educación IT
CODE YOUR FUTURE

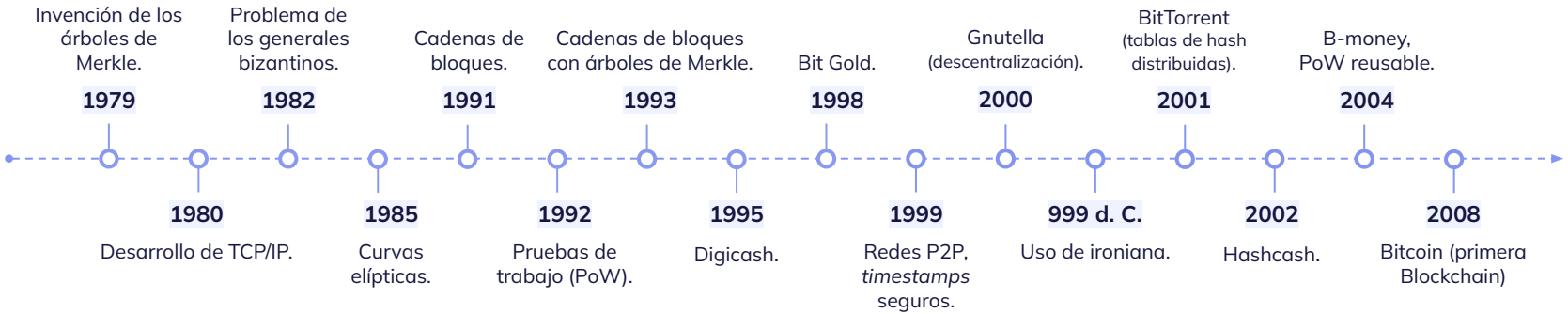
# Criptografía y Blockchain

Módulo 4

# Blockchain

# Historia de Blockchain

## Hitos



# Fundamentos

Una **cadena de bloques** (***Blockchain***) se puede caracterizar como un **sistema distribuido**.

> Un ***sistema distribuido*** es un paradigma de computación donde dos o más nodos trabajan juntos de manera coordinada para cumplir una meta en común enviándose mensajes.

Un ***nodo*** es una entidad individual (persona, computadora) en un sistema distribuido. Todos los nodos del sistema pueden enviar y recibir mensajes entre sí.

**Los nodos pueden ser honestos, defectuosos o maliciosos,** y tienen memoria y capacidad de procesamiento. Un nodo que exhibe un comportamiento arbitrario se conoce como ***nodo bizantino***.

Dos desafíos clave del diseño de un sistema distribuido son la **coordinación entre los nodos y la tolerancia a los fallos**. Incluso si los nodos se vuelven defectuosos o se interrumpe la comunicación en la red, el sistema distribuido debería tolerarlo y continuar trabajando para llegar al resultado deseado.

# Problema de los Generales bizantinos

**Es un experimento mental propuesto por Lamport en 1982.**

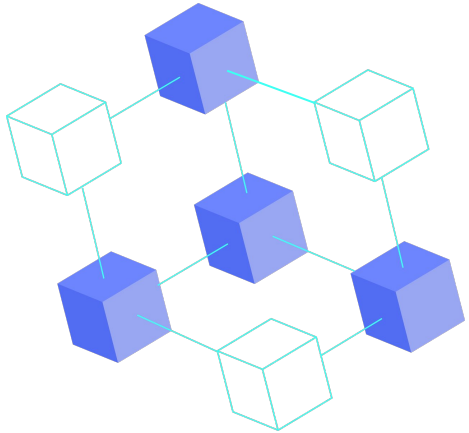
Supongamos un escenario de guerra en el que tenemos un grupo de generales bizantinos que están asediando una ciudad desde distintos lugares y tienen que ponerse de acuerdo para atacar o retirarse de forma coordinada. La única forma de comunicación es mediante mensajeros.

