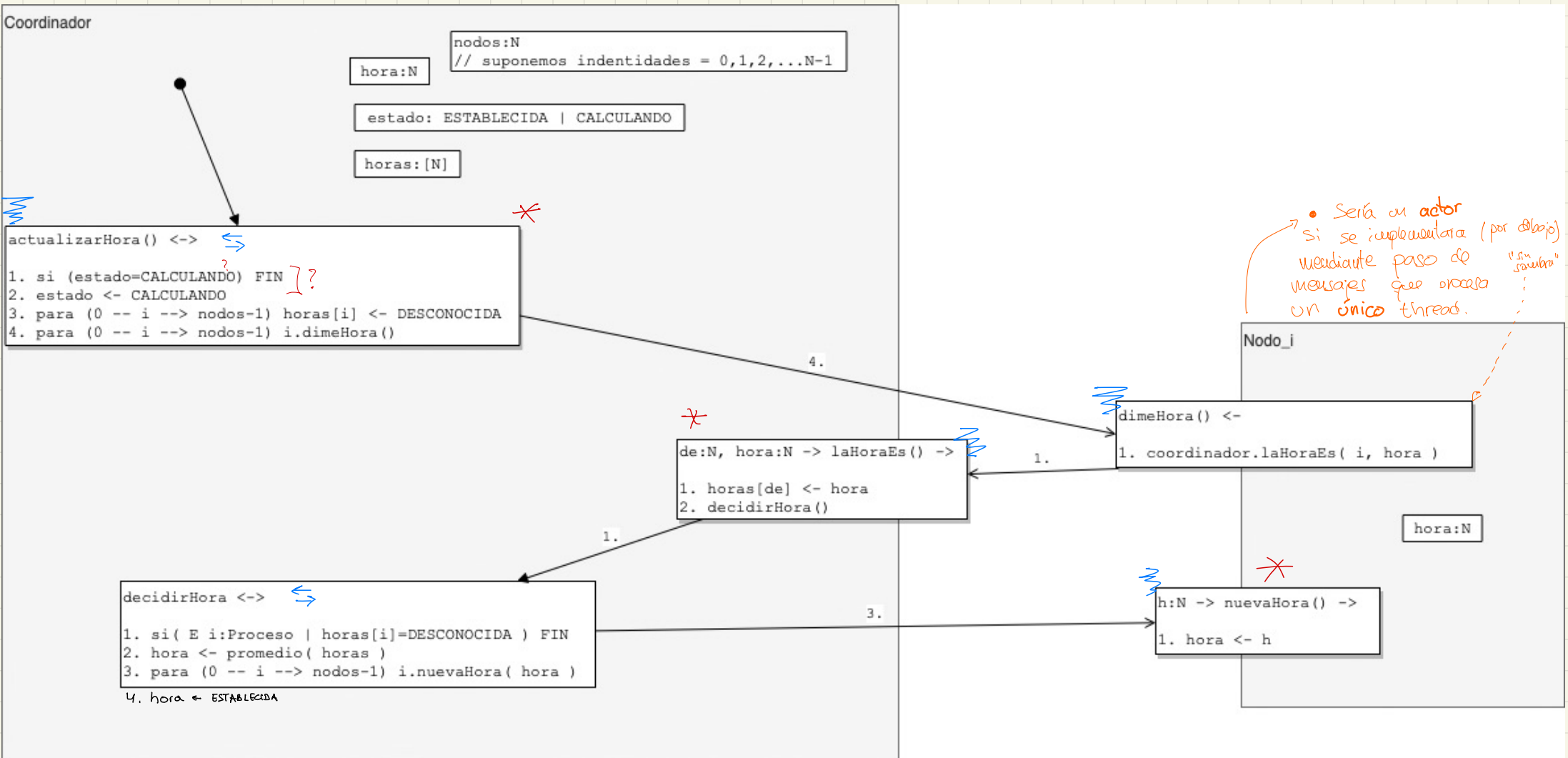


Notación

diseño lógico (con funciones asíncronas)

Algoritmo Berkeley (Gusella y Zatti, 1989)

(al menos, las remotas)

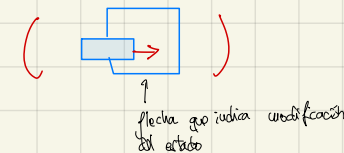


• Sería un actor si se implementara (por ejemplo) mediante paso de mensajes que procesa un único thread.

