

# Episode 447: Michael Perry on Immutable Architecture

ALBA GUERRERO GARCÍA – UO266007

PAULA SUÁREZ PRIETO – UO269745

## Introducción

---

Michael Perry es un arquitecto software estadounidense y autor del libro “The Art of Immutable Architecture”. Este libro explora la idea de una arquitectura inmutable en el desarrollo de aplicaciones software.

Michael ha cultivado una técnica de análisis y diseño de sistemas centrándose en hechos históricos inmutables. Debido a los problemas comunes encontrados en la comunidad, ha creado un proyecto de código abierto como base de motor de correspondencia punto a punto que se puede encontrar en el siguiente [enlace](#). Este proyecto se encuentra con versiones antiguas del framework .NET por lo que ha lanzado la versión de JavaScript llamada [Jinaga](#).

En el libro mencionado anteriormente, Michael se centra en los problemas que la inmutabilidad pretende resolver y proporciona técnicas para poder razonar sobre tus sistemas, analizando los problemas usando modelado histórico.

## ¿Qué es la arquitectura inmutable?

---

La arquitectura inmutable es una restricción autoimpuesta sobre la modificación y eliminación de datos. Esto nos ayuda a resolver problemas comunes en el desarrollo software como la complejidad, la cohesión, la duplicidad de información. La arquitectura inmutable se podría ver como el intercambio de información entre registros no cambiantes, el uso de estos disminuye la complejidad del sistema, ya que al ser inmodificables son fácilmente reproducibles.

Gracias a esto se podrían solventar parte de los problemas de los sistemas distribuidos como la escalabilidad, complejidad, la gestión de infraestructura y despliegue de aplicaciones

## Aspectos que destacar sobre el libro “The Art of Immutable Architecture”

---

En el libro Michael Perry ejemplifica los problemas a los que se ven sometidos los sistemas informáticos, por ejemplo, para ilustrar la dificultad de garantizar la comunicación fiable en un entorno de red no confiable, hace uso de la paradoja de los dos generales.

También menciona varios patrones que resuelven problemas típicos de los sistemas distribuidos como son el patrón de almacenamiento de direcciones de contenido (*Content Addressed Storage*) y el patrón de la bandeja de salida (*Outbox Pattern*).

Hace hincapié en el teorema CAP (*Consistency, Availability and Partition tolerance*) para referirse a los distintos niveles de consistencia que se pueden adoptar y en los algoritmos de consentimiento distribuido como Paxos.

En resumen, la solución a muchos de los problemas que presentan los sistemas distribuidos se basa en desarrollar un sistema basado en hechos históricos con una arquitectura inmutable.

## Beneficios de la arquitectura inmutable

---

Al utilizar una arquitectura inmutable se puede razonar sobre los sistemas. De esta manera se puede entender lo que el sistema va a hacer, aplicar esto a sistemas distribuidos sin los conceptos de arquitectura inmutable y hechos históricos era casi imposible de hacer.

Se pueden entender las restricciones necesarias para llegar a unas propiedades determinadas del sistema y a qué propiedades no se puede llegar.

Michael también da pautas para razonar sobre el grado de consistencia necesario en el sistema y dependiendo de este elegir una arquitectura u otra.

Se pueden realizar aplicaciones con estado, en las que por ejemplo el usuario pueda seguir accediendo a diversas funcionalidades sin conexión.

En resumen, se pueden resolver problemas que normalmente los desarrolladores se plantean como casos excepcionales

## Bibliografía

---

Brown, S. A. (8 de Mayo de 2022). *The Two Generals Problem*. Recuperado el 28 de Marzo de 2023, de Hayden James: <https://haydenjames.io/the-two-generals-problem/>

Chandrakant, K. (20 de Marzo de 2023). *Consensus Algorithms in Distributed Systems*. Recuperado el 28 de Marzo de 2023, de Baeldung: <https://www.baeldung.com/cs/consensus-algorithms-distributed-systems>

*Episode 447: Michael Perry on Immutable Architecture*. (18 de Febrero de 2021). Recuperado el 28 de Marzo de 2023, de Software Engineering Radio: <https://www.se-radio.net/2021/02/episode-447-michael-perry-on-immutable-architecture/>

*Michael Perry*. (s.f.). Recuperado el 28 de Marzo de 2023, de <https://michaelperry.net/>

Ozkaya, M. (8 de Septiembre de 2021). *Outbox Pattern for Microservices Architectures*. Recuperado el 28 de Marzo de 2023, de Medium: <https://medium.com/design-microservices-architecture-with-patterns/outbox-pattern-for-microservices-architectures-1b8648dfaa27>

*What is the CAP theorem?* (s.f.). Recuperado el 28 de Marzo de 2023, de IBM: <https://www.ibm.com/topics/cap-theorem#:~:text=The%20CAP%20theorem%20maintains%20that,%2C%20availability%2C%20and%20partition%20tolerance.>