Programación Reactiva from Zero to (Almost) Hero

Modelo Reactivo

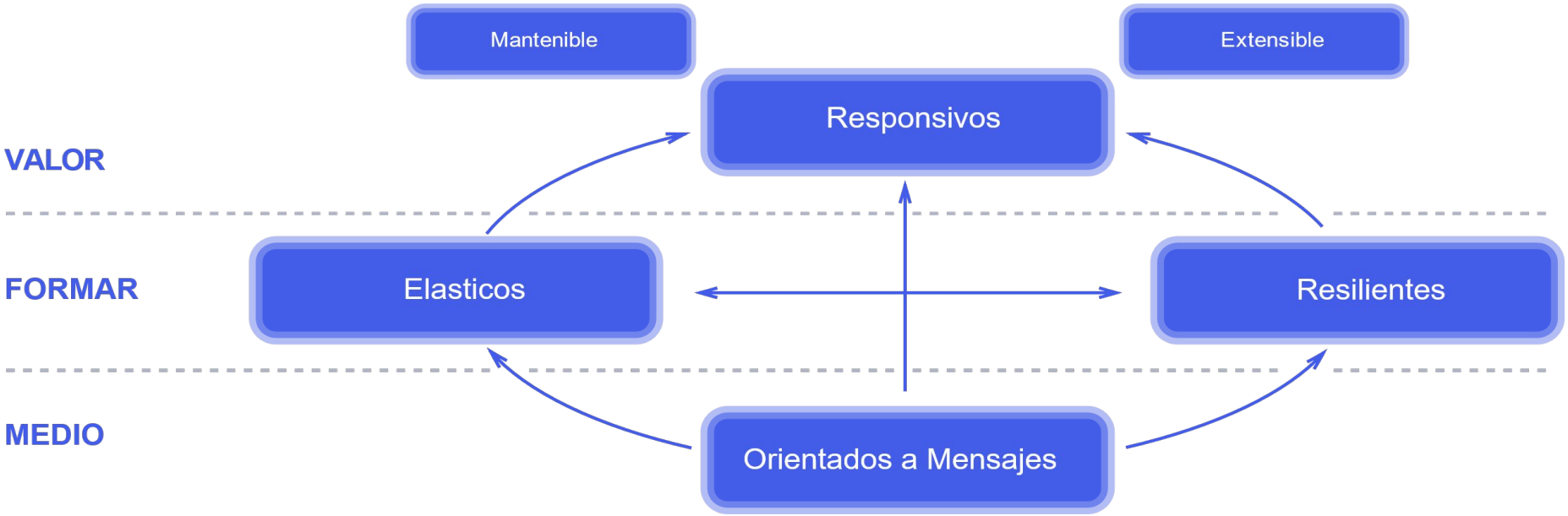
Paradigma de programación ***asíncrona***, ***concurrente*** (y ***distribuida***) que surge como alternativa a las limitaciones del modelo de concurrencia tradicional basado en hilos

- Estado Inmutable compartido
- Deadlocks
- Race Conditions

Modelo Reactivo

2013 -> **manifiesto reactivo** -> aplicaciones concurrentes, asíncronas y distribuidas - > modelos de actores **(Akka)**

