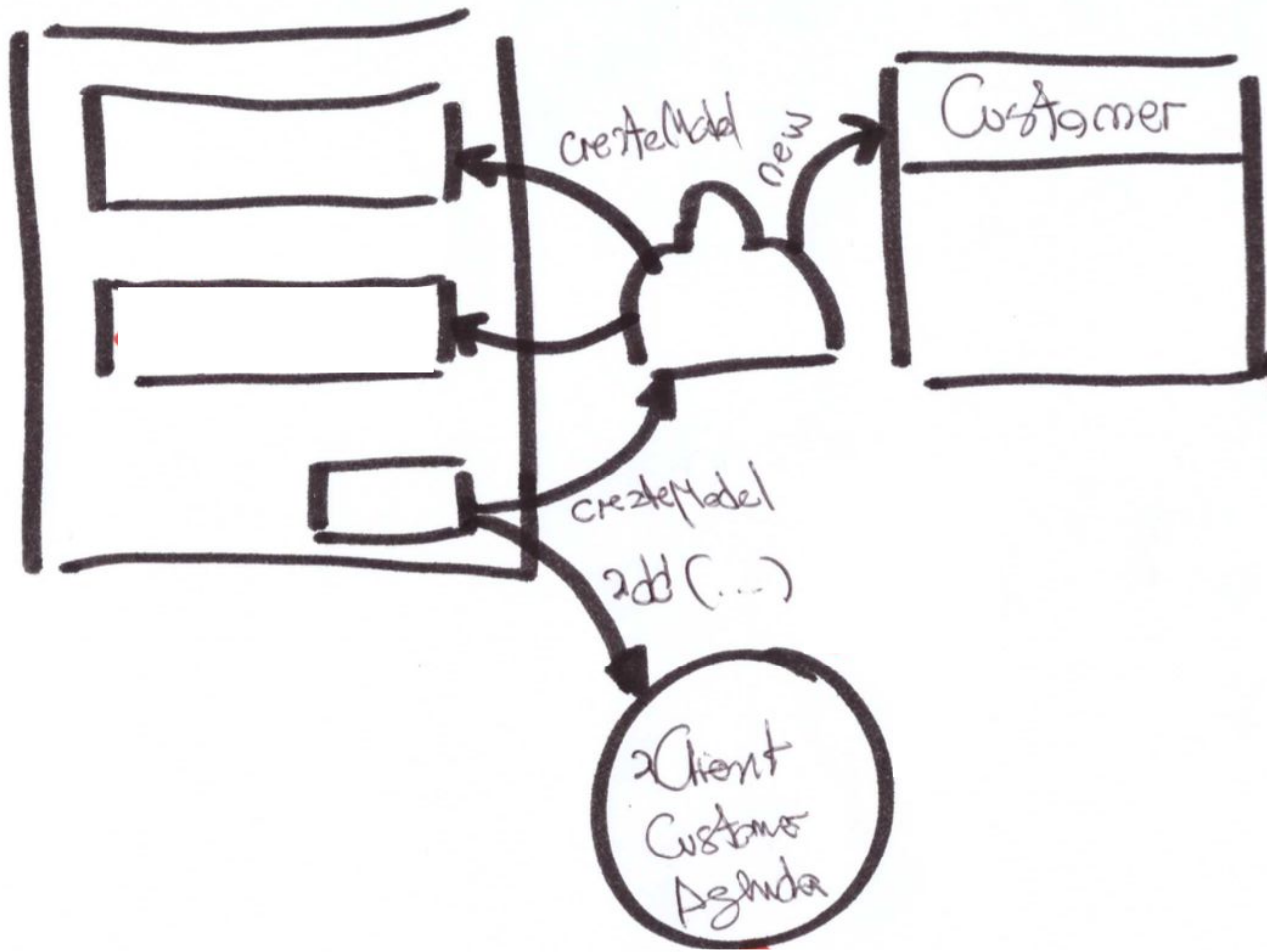


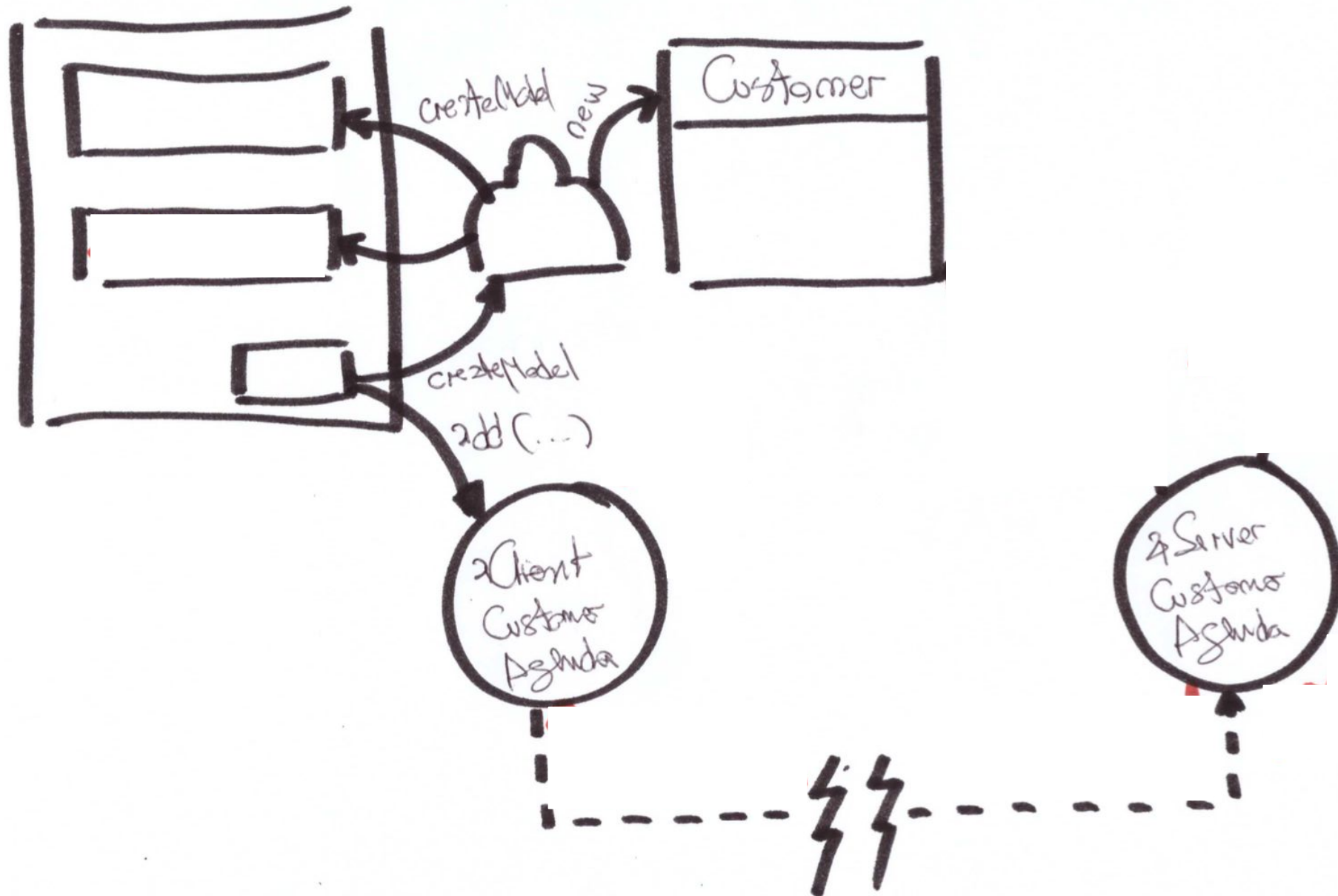


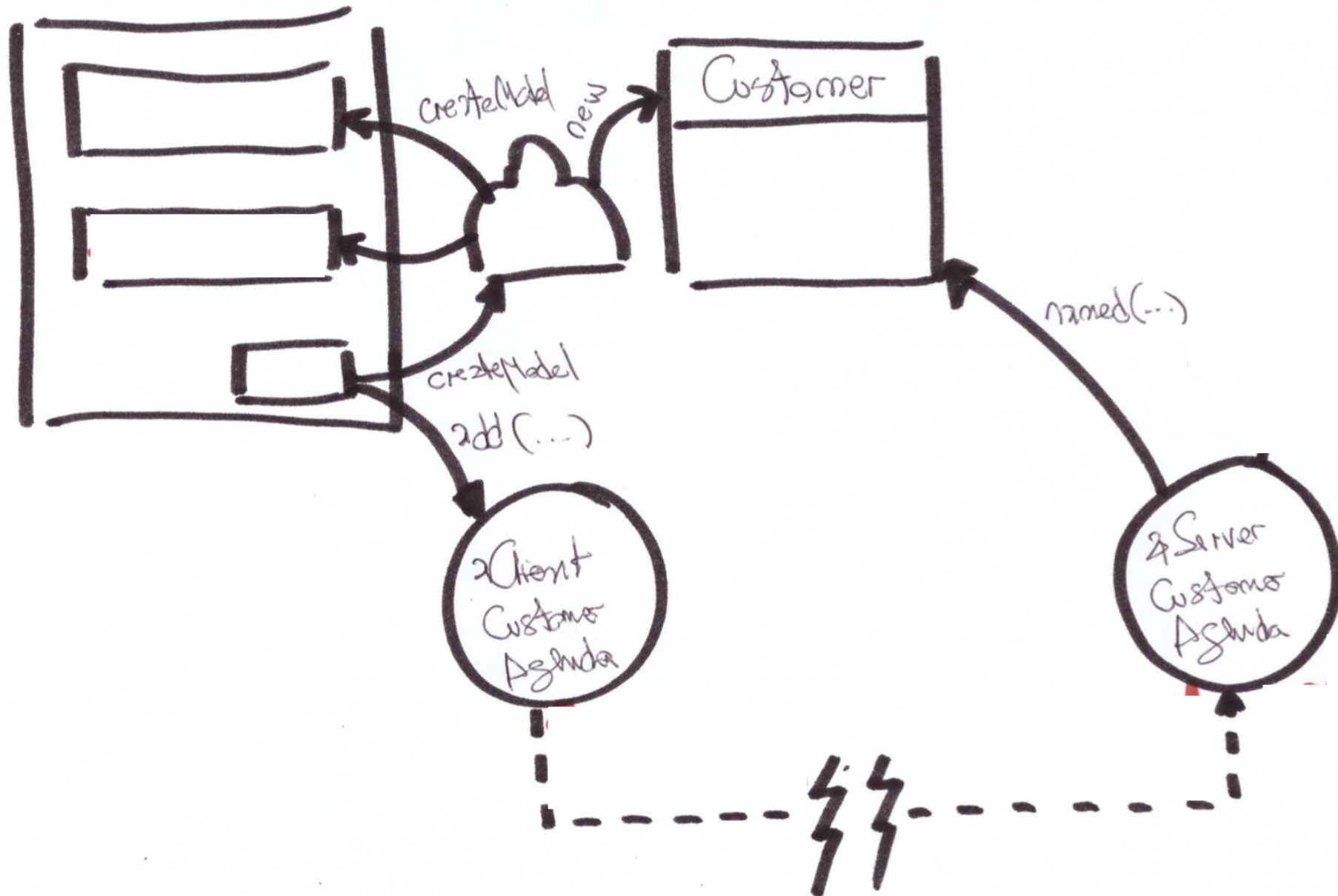


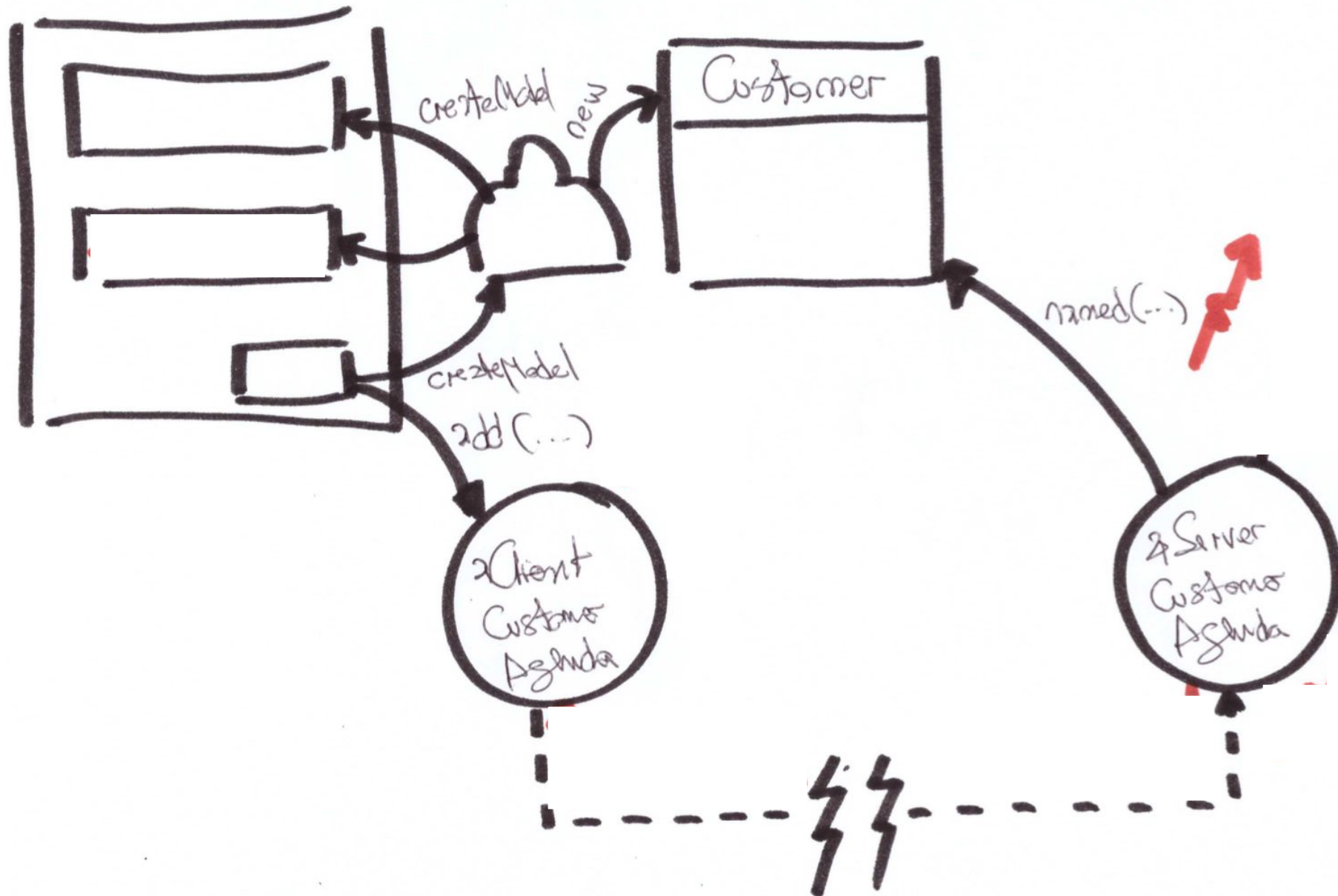


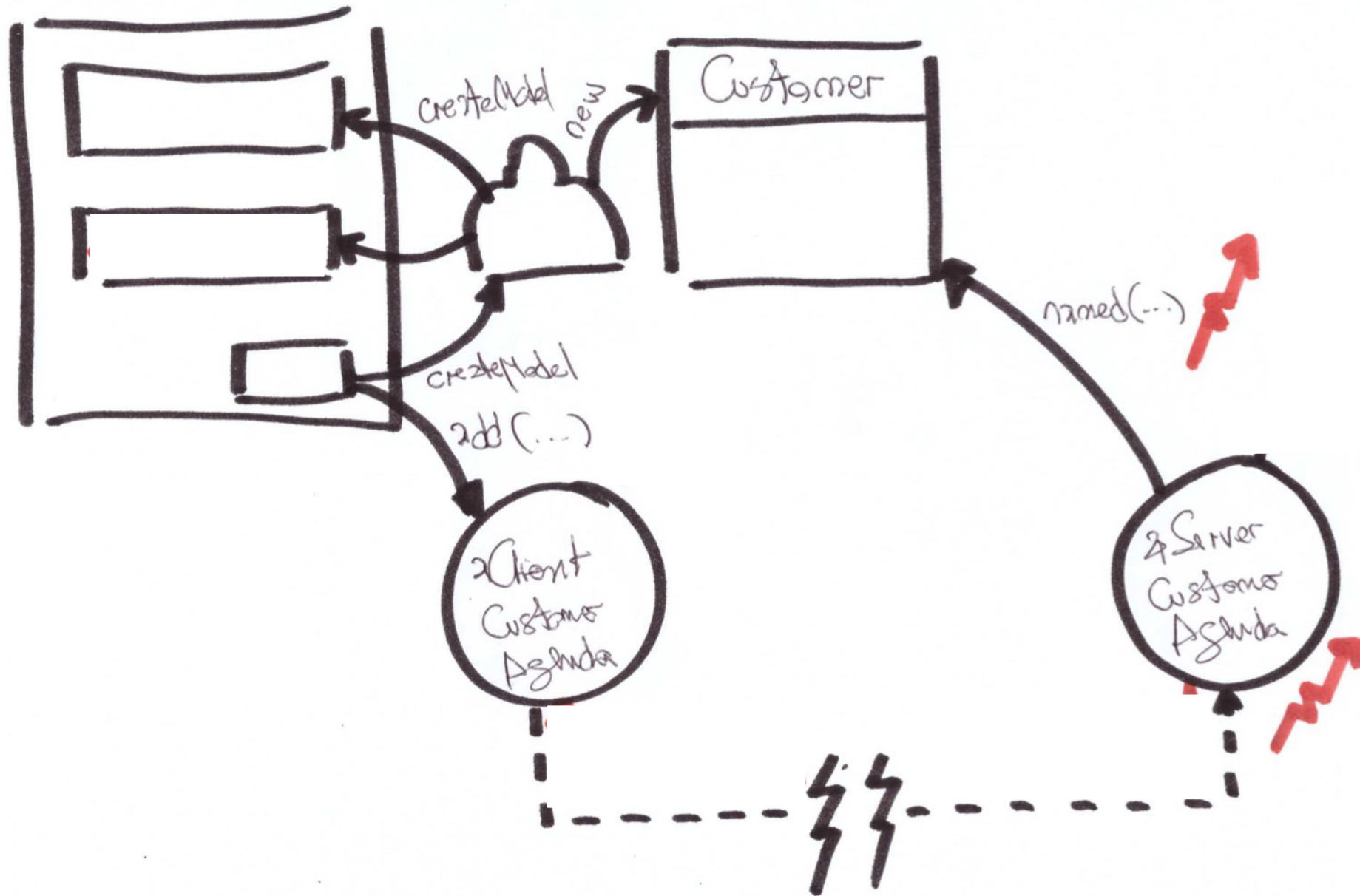


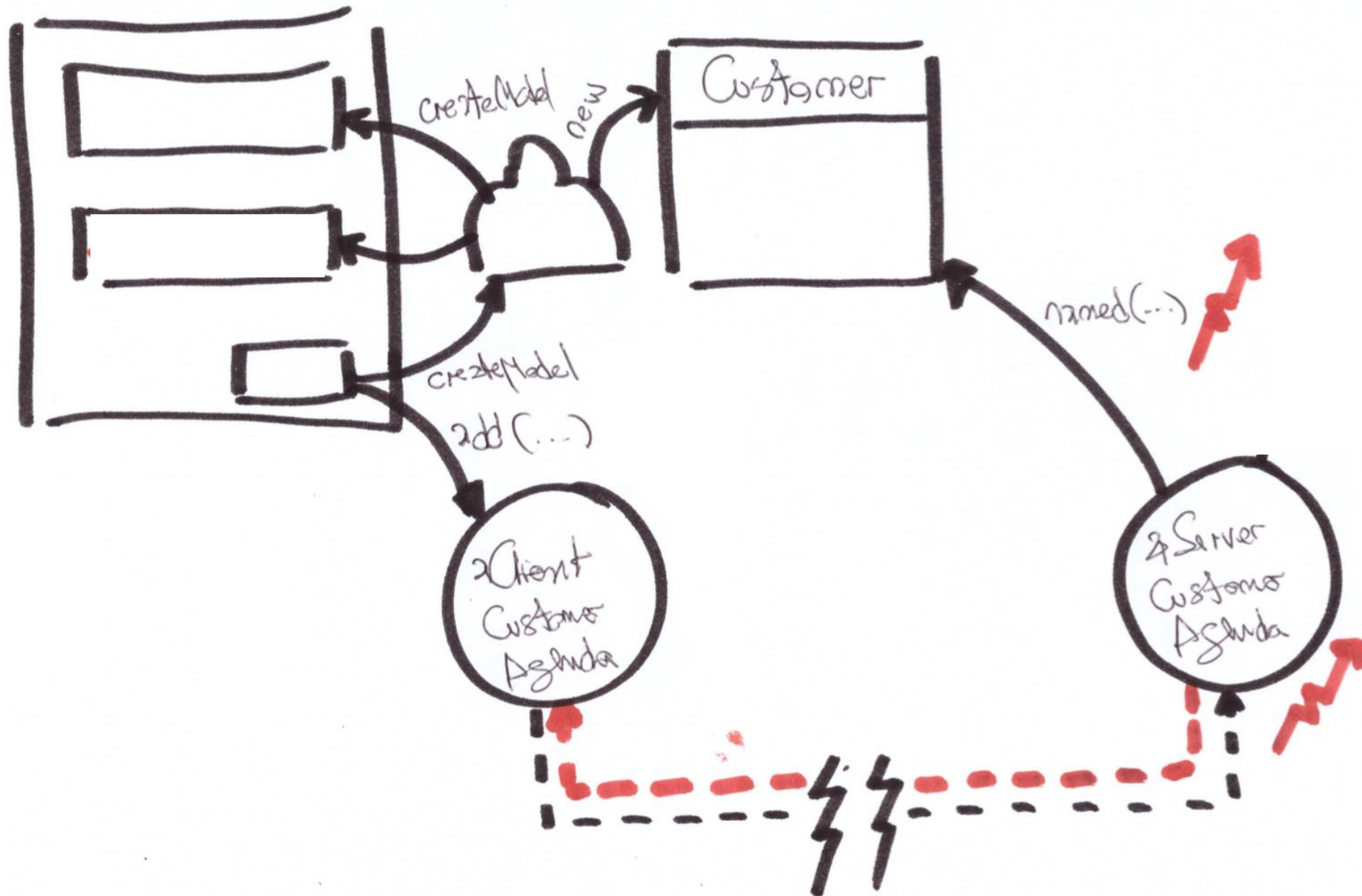


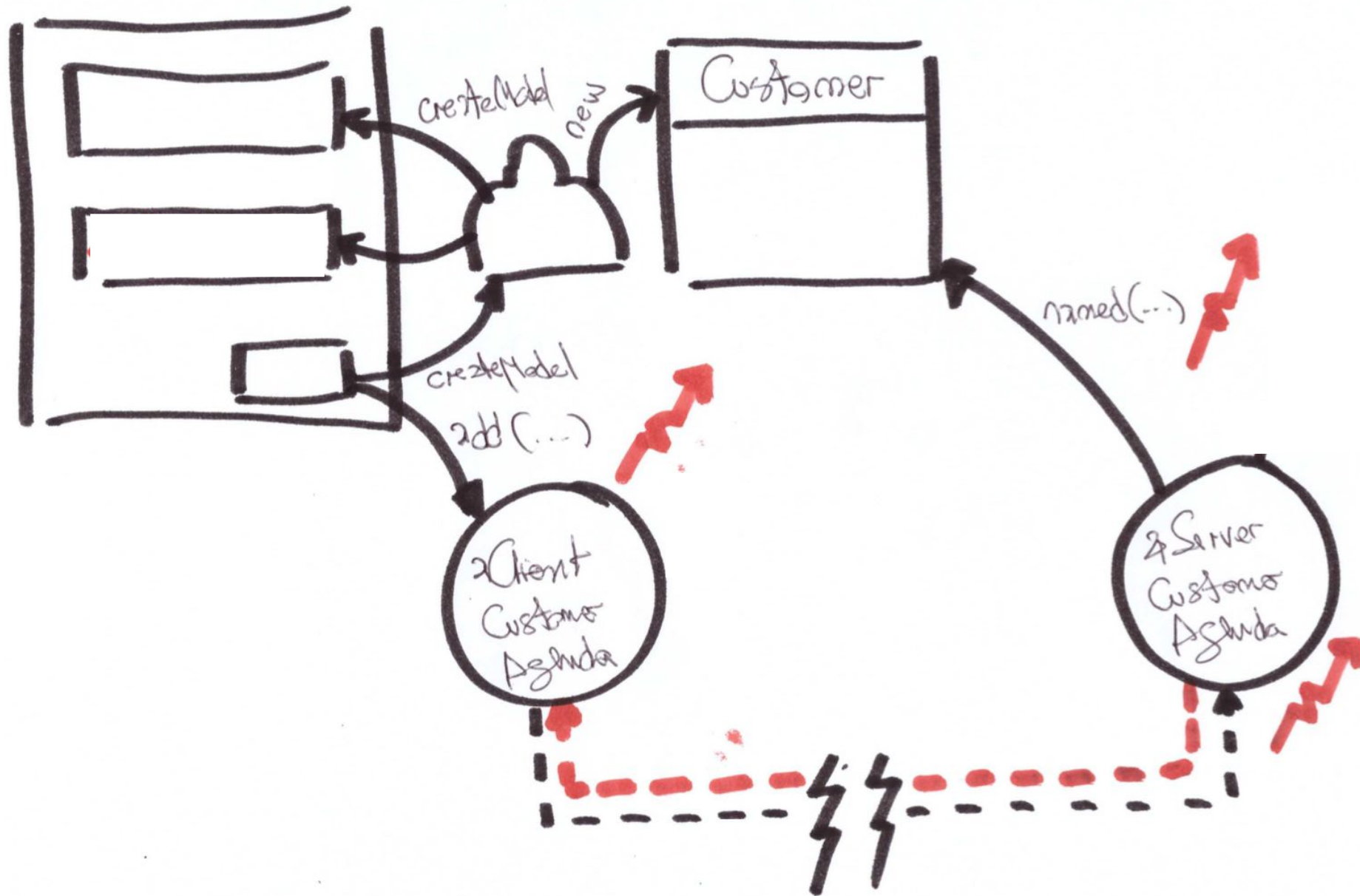


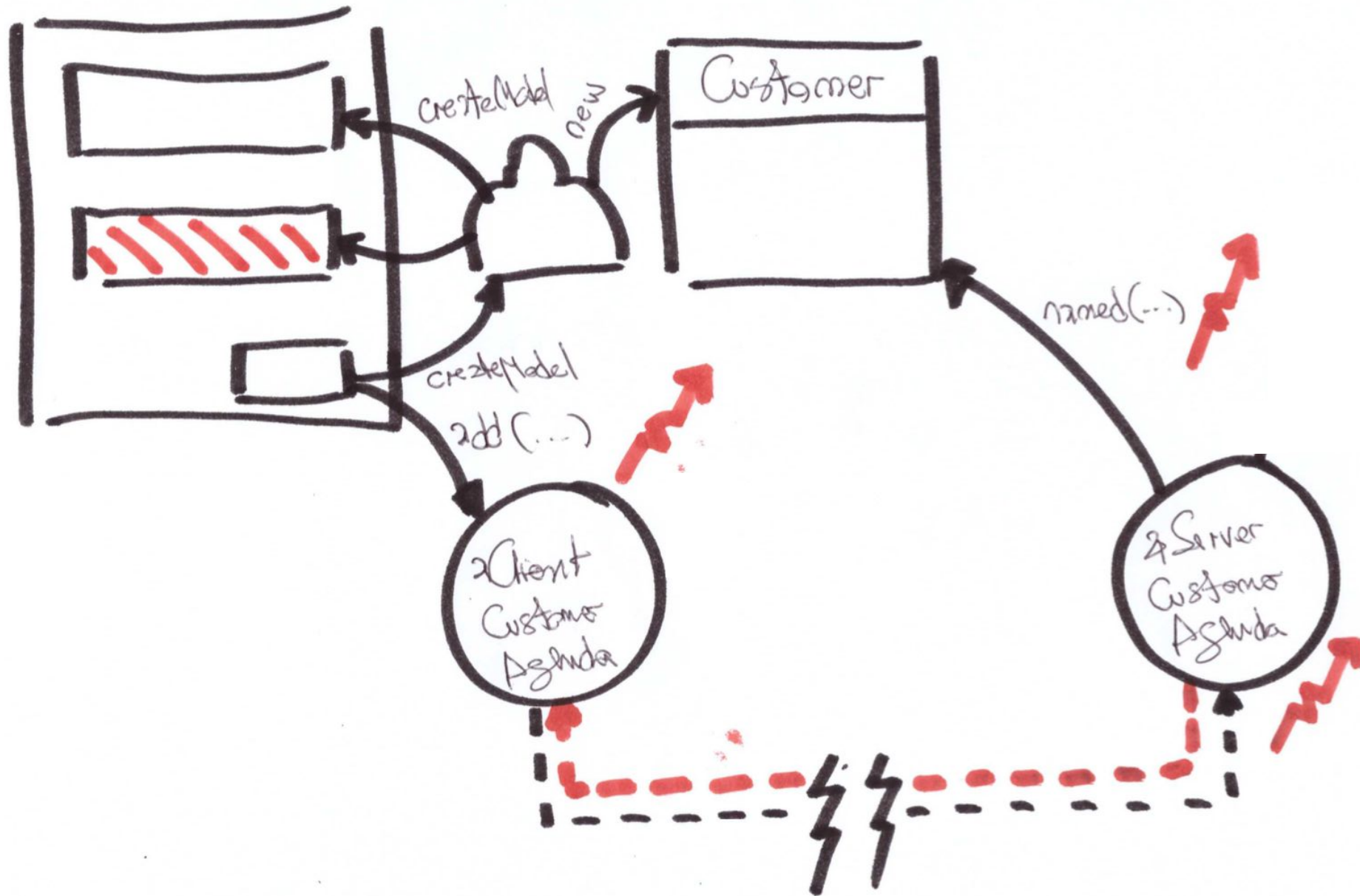


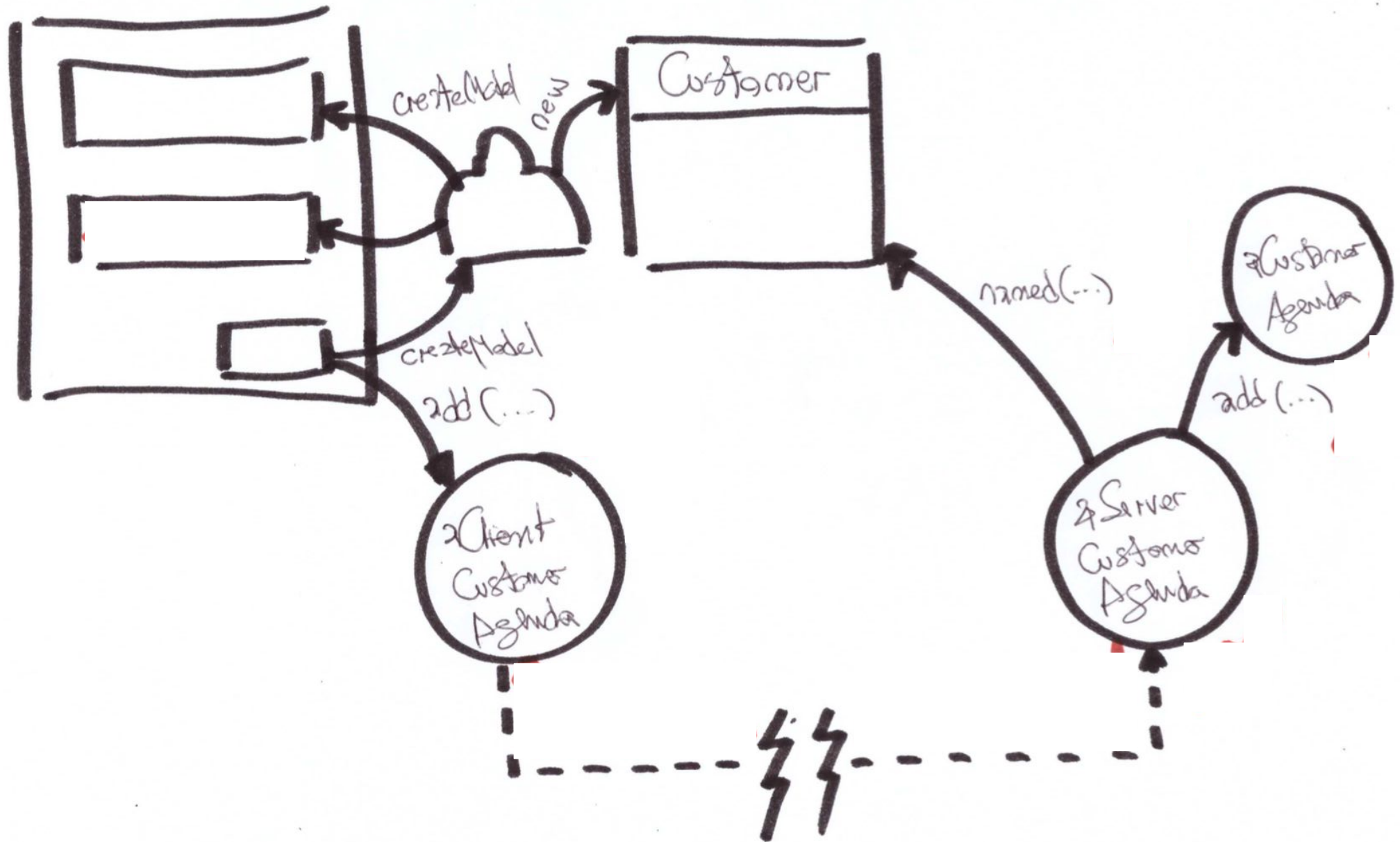


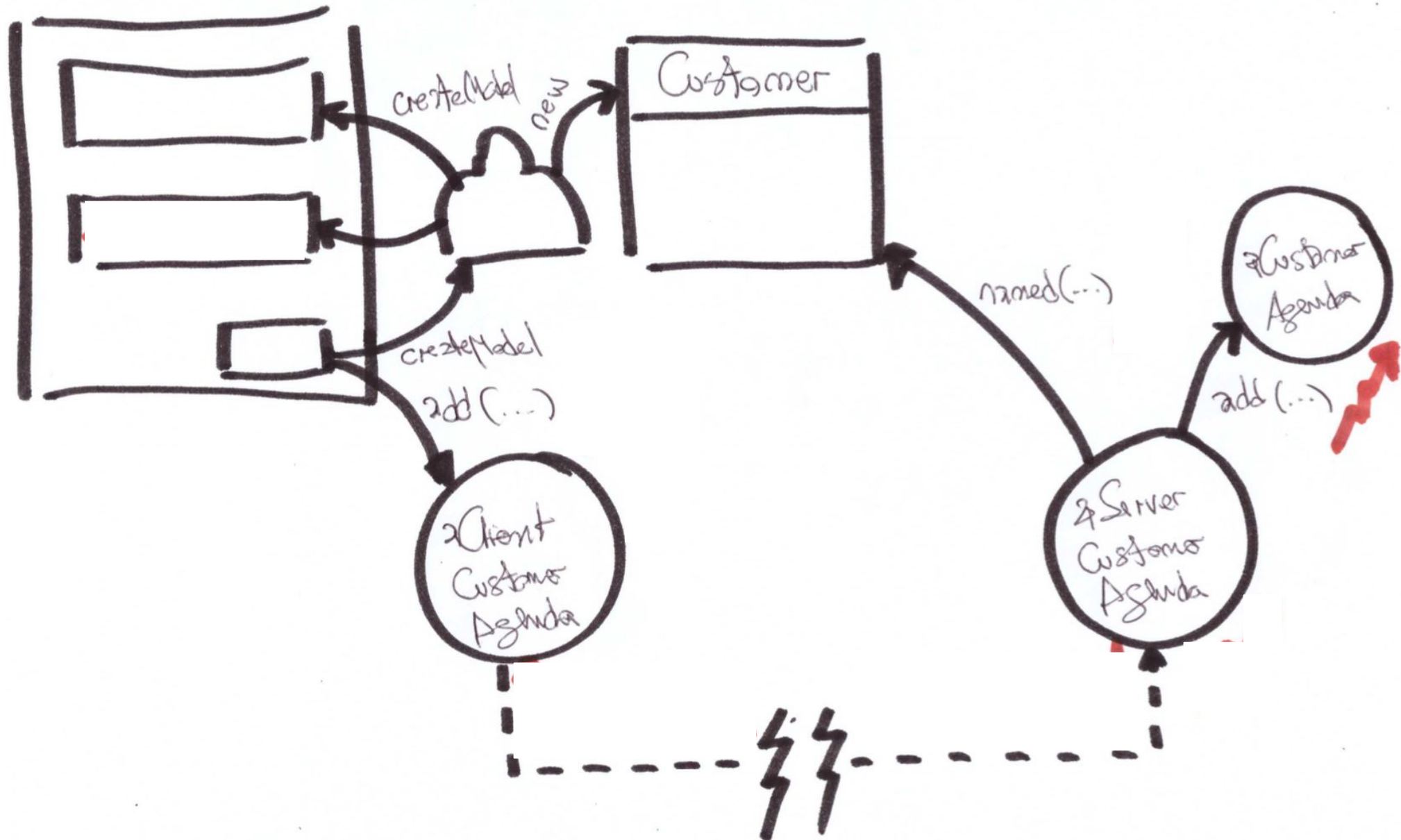


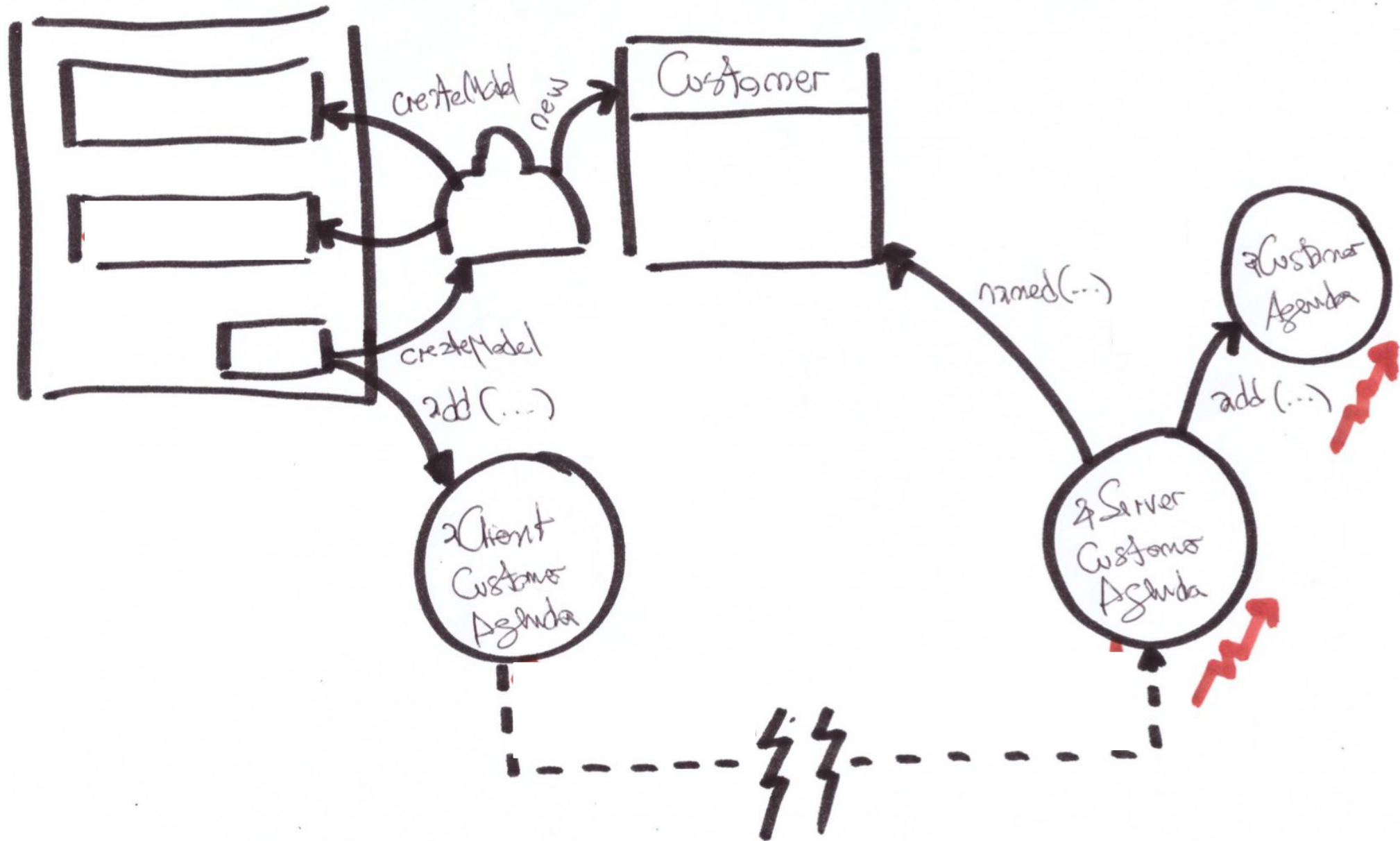