Uno o más de los Generales puede ser un traidor (al resto se les llama *leales*), por lo que su objetivo es conseguir que todos los generales leales no se pongan de acuerdo.

Para ello pueden ofrecer información errónea. Incluso los mensajeros puede ser capturados por la ciudad.

**El problema es hallar un mecanismo de consenso que permita llegar a un acuerdo entre todos los leales.**

Como una analogía de los sistemas distribuidos, **los leales pueden considerarse nodos honestos, los traidores nodos bizantinos** (nodos con comportamiento arbitrario, malicioso o no) y los mensajeros capturados como mensajes demorados o perdidos. Se han presentado diversas soluciones.

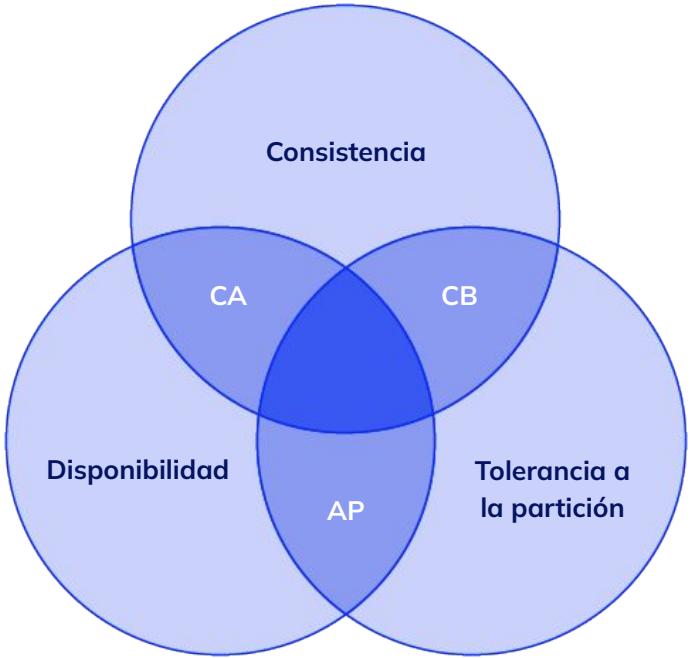


# Teorema CAP

Se ha probado que un sistema distribuido no puede cumplir al mismo tiempo las propiedades deseadas de **consistencia** *(Consistency)*, **disponibilidad** *(Availability)* **y tolerancia a la partición** *(Partition Tolerance)* **simultáneamente**. Este principio se conoce como ***Teorema CAP*** o ***Teorema de Brewer***.

- La **consistencia** asegura que todos los nodos tienen una copia idéntica y actualizada de los datos.
- La **disponibilidad** significa que los nodos están activos, accesibles y aceptando peticiones y respondiendo con datos sin fallas cuando son requeridos.
- La **tolerancia a la partición** significa que si un grupo de nodos son incapaces de comunicarse con otros debido a fallas en la red y/o dichos nodos, el sistema distribuido puede continuar operando correctamente.

## Teorema CAP


Consistencia
CA
CB
Disponibilidad
AP
Tolerancia a
la partición

# *Blockchain*

En una *blockchain*, la consistencia es sacrificada en virtud de la disponibilidad y la tolerancia a la partición (AP). **La consistencia se logra a través de la validación de múltiples nodos a través del tiempo.** A esto se lo denomina ***consistencia eventual***.

En **Bitcoin** se requieren múltiples confirmaciones de transacciones para alcanzar una consistencia eventual. Para lograr este propósito, se introdujo el **proceso de minado**.

El proceso de minado se utiliza para añadir bloques a la *blockchain.* Utiliza el **algoritmo PoW** (***Proof of Work***).

Ningún intento previo de crear monedas digitales anónimas y descentralizadas pudo resolver el problema de prevención del **doble gasto** (usar dos veces el mismo dinero) en un entorno de completa desconfianza. Esto fue resuelto en 2008 por **Satoshi Nakamoto**, que introdujo la **criptomoneda Bitcoin**.