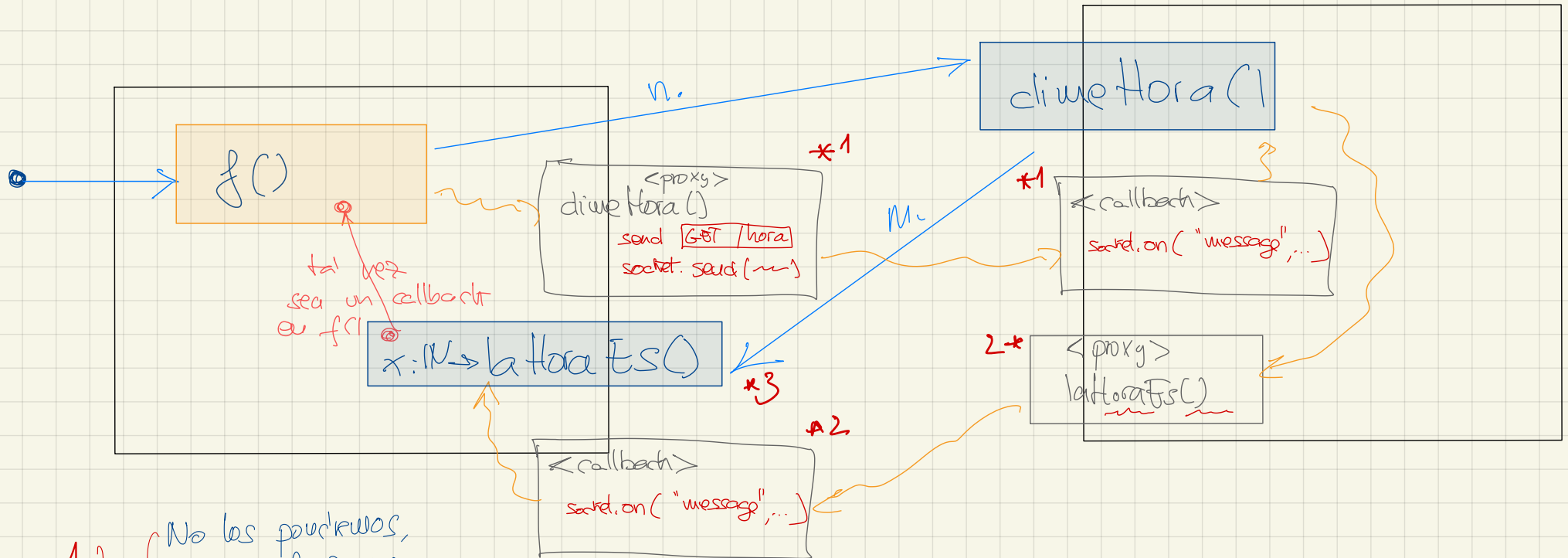
* Vigilar/Revisar métodos que cambien variables globales (del objeto o del proceso)

por si hay ¡CONDICIONES DE CARRERA!

Ver transparencia anterior: ahí se ve más claro los problemas



Sobre el diseño y "la realidad"