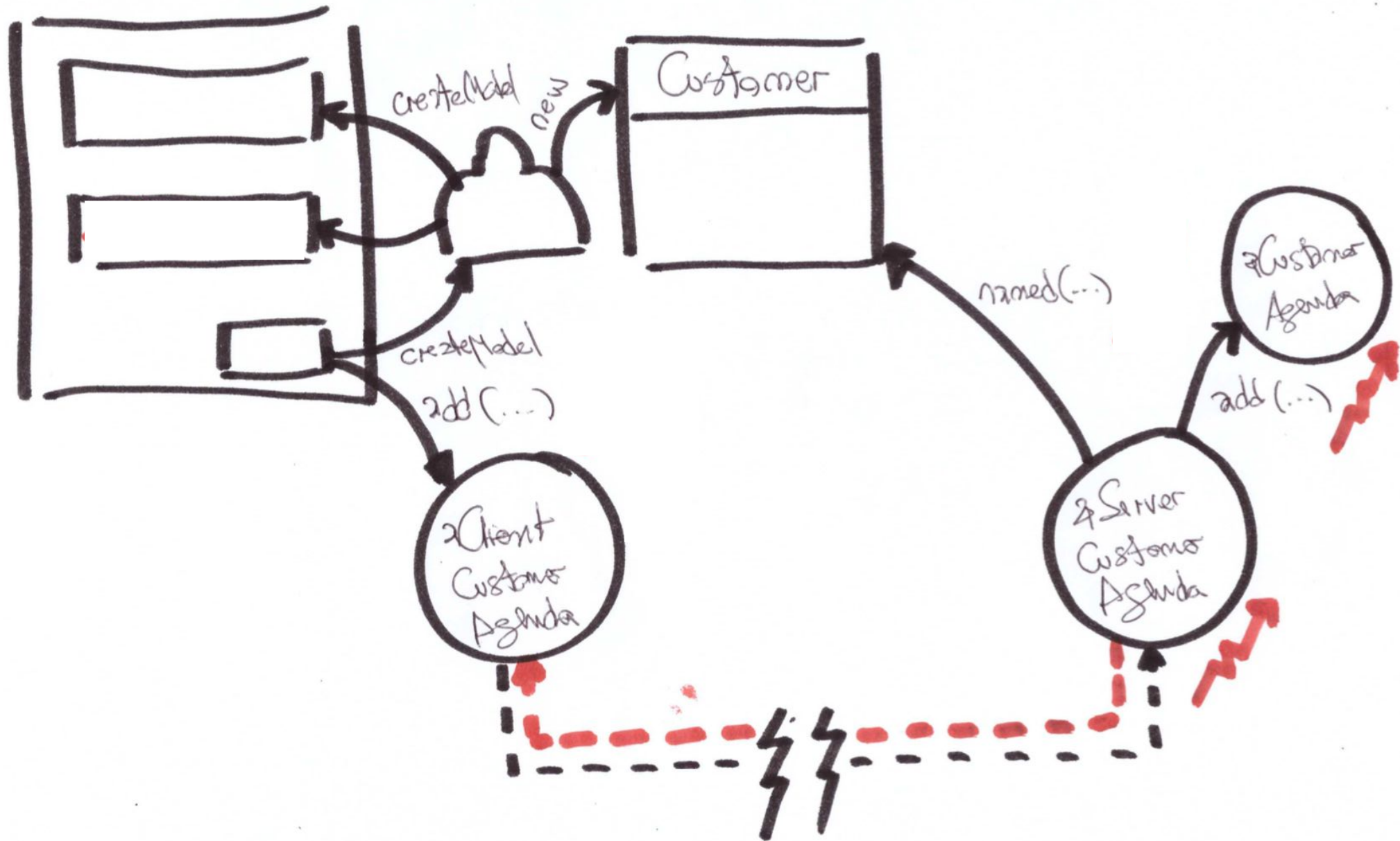


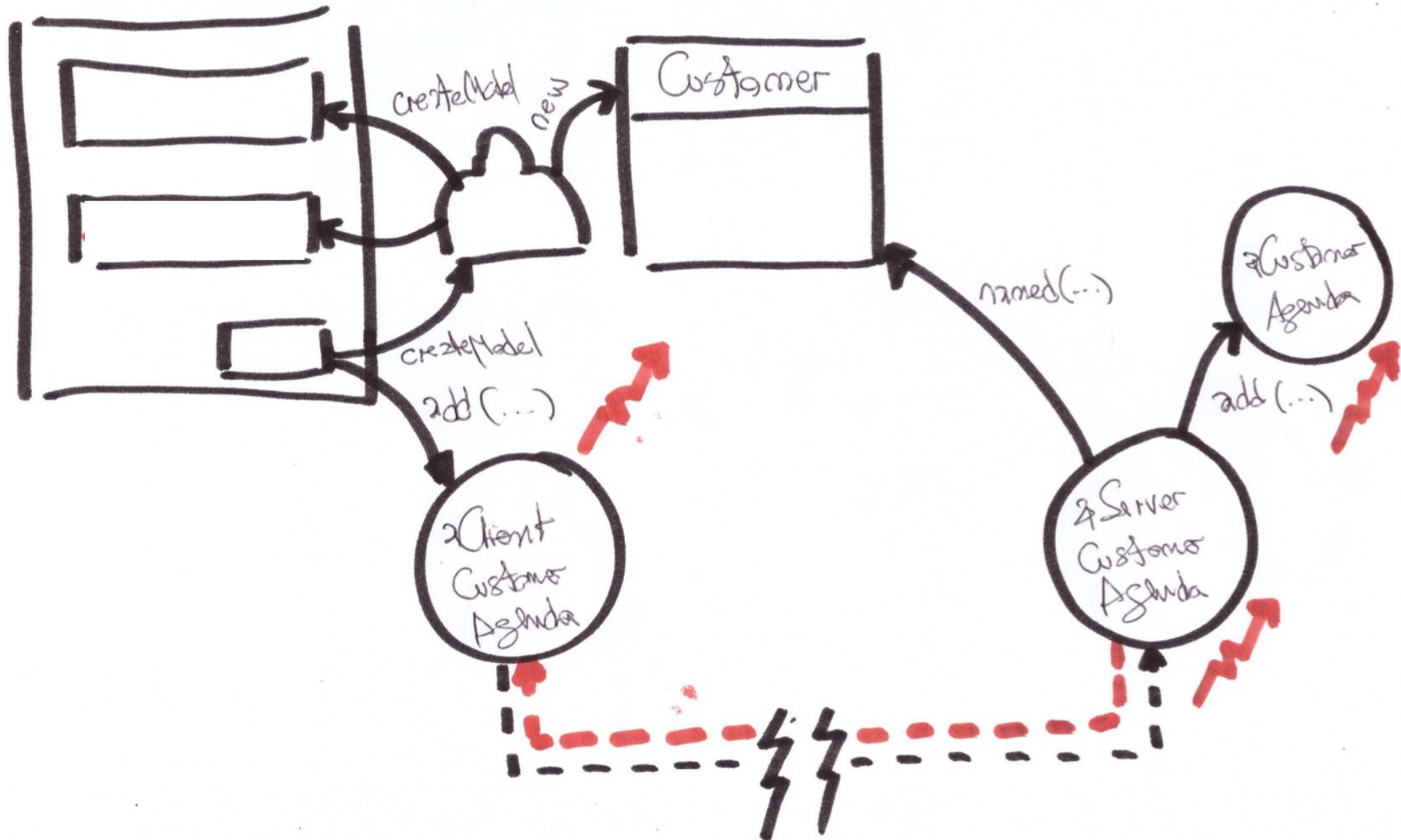


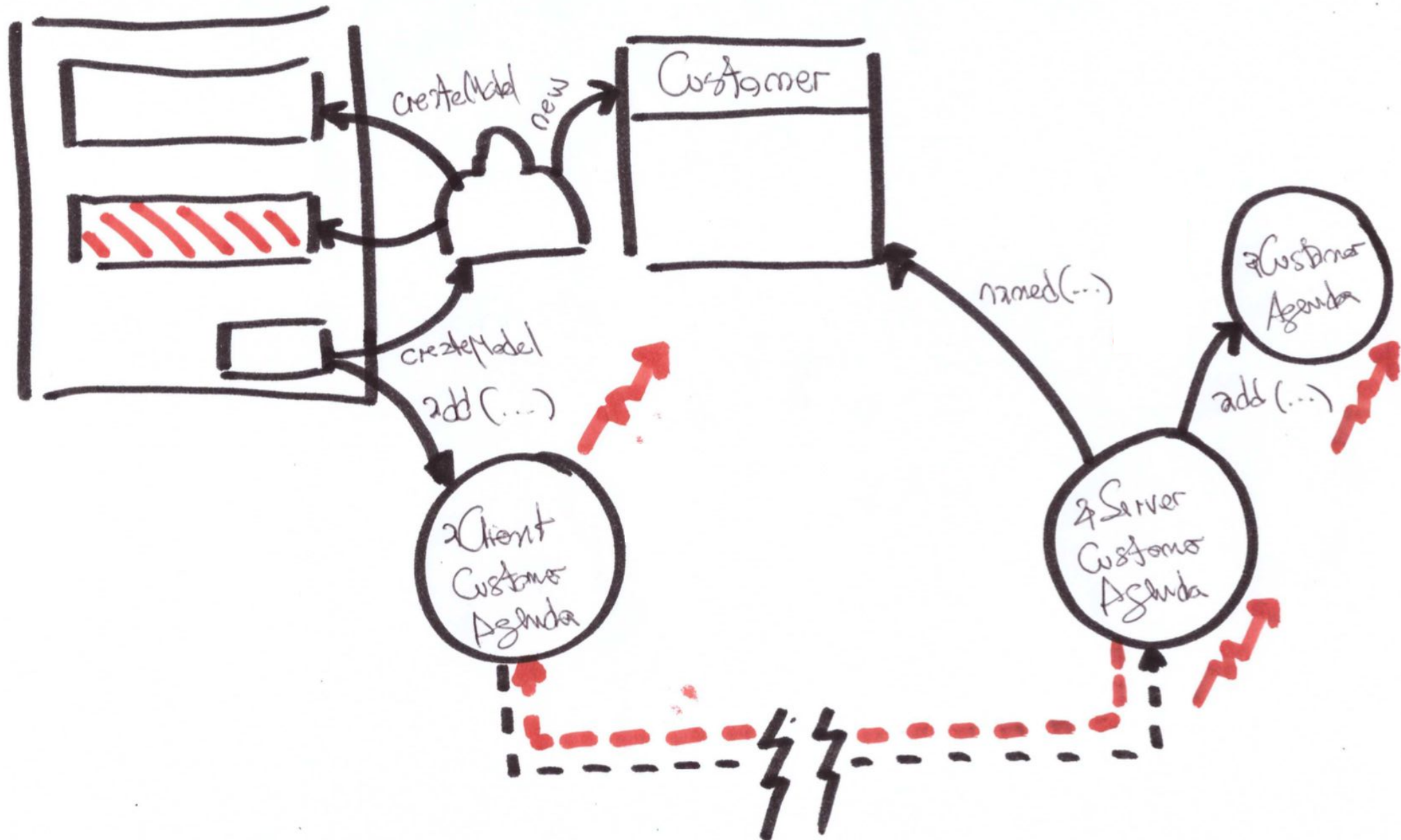












¿Preguntas?



¿Qué sucede si no encuentro
lo que busco?



```
update(anOriginalCustomer, aNewCustomer){
  AssertionsRunner.assertAll([...]);

  const customerToUpdate = this.customers.find(
    customer => customer.isIdentifiedAs(anOriginalCustomer.getDNI()));

  if(customerToUpdate !== undefined) ←
    customerToUpdate.syncWith(aNewCustomer);

  /* --...*/
}
```

```