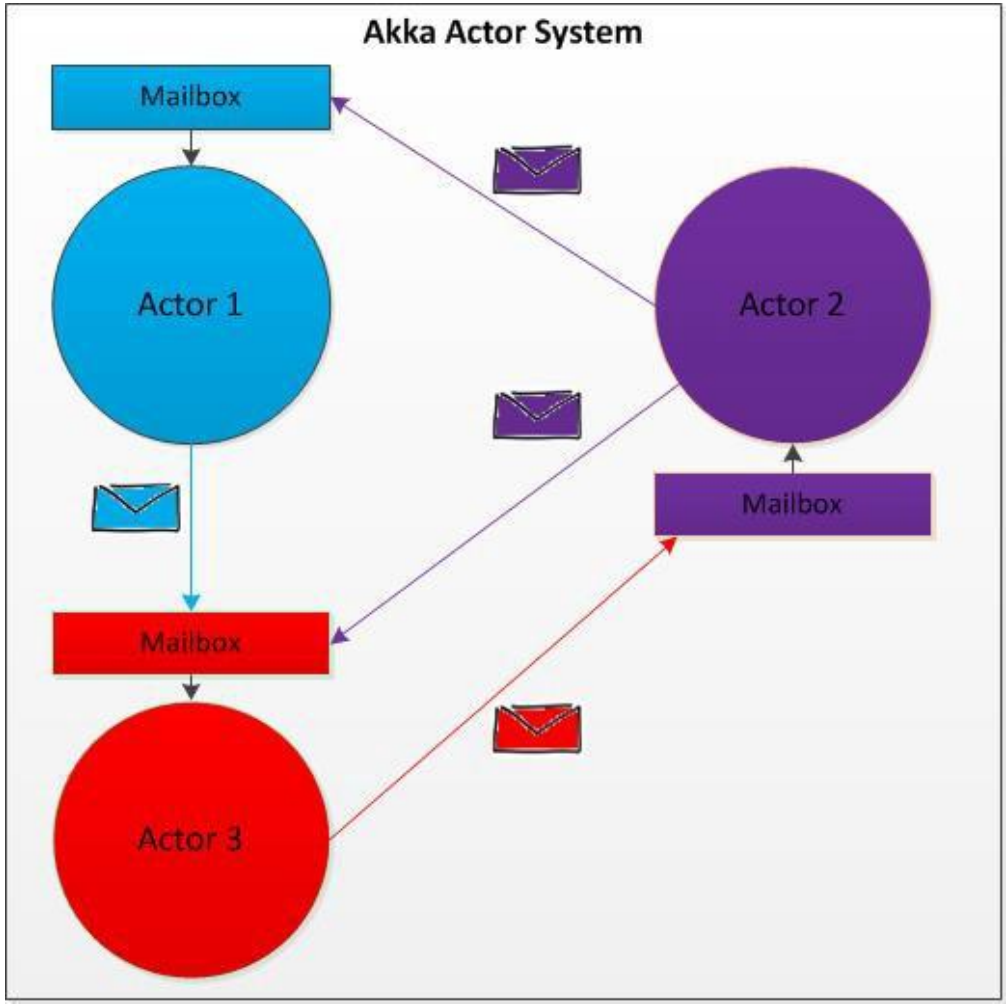
Reactive Manifesto



Reactive Traits

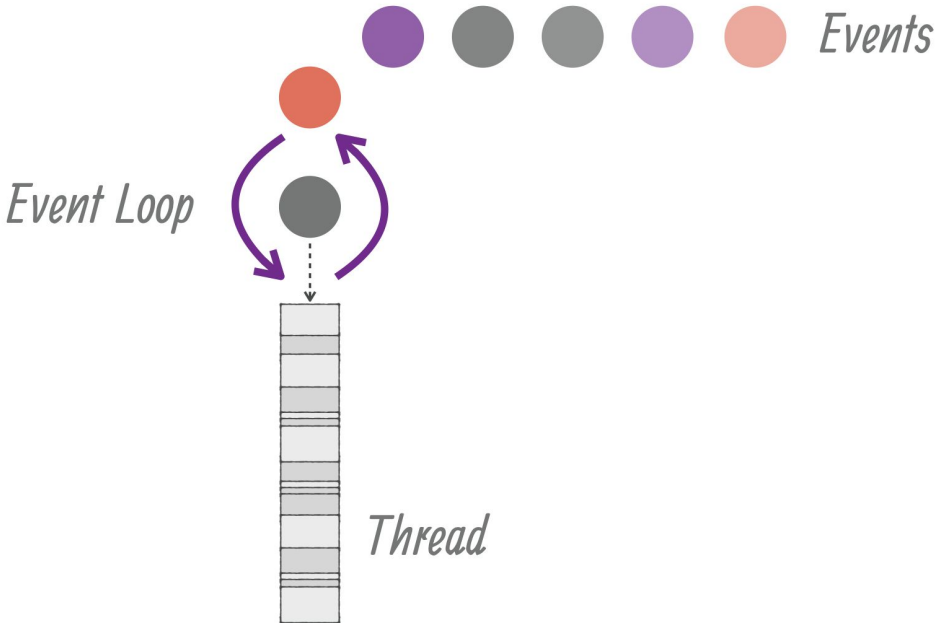
- **Responsive** -> Reacciona al usuario -> El sistema responde siempre de *“manera amigable”* -> tiempos de respuesta
- **Elastic** -> Reacciona ante la carga -> aumenta o disminuye recursos para atender peticiones -> no puntos de contención -> 0 cuellos de botella -> ***Única Responsabilidad***
- **Resilent** -> Reacciona ante el fallo -> *replicación, contención, aislamiento, delegación* -> el sistema afectado es capaz de fallar y recuperarse sin afectar al resto.
- **Message Driven** -> Reacciona a Eventos -> *Asincronía, desacoplamiento* -> delegación de errores como mensaje -> gestion de carga, control del flujo en *Colas de Sistema* -> Mecanismo de ***Backpressure***