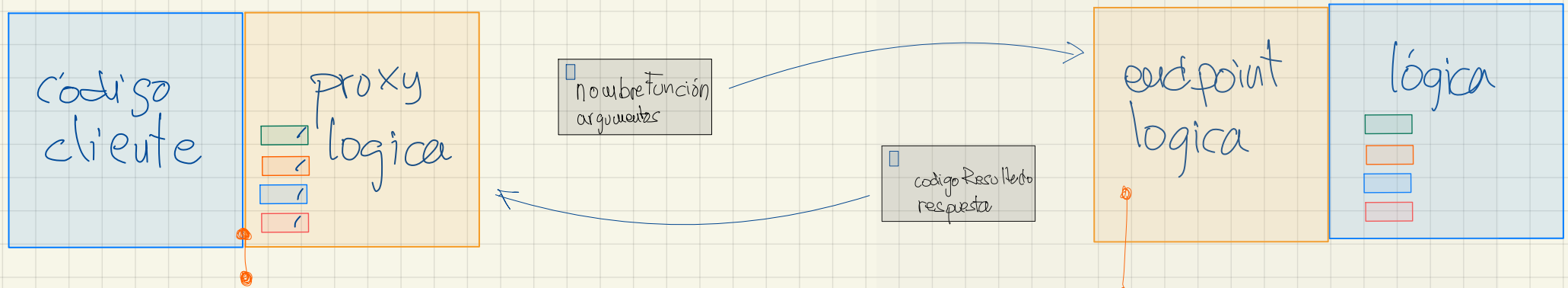
*1
*2
*3 } No los poníamos,
en general porque
se dan por sentados.

Pero: Si poníamos *3 cuando estamos explicando la interacción/algoritmo porque nos interesa señalar que siempre es asíncrono.

Si no ponemos *3 entonces

clienteHora()

Sobre el diseño y "la realidad" : proxies y stubs (evitar código repetido) Posibilidad ①



llamar a método

opciones

```

1 proxyLogica.método(argumentos)
  ↑ * método en el objeto proxyLogica
  para capturar la llamada
  para enviar la petición *
2 proxyLogica.llamar("método", argumentos)
  ← si lo anterior no lo permite el lenguaje
  
```

recibir (petición)

```

try {
  nombreFunción ← petición.nombreFunción
  argumentos ← decodifica petición.argumentos
  resultado ← logica[nombreFunción](argumentos)
  enviar [ OK
          resultado
        ]
} catch (err) {
  enviar [ ERROR
          err
        ]
}
  
```

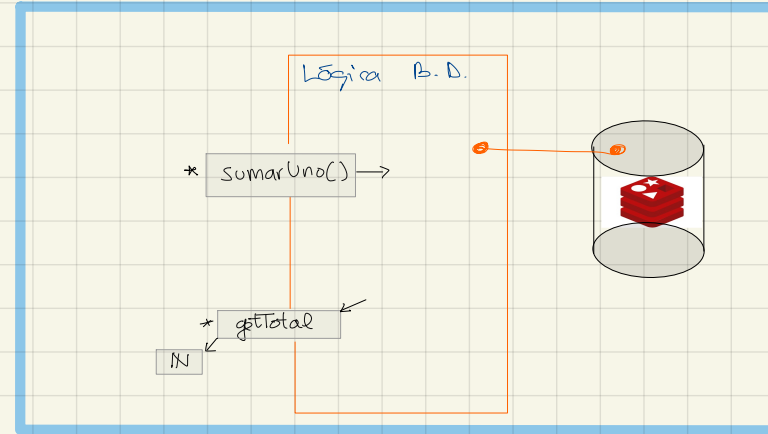
Soportado por el lenguaje (Js) o array asociativo

* Ejemplos / 00_EcmaScript / 07.2 - Proxy Decorador / Missed Calls

Ejemplo separación lógica/comunicación: proxies/stubs (endpoints) (Posibilidad 1)

