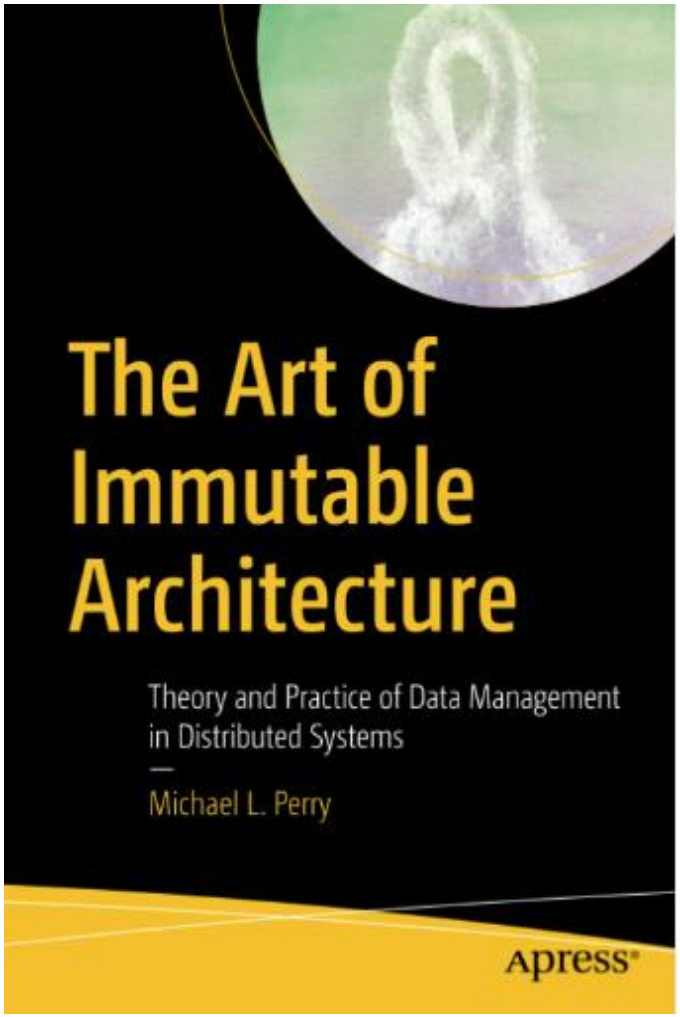
Arquitectura inmutable

Saúl Tuñón Fernández – UO277490
Miguel González Navarro – UO282337
José Jiménez García – UO276008

ASW – EII 2022-2023

¿Infraestructura inmutable?



NO!

Arquitectura inmutable

≠

Infraestructrura inmutable

Programación funcional

"La programación funcional trata de minimizar el cambio del estado y maximizar la inmutabilidad de los datos".