removeCustomerIdentifiedAs(aDNI){
  const customerToRemove = this.customers.find(
    customer => customer.isIdentifiedAs(aDNI));

  if(customerToRemove !== undefined) ←
    this.removeCustomer(customerToRemove);

  /* --...*/
}
```



```
update(anOriginalCustomer, aNewCustomer){  
  AssertionsRunner.assertAll([...]);
```

```
  const customerToUpdate = this.customers.find(  
    customer => customer.isIdentifiedAs(anOriginalCustomer.getDNI()));
```

```
  if(customerToUpdate !== undefined)  
    customerToUpdate.syncWith(aNewCustomer);
```

```
  /* --...*/  
}
```

```
removeCustomerIdentifiedAs(aDNI){
```

```
  const customerToRemove = this.customers.find(  
    customer => customer.isIdentifiedAs(aDNI));
```

```
  if(customerToRemove !== undefined)  
    this.removeCustomer(customerToRemove);
```

```
  /* --...*/  
}
```




```
update(anOriginalCustomer, aNewCustomer){
  AssertionsRunner.assertAll([...]);

  const customerToUpdate = this.customers.find(
    customer => customer.isIdentifiedAs(anOriginalCustomer.getDNI()));

  if(customerToUpdate !== undefined) ←
    customerToUpdate.syncWith(aNewCustomer);

  /* --...*/
}
```

```
removeCustomerIdentifiedAs(aDNI){
  const customerToRemove = this.customers.find(
    customer => customer.isIdentifiedAs(aDNI));

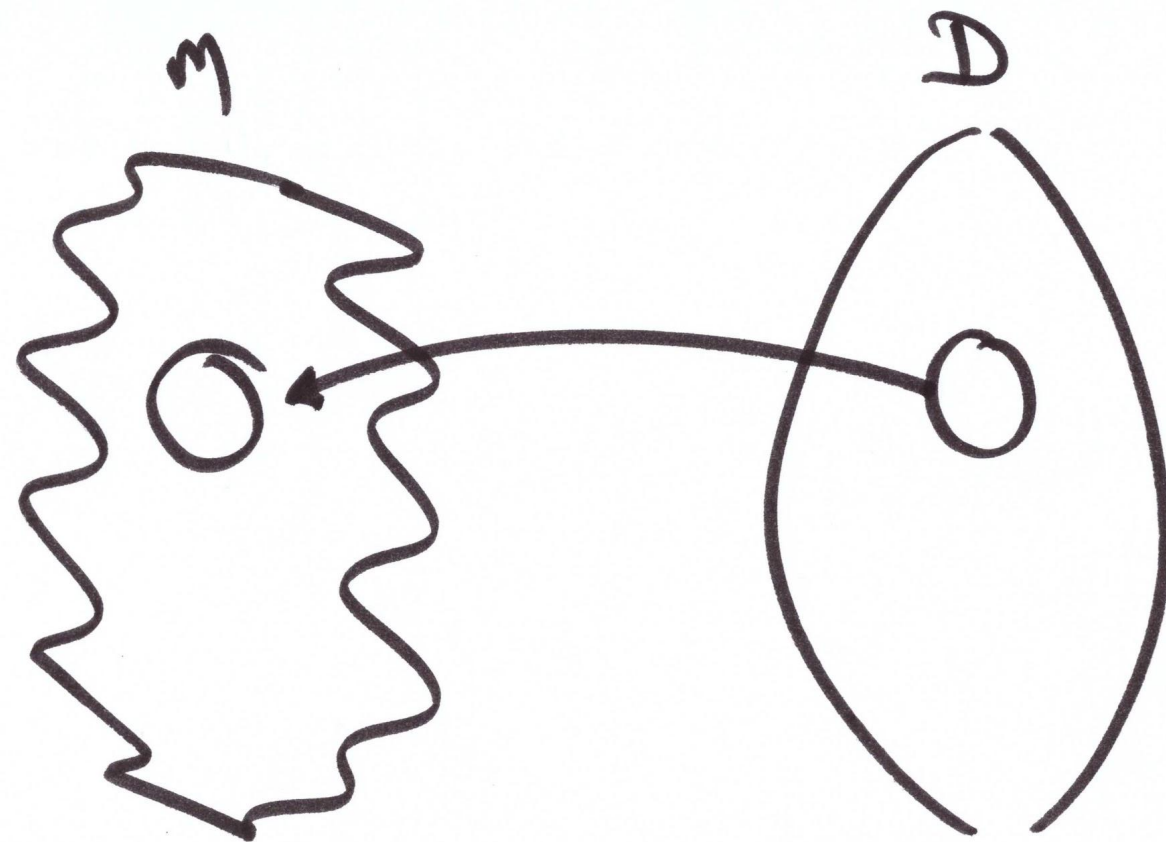
  if(customerToRemove !== undefined) ←
    this.removeCustomer(customerToRemove);

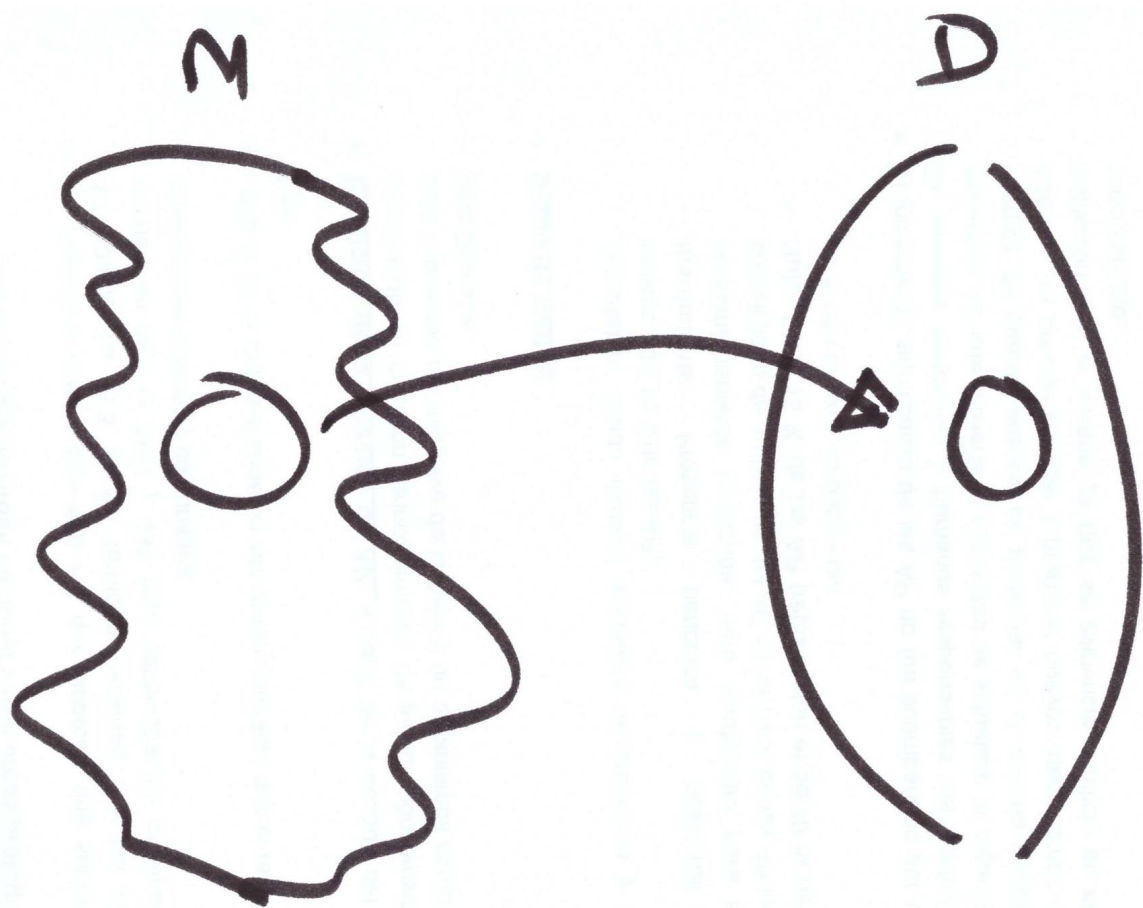
  /* --...*/
}
```