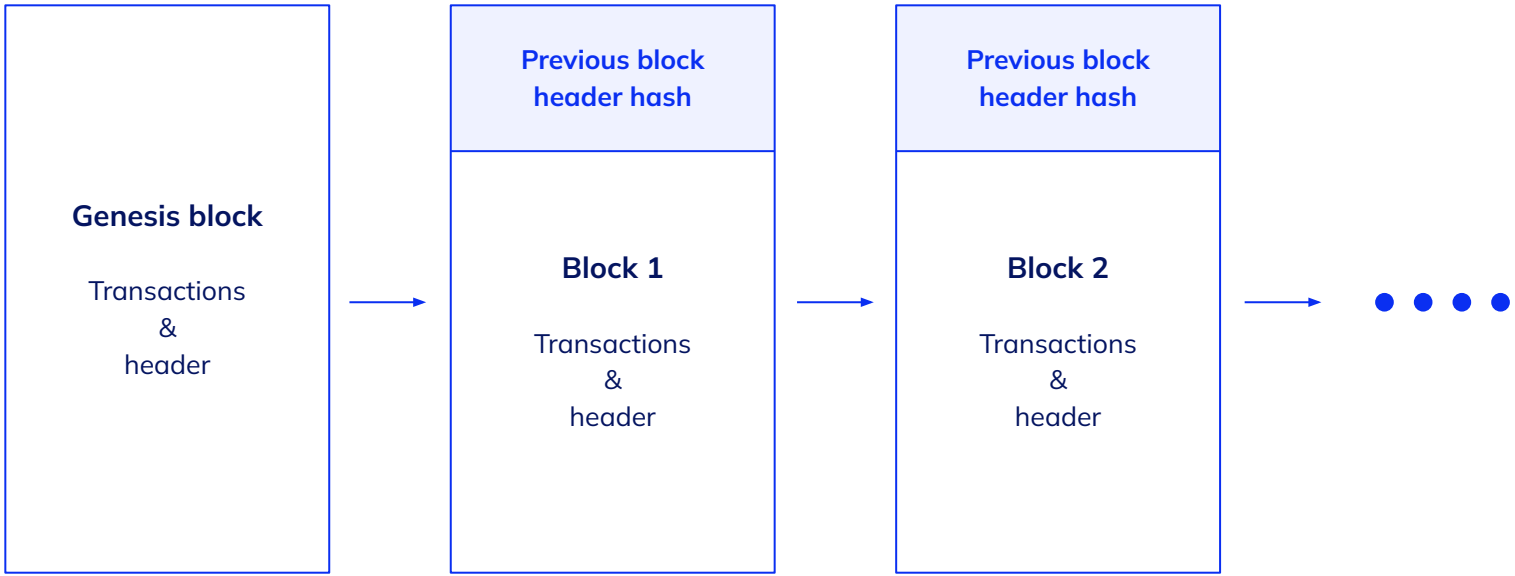
**Definición técnica de *Blockchain***

Es un libro de contabilidad distribuido en una red entre pares que está **criptográficamente seguro,** al que solo se le pueden añadir datos, inmutable (extremadamente difícil de cambiar) y **actualizable solamente vía consenso entre pares.**

# *Blockchain* por capas

| | |
|---|---|
| Aplicaciones | • *Smart contracts.*<br>• Aplicaciones descentralizadas.<br>• Organizaciones autónomas descentralizadas.<br>• Agentes autónomos. |
| Ejecución | • Máquinas virtuales.<br>• Bloques.<br>• Transacciones. |
| Consenso | • Replicación de la máquina de estados.<br>• Consenso basado en pruebas.<br>• Protocolos tradicionales bizantinos tolerantes a fallas. |
| Criptografía | • Criptografía de clave pública.<br>• Firmas digitales.<br>• Funciones de *hash.* |
| P2P | • Protocolos de chismes / protocolos epidémicos.<br>• Protocolos de enrutamiento.<br>• Protocolos de inundaciones. |
| Network | • Internet<br>• TCP/IP |

# Estructura genérica de *Blockchain*



# Estructura de un bloque

Una ***dirección*** es un identificador único usado en una transacción de *blockchain* que identifica a emisores y a receptores. Usualmente son claves públicas o se derivan de ellas.