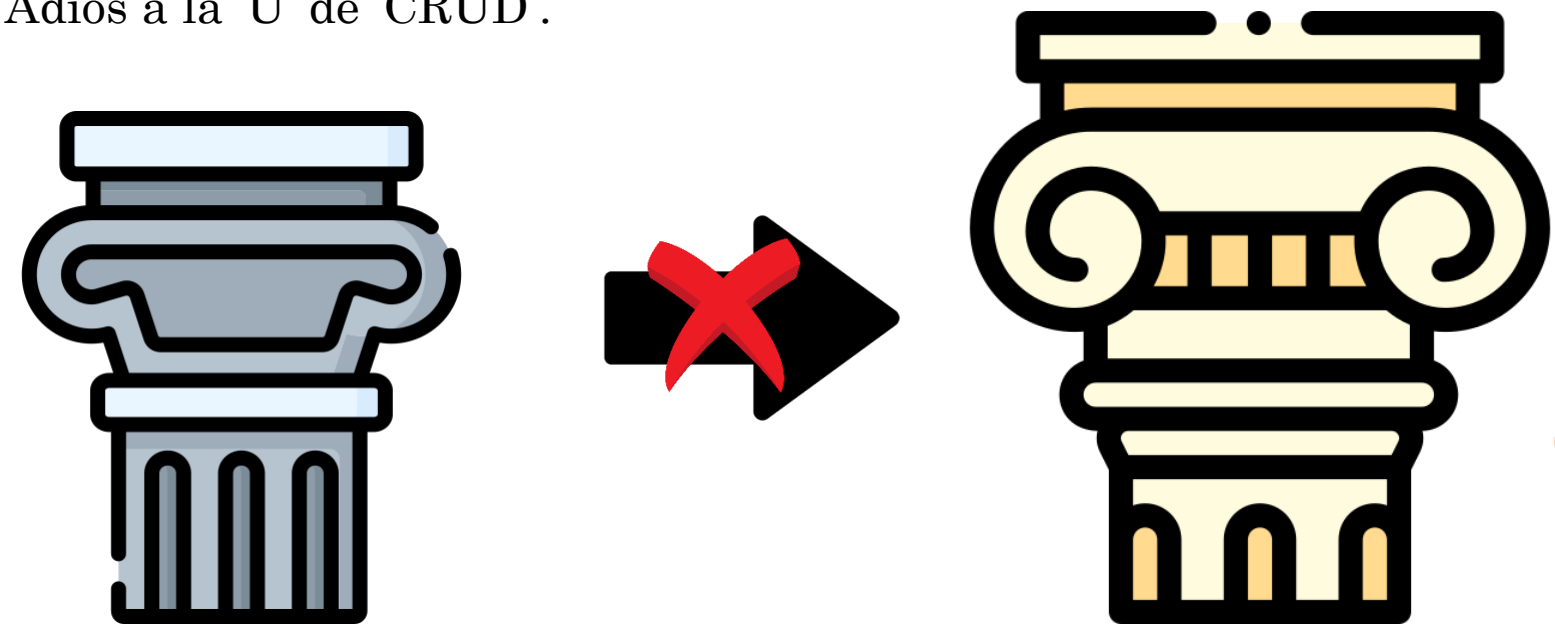
Michael Fogus -
Functional Javascript

Objetos inmutables

Los objetos no cambian su estado después de su creación

Sus propiedades y valores no pueden ser alterados después de la inicialización.

Adiós a la ‘U’ de ‘CRUD’.



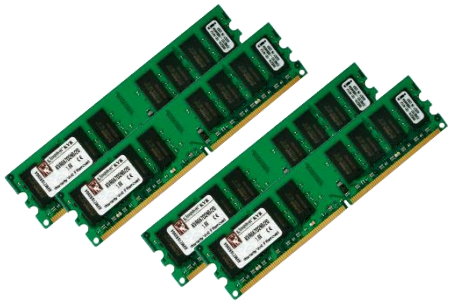
ASPECTOS GENERALES

VENTAJAS

- Razona sobre sistemas
- Seguridad
- Concurrencia
- Código legible
- Escalabilidad

INCONVENIENTES

- Complejidad
- Dificultad para trabajar con estructuras de datos complejas
- Consumo de memoria

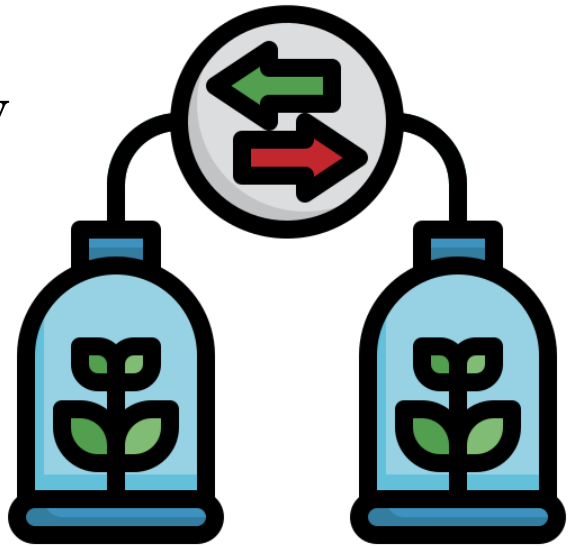


¿Cómo cambiamos los objetos?