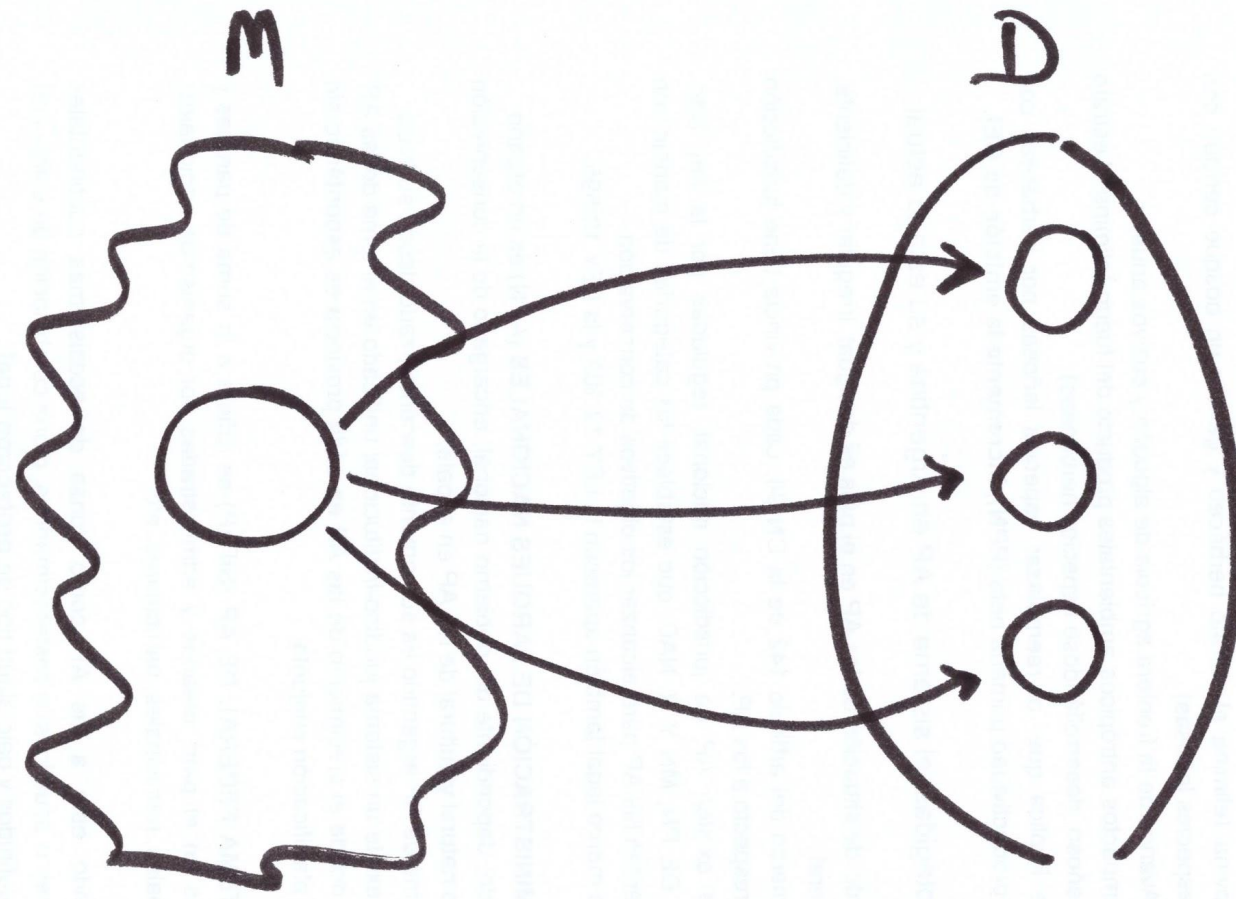


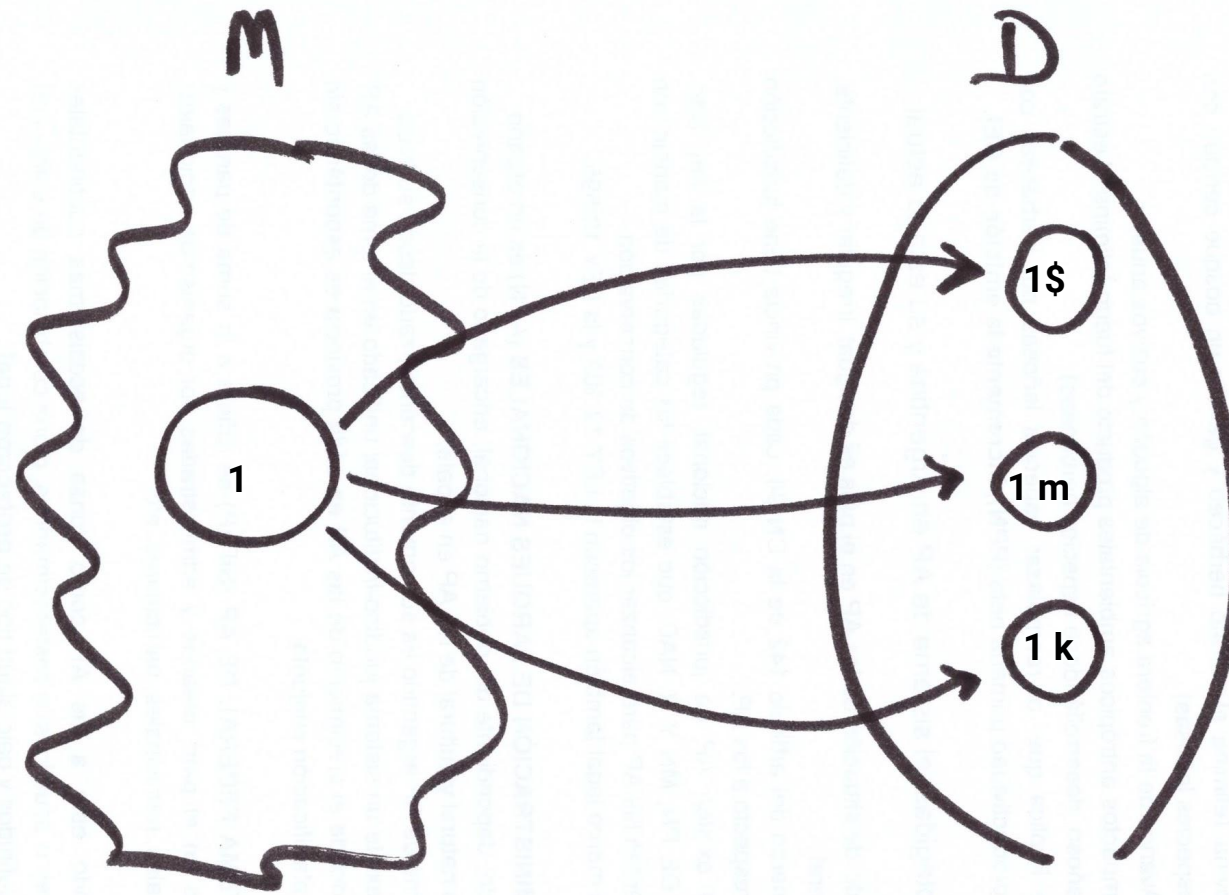
Mapeo ente<->objeto

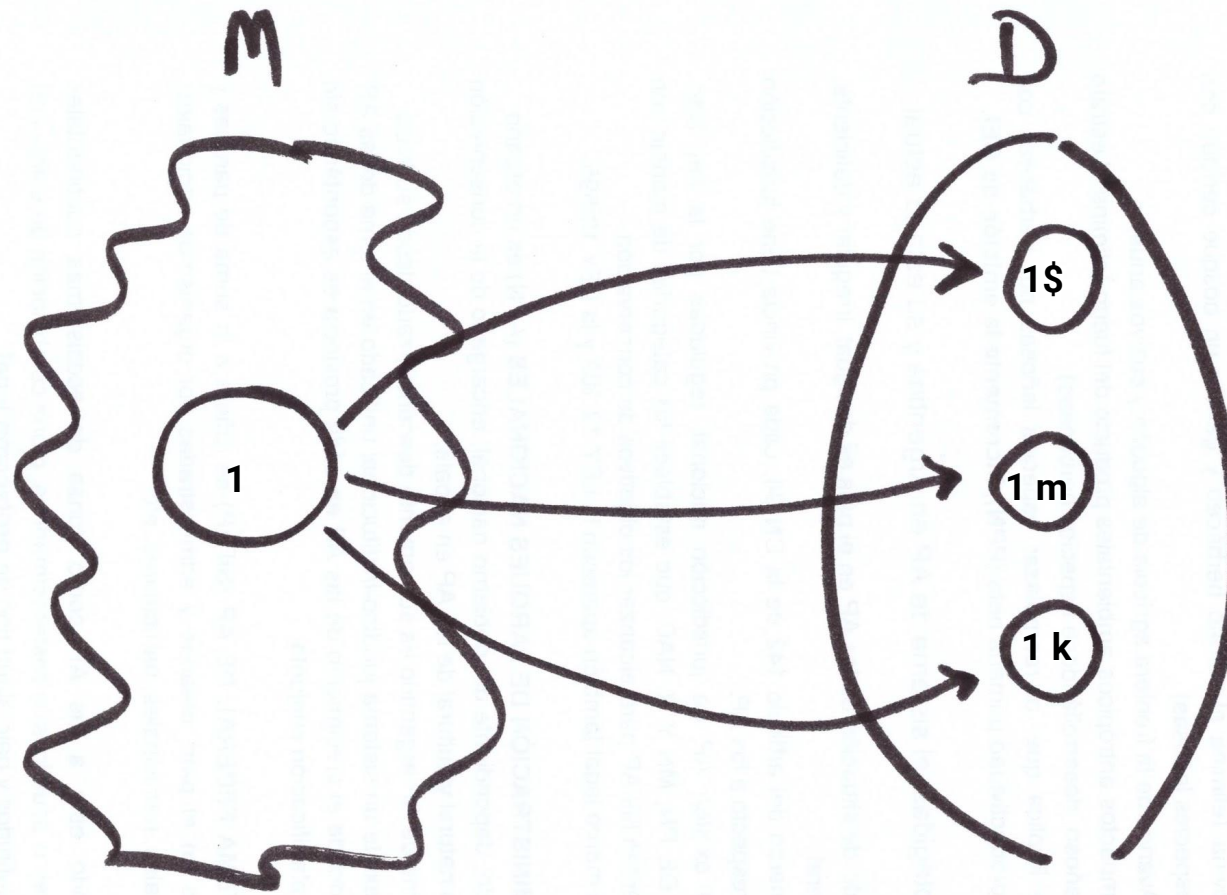






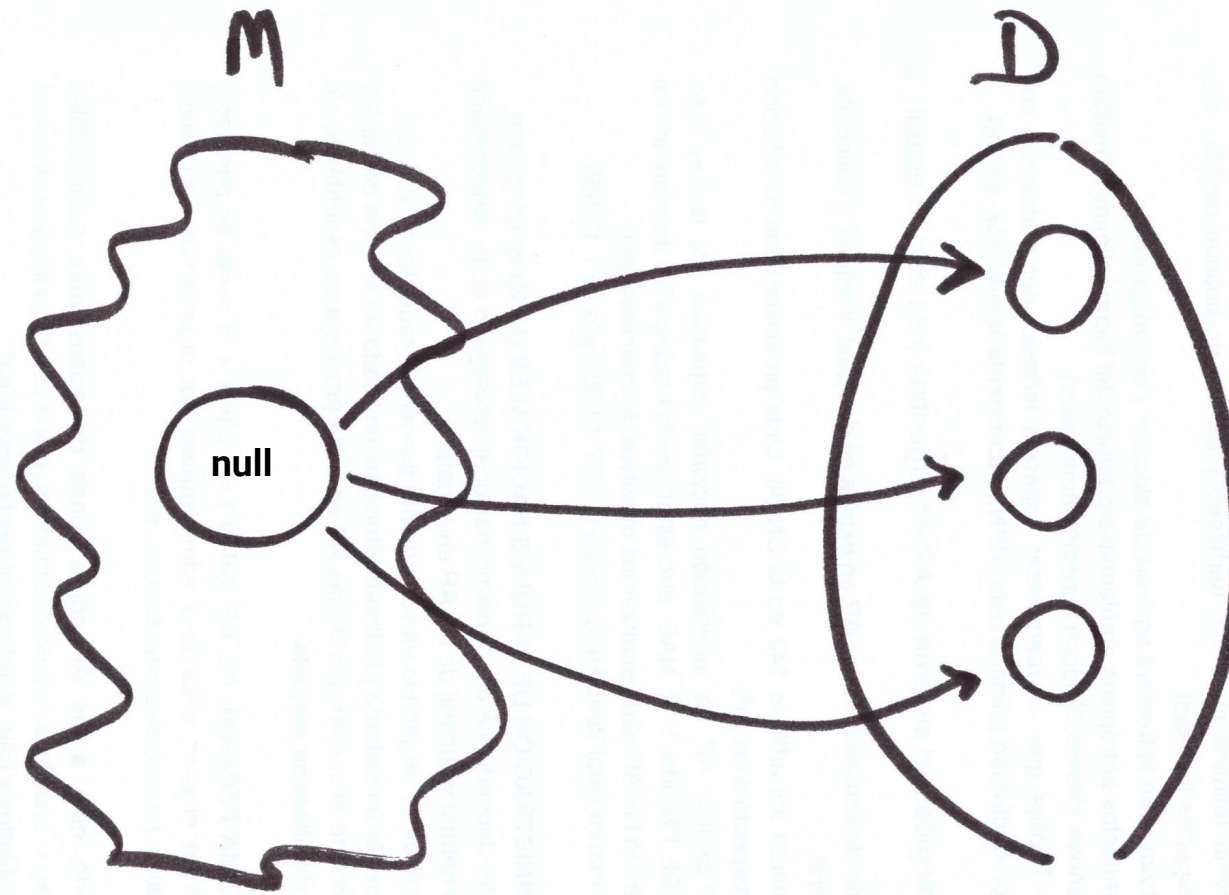






Perdida de Información





Configuraciones



GLOBAL

Idioma

Español

Moneda

null

Utilizar el formato simple de fecha

Las fechas marcadas se mostrarán como hoy, aver ..