Una ***transacción*** representa una transferencia de valor de una dirección a otra.

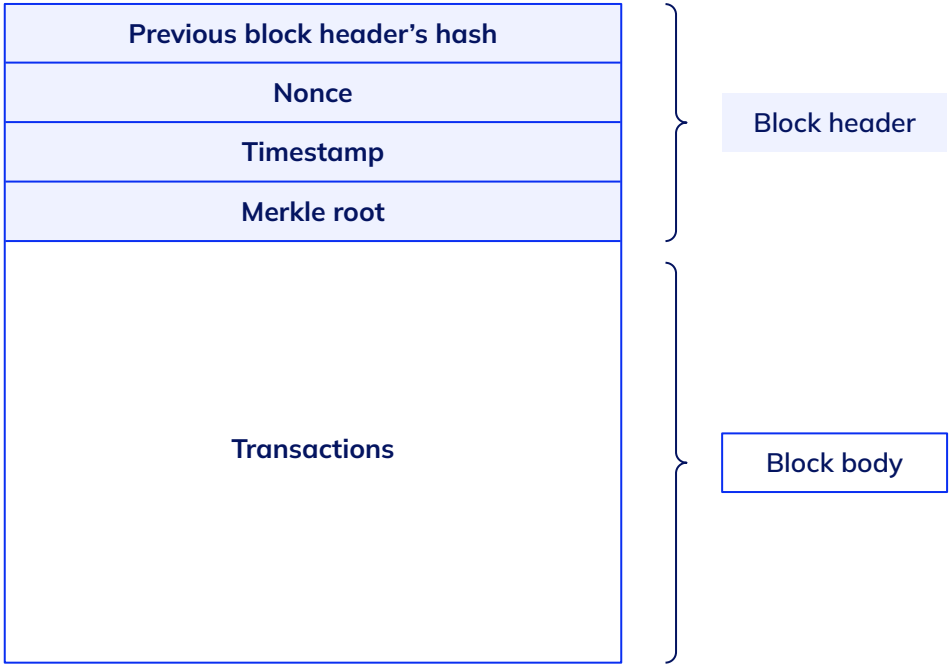
Un ***bloque*** está compuesto de varios elementos:

- múltiples **transacciones**,
- el puntero al ***hash*** (hash del bloque previo),
- el ***timestamp*** *(*tiempo de creación de un bloque) y,
- el ***nonce***.

El **bloque génesis** es el primer bloque. El *timestamp* es el tiempo de creación de un bloque.

La **raíz de Merkle** es el *hash* de todos los nodos en un árbol de Merkle. En un bloque, es el *hash* combinado de todas las transacciones del bloque. Los árboles de Merkle se usan para verificar transacciones.

# Estructura de un bloque



# Tipos de *Blockchain*

## Públicas

Son **abiertas**, y cualquier persona puede participar como nodo. Los participantes pueden ser recompensados o no por su participación.

Todos los usuarios mantienen una copia del libro de contabilidad en sus nodos locales y utilizan un mecanismo de consenso para decidir el estado eventual del libro de contabilidad.

> **Por ejemplo**
>
> Bitcoin, Ethereum.

## Privadas

Están **abiertas solamente para un consorcio o grupo de individuos u organizacione**s quienes decidieron compartir el libro de contabilidad. También se las conoce como *blockchain de consorcios* o *de empresas.*

> **Por ejemplo**
>
> Hyperledger Fabric, Quorum.