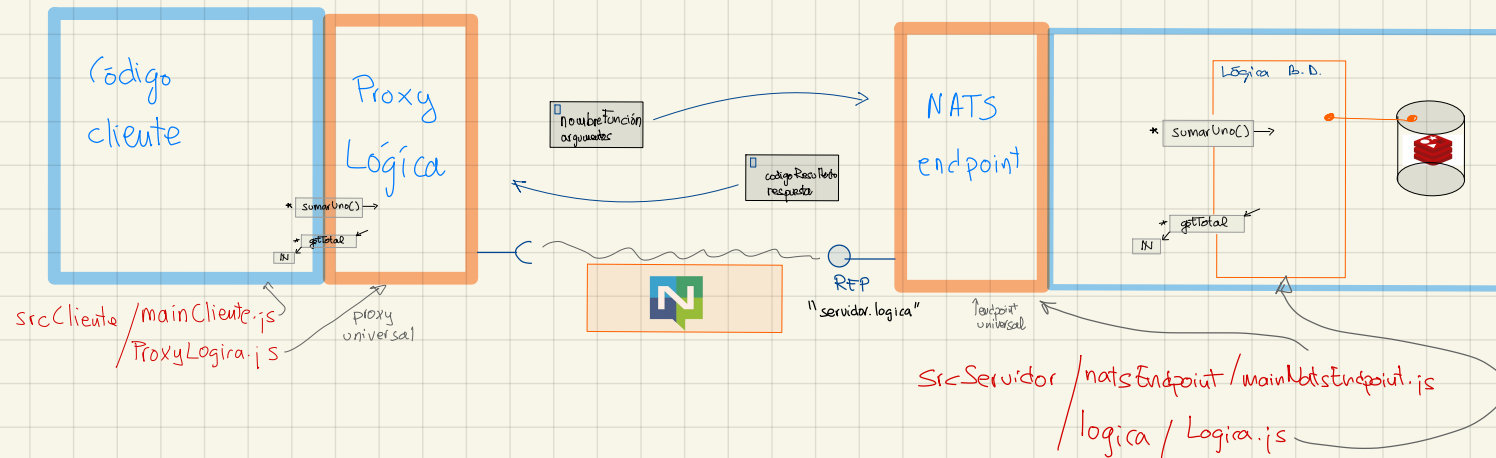
Ejemplo
separación
lógica / comunicación

Diseño



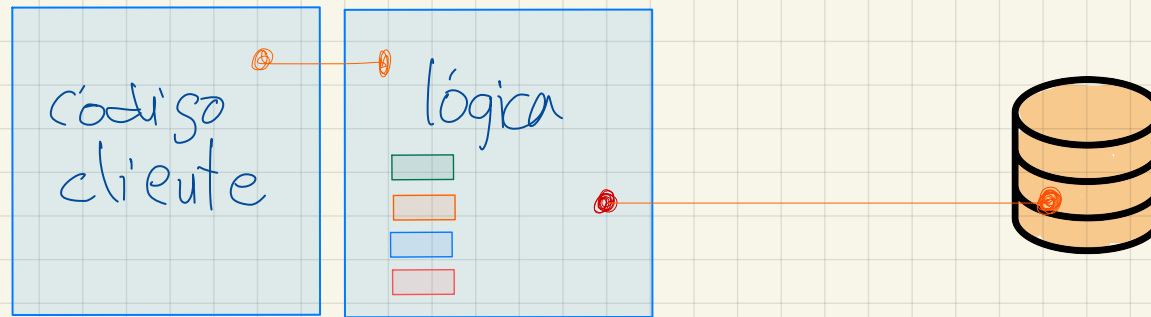
→ 04.3 - NodeJS - NATS - Redis

Realidad



Sobre el diseño y "la realidad"

Posibilidad ②

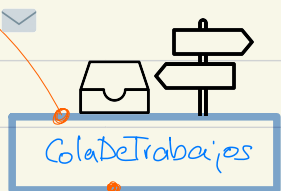
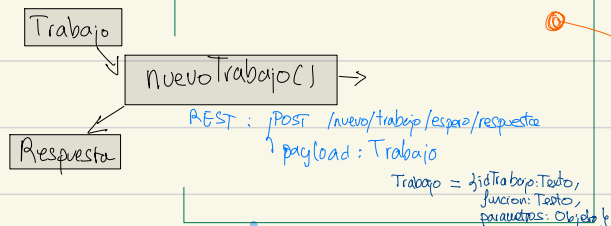


"FAT cliente"

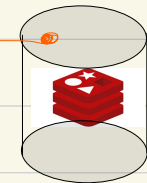
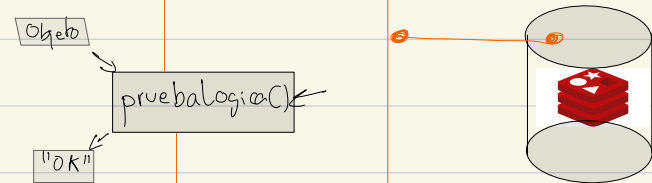
Posibilidad 2



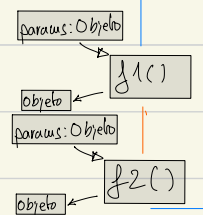
Logica Frontend



Logica B.D.