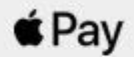


No se puede realizar la llamada

null y tú no son contactos en Messenger. null recibirá un mensaje con una solicitud para conectarse.

ACEPTAR

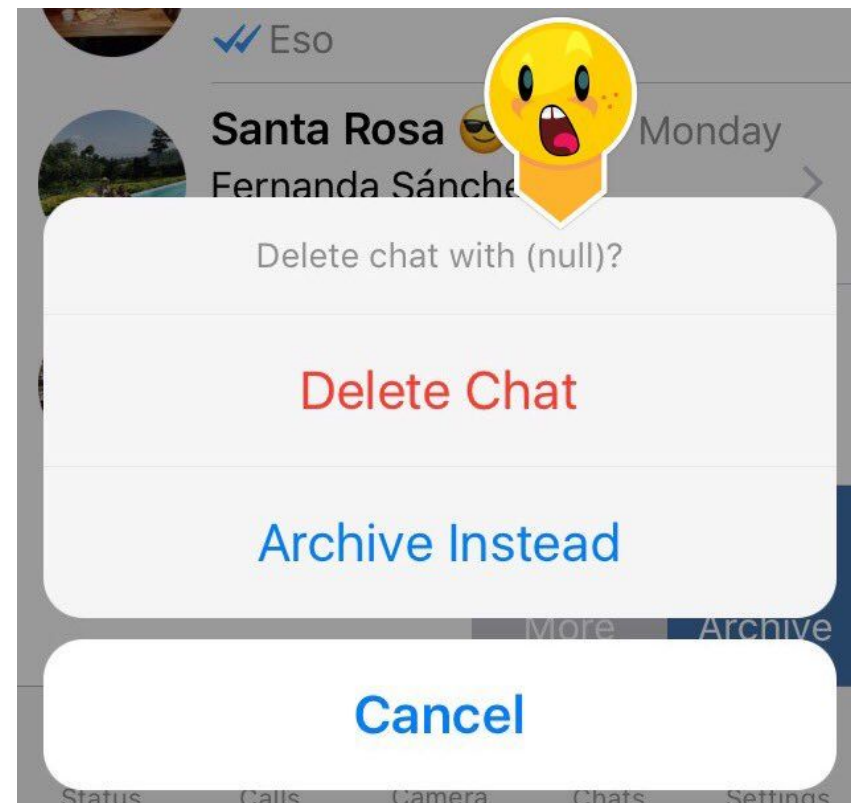
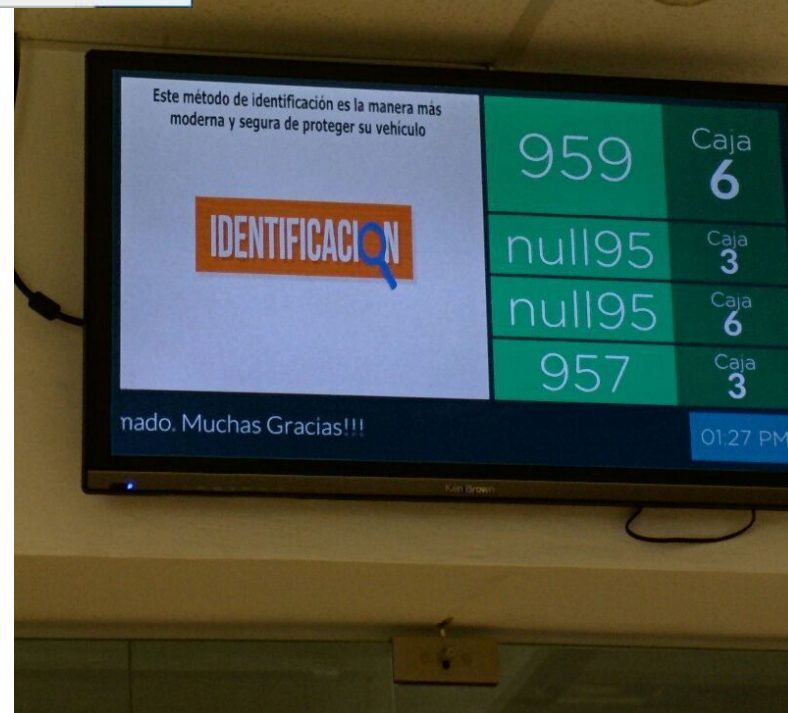




“(null)” could not be found to confirm this payment.

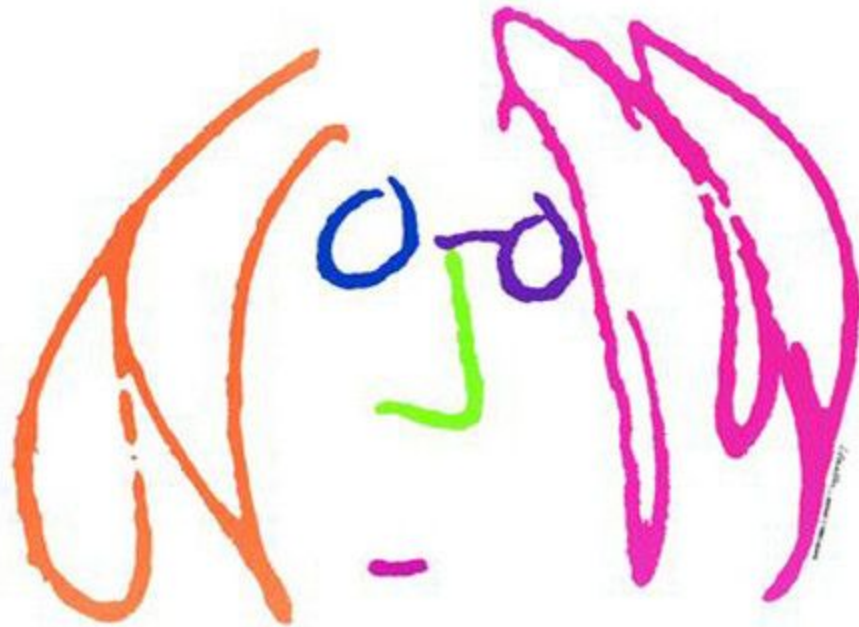
Make sure “(null)” is nearby, powered on, and has Bluetooth enabled.

OK



I M A G I N E

*A world without **null***



M U S I C F R O M T H E M O T I O N P I C T U R E



Null References: The Billion Dollar Mistake



LIKE



6



View Presentation



Speed: 1X 1.25X 1.5X 2X



01:01:58

Summary

Null References: The Billion Dollar Mistake

Tony Hoare



Ejemplo de Customer (ahora en Ruby :-))



¿Cómo sacar esos ifs?



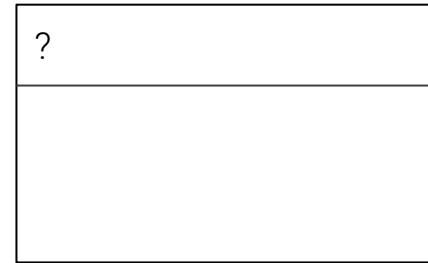
```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```



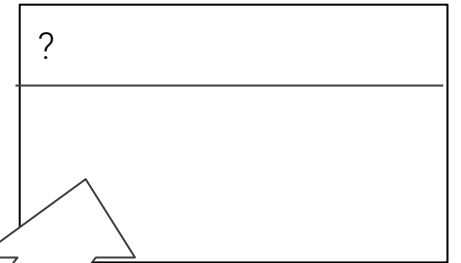
```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```




```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```



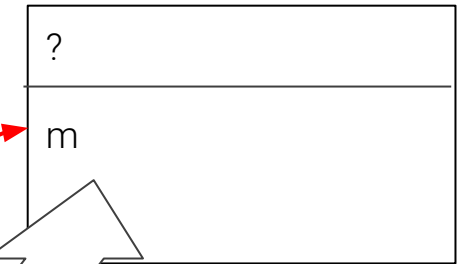
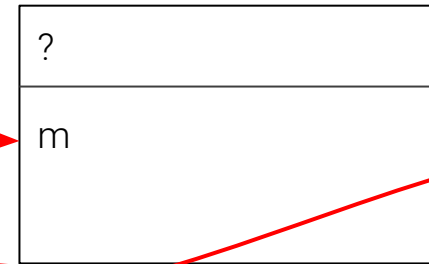
| |
|---|
| ? |
| |



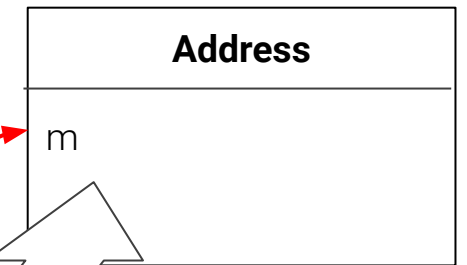
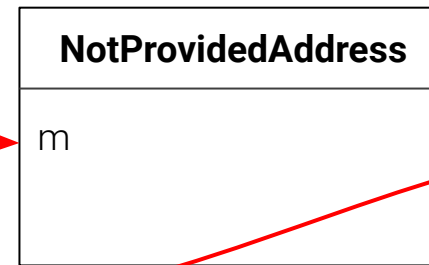
| |
|---|
| ? |
| |



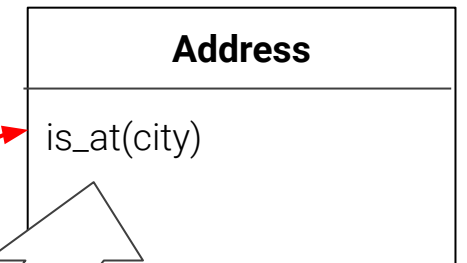
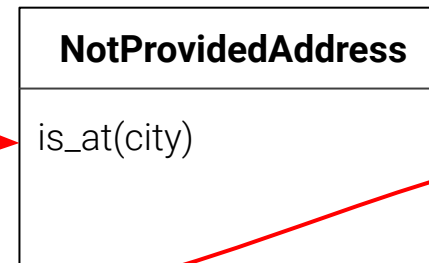
```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```

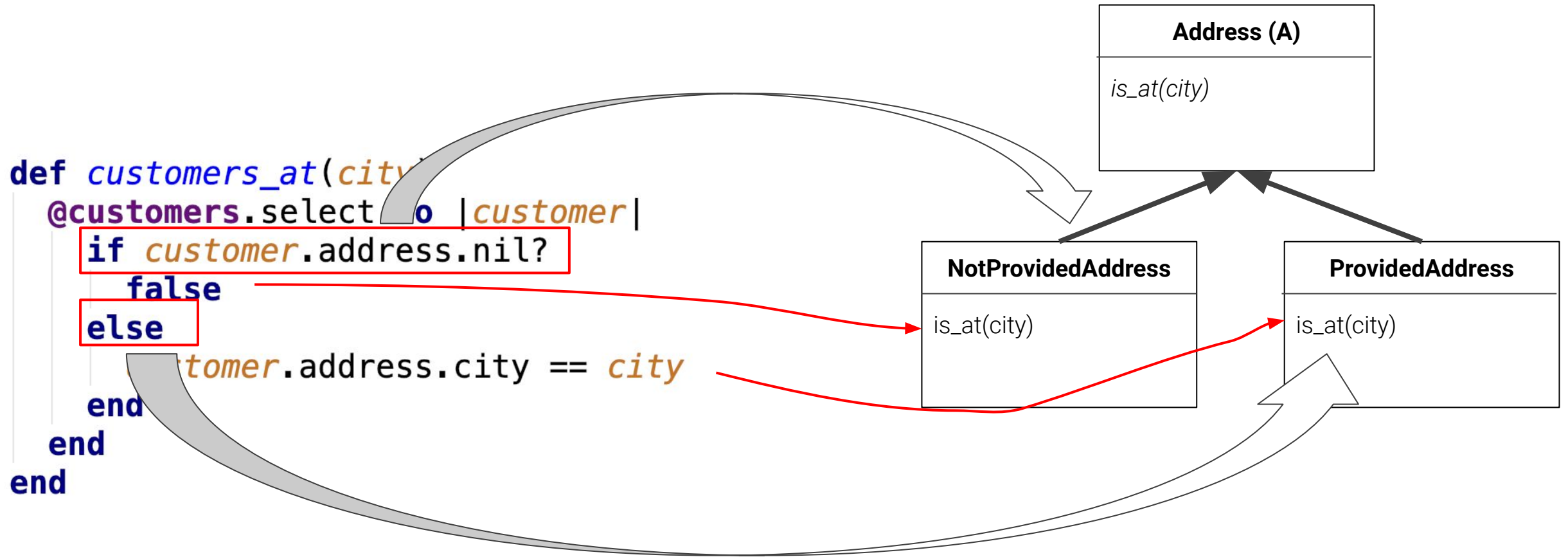


```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```

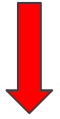