COPIA ➡ MODIFICACIÓN ➡ REEMPLAZO

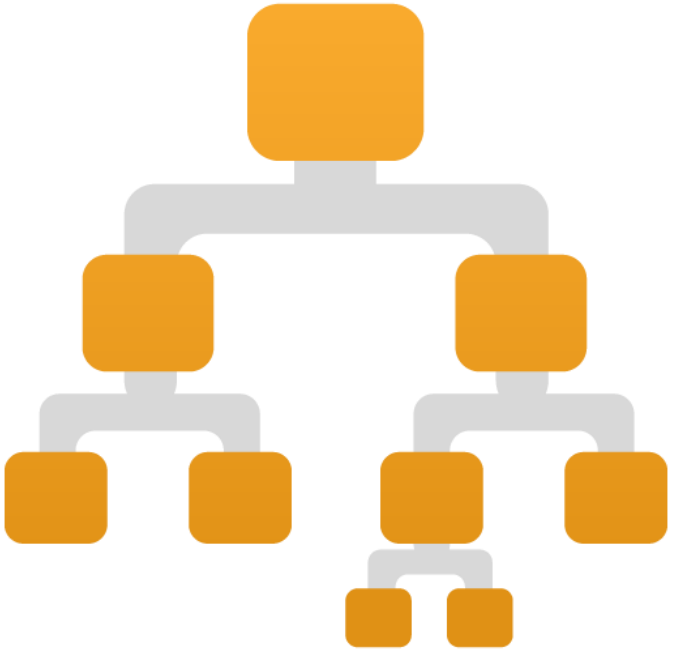
Patrón: Inmutabilidad en cascada

- Inmutabilidad en cascada
  - Crear objetos inmutables a partir de otros
  - Mejora la coherencia, integridad y eficiencia

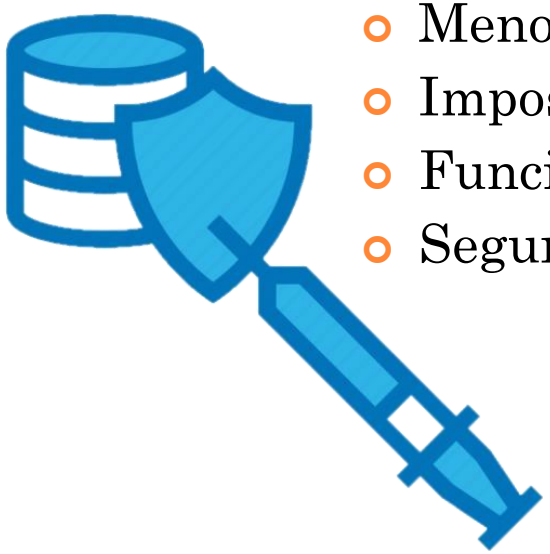


Patrón: Cambio estructural

- Cambio estructural
  - Actualización de objeto con muchas dependencias
  - Nueva versión del común y actualización (sin recrear) del resto
  - Árbol o grafo



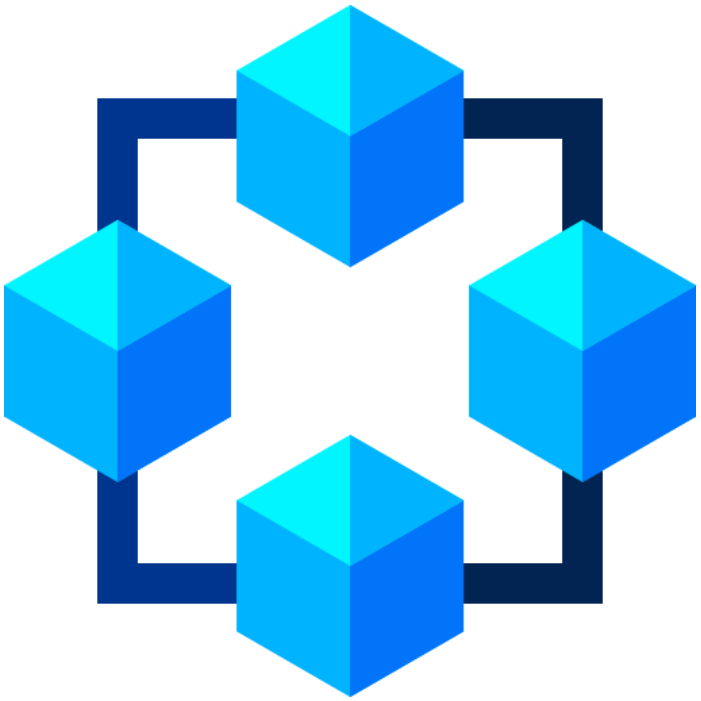
Más Seguridad 🔒



- Menor superficie de ataque
- Imposibilidad de inyectar datos
- Funciones puras y separación lógica
- Seguridad a nivel de datos, integridad

Blockchain

- Inmutabilidad como requisito crítico
- Bloques de datos inmutables



Cada transacción se registra en un bloque de datos que no se puede modificar

Rendimiento y recuperación

- Mejora la capacidad de recuperación, volviendo a un estado anterior al problema.
- Los errores no se pueden propagar.

- Los objetos pueden almacenarse en el caché de manera segura