Logica Worker

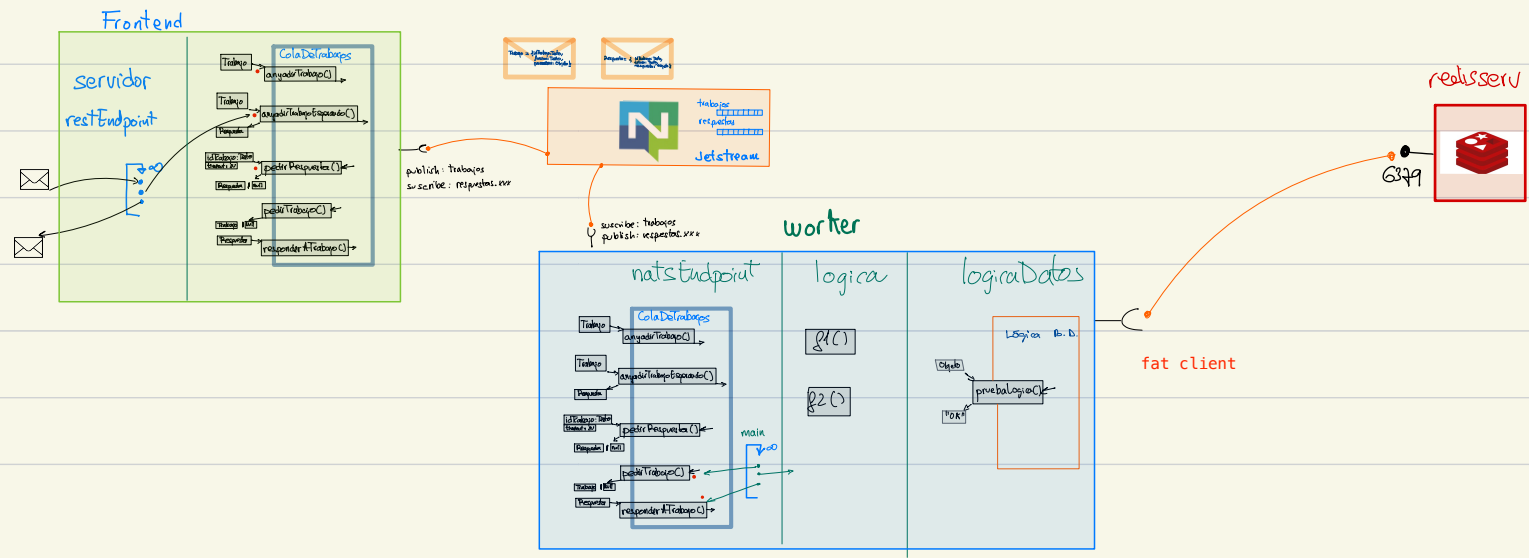


En realidad, el interfaz de la lógica del worker, sería el interfaz del sistema.

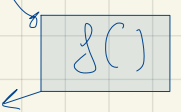
Versión 2 (quedaria Nats - jetstream)
Ejemplos Básicos
21 - Proyecto TaaS Full CC-2021
version-1

Diseño
PUSH-PULL
asignatura CC 2021

Infraestructura / Implementación



Notación "a bajo nivel"

- A "bajo nivel" = nivel de implementación
 (porque  es a nivel lógico = "alto nivel")
 necesitamos especificar

- = si usamos sockets : cómo están conectados → diagrama de conexión
- = los mensajes → entre sockets
 → de API REST
 → entre procesos (ej: "actor model")