```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```

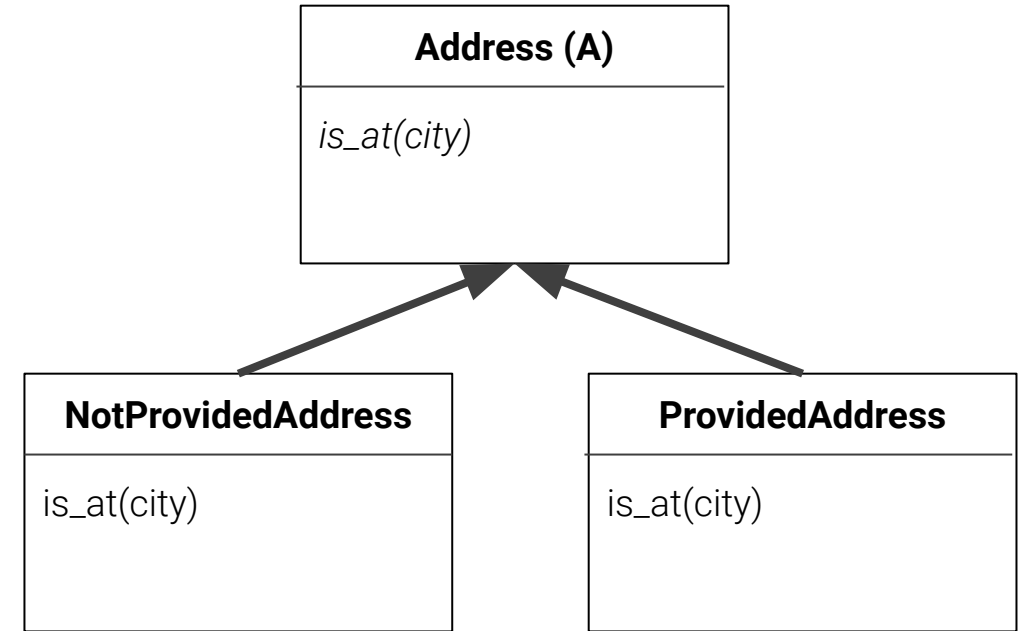




```
def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end
```



```
def customers_at(city)
  @customers.select do |customer|
    customer.address.is_at city
  end
end
```



The Null Object Pattern

Bobby Woolf

Knowledge Systems Corp.

4001 Weston Pkwy, Cary, NC 27513-2303

919-677-1119 x541, bwoolf@ksccary.com

NULL OBJECT**Object Structural**

Intent

Provide a surrogate for another object that shares the same interface but does nothing. The Null Object encapsulates the implementation decisions of how to “do nothing” and hides those details from its collaborators.



```

def customers_at(city)
  @customers.select do |customer|
    if customer.address.nil?
      false
    else
      customer.address.city == city
    end
  end
end

```



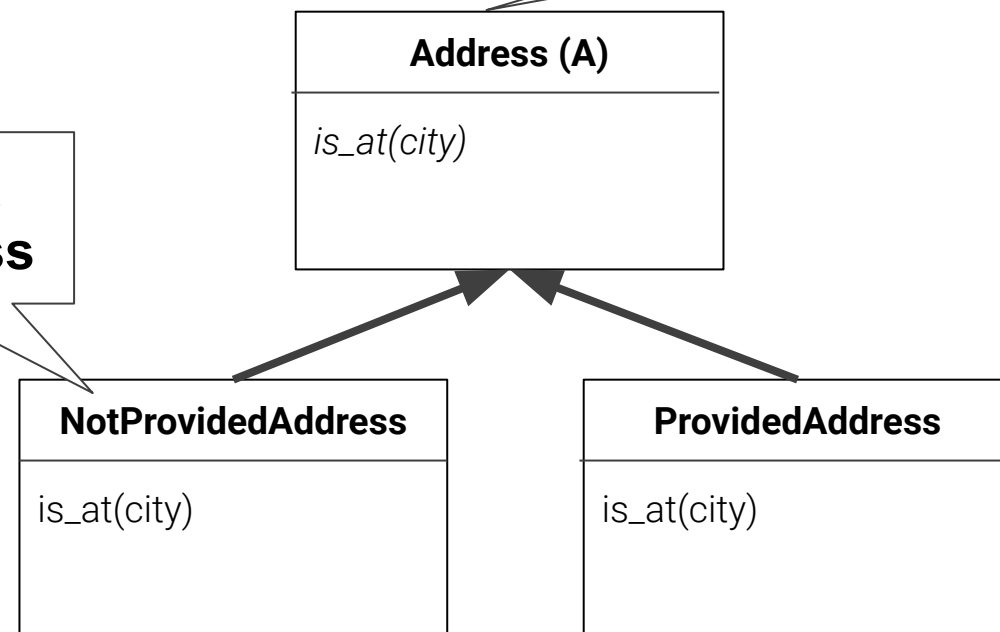
```

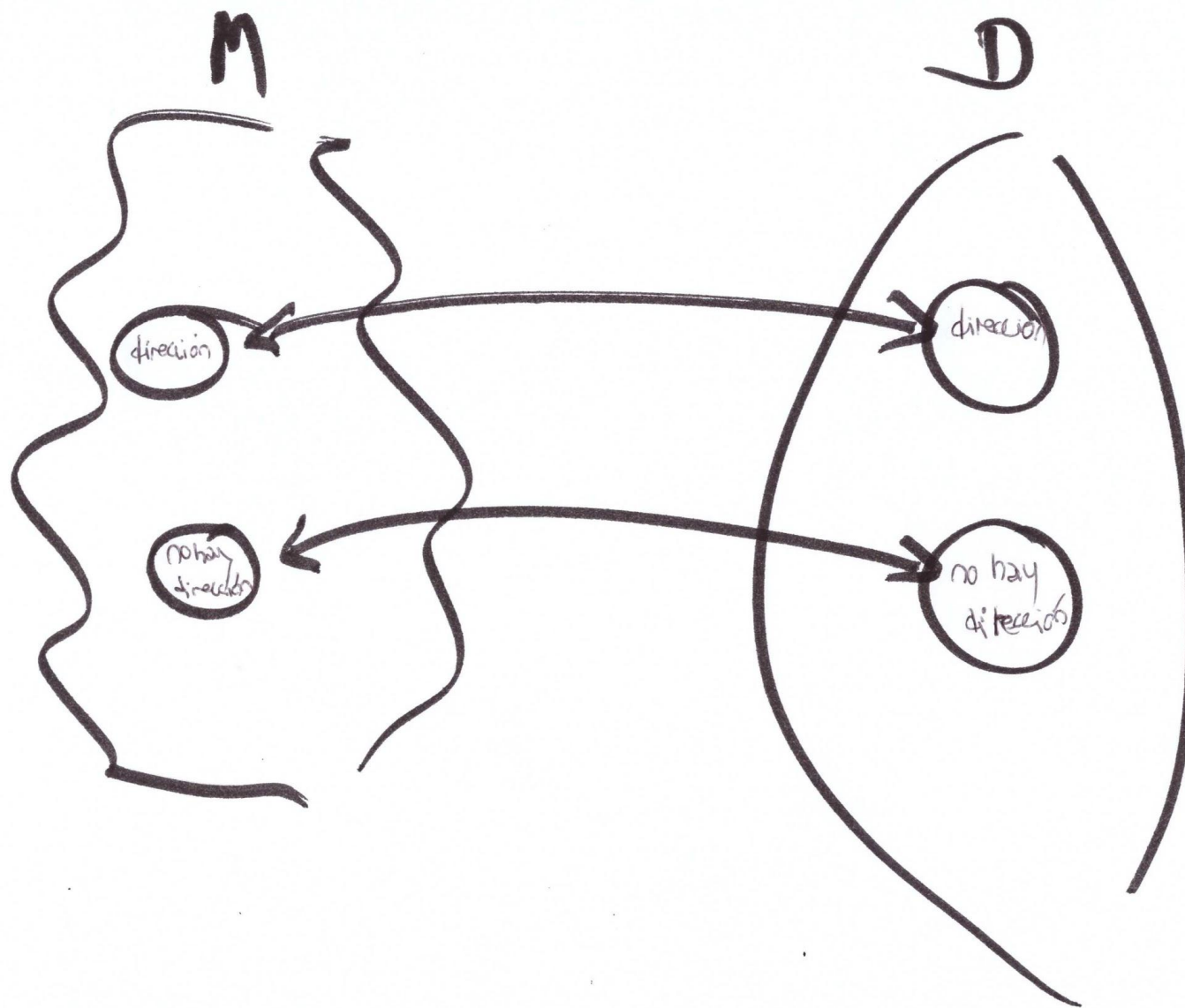
def customers_at(city)
  @customers.select do |customer|
    customer.address.is_at city
  end
end

```

No se llama
NullAddress

No se llama
AbstractAddress





Implementemoslo



Optional/Maybe



No Romper encapsulamiento: Explicit Absent Message



Ruby: safety navigation operator



¿Cuándo usar cada uno?



- Sistema nuevo y se que puede ser “null”
 - Leng. Dinámicamente tipado:
 - i. Explicit Absent Message
 - ii. Null Object
 - Leng. Estáticamente tipado:
 - i. Optional
 - ii. Null Object



- Sistema existente, no podía ser “null” y ahora si
 - Leng. Dinámicamente tipado:
 - i. Null Object
 - ii. Explicit Absent Message
 - Leng. Estáticamente tipado:
 - i. Null Object
 - ii. Optional



- Sistema existente, se rompió encapsulamiento y hay chequeo por null en todos lados
 - Leng. Dinámicamente tipado:
 - i. Explicit Absent Message y a arreglar todo!
 - Leng. Estáticamente tipado:
 - i. Optional y a arreglar todo!



Conclusiones



H6: **No usar** *null/nil*

Tip: Representar la nada especializada por con null object o encapsular que puede ser null con Optional o Explicit Absent Message





Diseño ¡a la gorra!

¡Bienvenidos!

Durante esta serie de Webinars exploraremos *qué* significa **Diseñar Software con Objetos** y *cómo* lo podemos hacer cada vez mejor.

Trataremos muchos temas que irán desde cuestiones filosóficas como qué significa Diseñar en nuestra profesión y dónde está expresado ese Diseño, pasando por consejos y heurísticas para diseñar "mejor" y terminado con ejemplos concretos de cómo aplicar esas heurísticas en la *vida real*.

Los webinars son "*language agnostic*", o sea que no dependen de un lenguaje de programación en particular, aunque los ejemplos que usaremos estarán hechos principalmente en **Java**, **JavaScript**, **Ruby**, **Python** y mi querido **Smalltalk** cuando amerite 😊.

Te esperamos todos los Martes a las 19 Hrs GMT-3 a partir del Martes 11 de Agosto de 2020. Para poder participar tenes que registrarte [acá](#).

Todo el código y presentaciones estarán disponibles para que lo puedan usar y consultar en cualquier momento [acá](#).

¡Trae ganas de aprender y pasarla bien!

¿Por qué a la Gorra?

Al igual que cuando Diseñamos Software está bueno usar una **Metáfora** para entender qué estamos modelando, en este caso usamos una metáfora para explicar cómo *financiaremos*

Donaciones



\$100 - Casi una 🍷

Pagar

\$250 - Una buena 🍺

Pagar

\$500 - Menos que 🍔 + 🍷

Pagar



Donate



¿Querés donar un monto variado o en otra plataforma?



Dictado por:
Hernán Wilkinson

<https://alagorra.10pines.com>

Muchas gracias





10 Pines

Creative Software Development



10pines.com



info@10pines.com



+54 (011) 6091-3125 / 4893-2057



Av. Leandro N. Alem 896 6° - Bs. As. - Argentina



@10pines