Implementaciones



Implementaciones

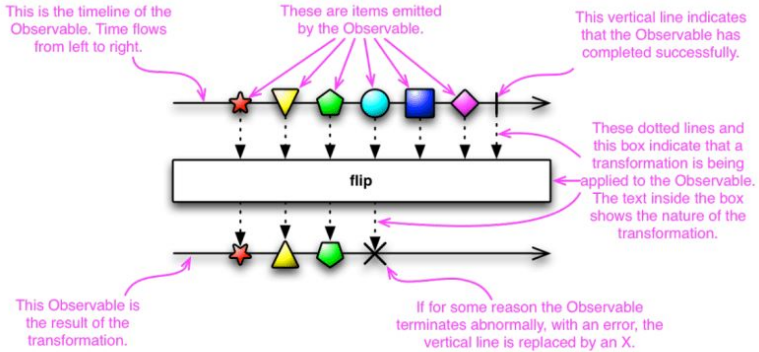
VERT.X



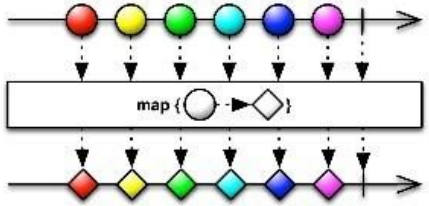
Implementaciones

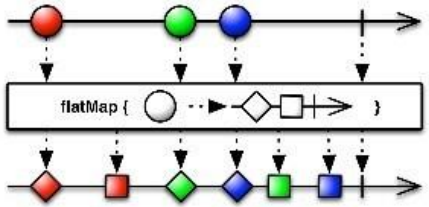
RxJava

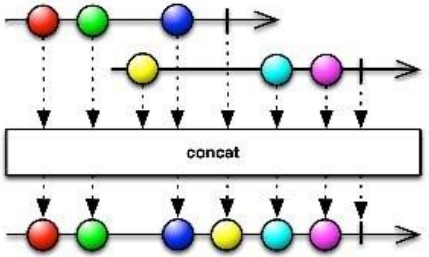
Marble Diagrams

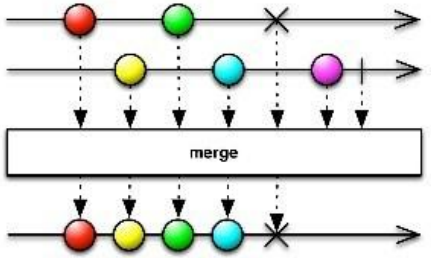


RxJava operations as marble diagrams

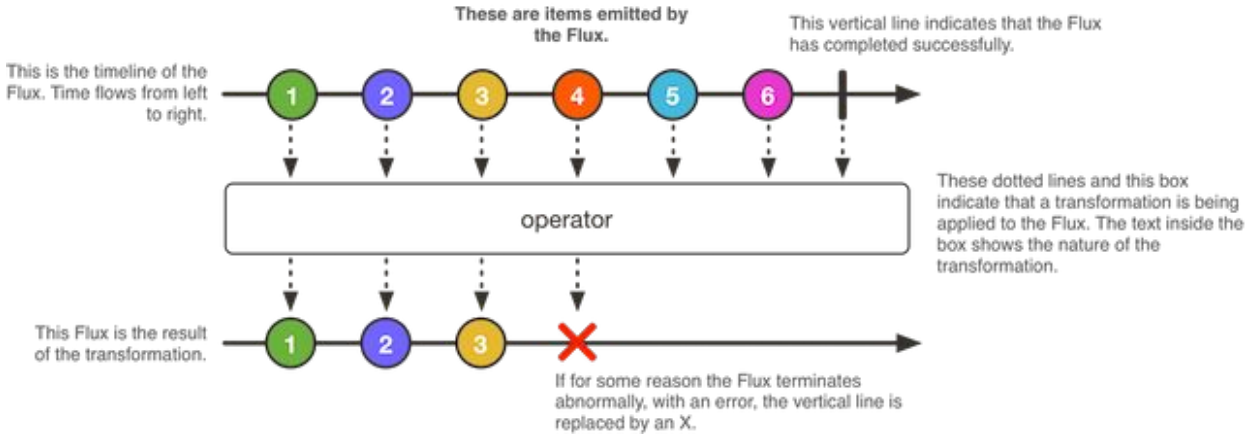


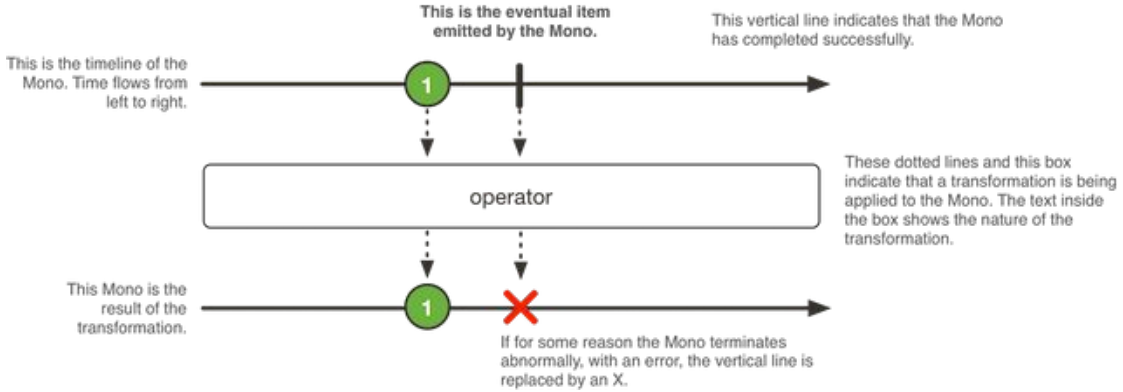






Implementaciones





Qué haremos

Place Order

HTTP Request no Bloqueante (MONO)

Post

/Orders

Post (sin body) que desencadena la creación del pedido (Order), esto implica guardar el documento en Mongo, y notificar al resto de actores.

Evento

Topic: orders

Evento de creación del pedido. Estado **ORDERED**

Será escuchado por ***Kitchen*** que actualizará su estado a **READY**

Kitchen

Orders

Kafka

Order
+orderID: String (UUID)
+orderStatus: ENUM :
- ORDERED
- READY
- DELIVERED
- ERROR

/orders

Save Order

Save Order

MongoDB

OrderEvent (JSON STRING)
+orderID: String (UUID)
+orderStatus: ENUM :
- ORDERED
- READY
- DELIVERED
- ERROR

Evento

Topic: orderStatus

Evento de Actualización del pedido.

Escuchado por **Order** que ejecutará el comando **SaveOrder** para guardar cada uno de los cambios de estados del pedido con un patrón ***Event Sourcing.***

Evento

Topic: orderStatus

Evento de Actualización del pedido.

Producido por **Kitchen** al actualizar el pedido a estado **READY**

Escuchado por **Delivery,** que lo usará para cambiar el estado a **DELIVERED**

Kitchen

Get Orders

SSE (Server Sent Event)

Verbo: GET

/Orders

Continous Query a MongoDB para traer todos los documentos Order, que contendrán los cambios de estado de estas.
Implementado con Flux sobre el repositorio reactivo de Mongo.

OrderDocument
+documentId {@id): String (UUID)
+orderID: String (UUID)
+orderStatus: ENUM :
- ORDERED
- READY
- DELIVERED
- ERROR

https://github.com/ogomezso/event-deliveroo

Que usaremos