### Semi privadas

Una parte de la *blockchain* es **privada y otra parte es pública**. Aún son conceptuales. La parte privada es controlada por un grupo de individuos y la pública está abierta para todos.

### Libros contables autorizados

Los **participantes de la red se conocen y confían unos en otros**. No necesitan un mecanismo de consenso distribuido, las transacciones son verificadas por un conjunto de nodos preseleccionados por una autoridad central, y no necesitan mecanismo de minado.

### Totalmente privadas y propietarias

**No existen en la práctica, son una desviación del concepto de descentralización**. En determinadas situaciones, alguna organización podría necesitar compartir datos y brindar algún tipo de seguridad sobre su autenticidad.

> **Por ejemplo**
>
> Departamentos de gobierno.

### Tokenizadas

Son las ***blockchains* estándares que generan criptomonedas** como resultado de un proceso de consenso por medio de minado o distribución inicial.

### Sin tokens

Se utilizan cuando solamente se requiere **compartir datos entre varias partes confiables** sin necesidad de transferir valores.

### *Blockchain* de capa 1

Todas las **operaciones se desarrollan en el mismo nivel**. Pueden ser de arquitectura:

- **Monolítica**: no poseen componentes fuera de la cadena de bloques única. Por ejemplo: Bitcoin, Ethereum, Solana.
- **No monolíticas**: *polylithics*, múltiples cadenas conectadas a una cadena principal. Por ejemplo: Polkadot, Avalanche, Cosmos.

### *Blockchain* de capa 2

Emergen como **solución a los problemas de escalabilidad y privacidad a las *blockchain* de capa 1**. El consenso se desarrolla en la capa 1 y las transacciones en la capa 2.

> **Por ejemplo**
>
> *Sidechains, zero-knowledge rollups, optimistic rollups, plasma chains, Lightning Networks.*

# Ventajas y limitaciones de Blockchain

## Ventajas

- Ahorros en los costos.
- Simplificación en los negocios.
- Propiedad digital.
- Descentralización.
- Transparencia y confianza.
- Inmutabilidad.
- Alta disponibilidad.
- Alta seguridad.
- Plataforma para *smart contracts*.

## Limitaciones

- Escalabilidad (a otras redes).
- Regulaciones.
- Privacidad.
- Tecnología relativamente inmadura.
- Interoperabilidad con otros sistemas.
- Adopción.

# ¿Cuándo usar una *Blockchain?*

| Pregunta | Si / No | Solución recomendada |
|---|---|---|
| ¿Necesita alto rendimiento de datos? | Si | Usar una base de datos tradicional. |
| | No | Una base de datos centralizada puede ser útil, excepto que no pueda establecer confianza. |
| ¿Deben controlarse centralmente las actualizaciones? | Si | Use una base de datos tradicional. |
| | No | Podría usar una blockchain pública o privada. |
| ¿Confían los usuarios unos en otros? | Si | Use una base de datos tradicional. |
| | No | Use una blockchain pública. |
| ¿Los usuarios son anónimos? | Si | Use una blockchain pública. |
| | No | Use una blockchain privada. |
| ¿Se requiere que el consenso sea mantenido dentro de un grupo? | Si | Use una blockchain privada. |
| | No | Use una blockchain pública. |
| ¿Se requiere estricta inmutabilidad de datos? | Si | Use una blockchain. |
| | No | Use una base de datos tradicional/centralizada. |

# Sistemas

- Los **sistemas centralizados** son sistemas cliente servidor convencionales donde una autoridad única controla y se encarga de todas las operaciones del sistema. Todos los usuarios dependen de esta única fuente de servicios.
- Los **sistemas distribuidos** replican los datos y la computación a través de múltiples nodos en una red donde los usuarios ven un sistema único y coherente.
- Los **sistemas descentralizados** son un tipo de red donde el control se distribuye entre varios nodos. Una innovación significativa es el consenso descentralizado.

Una ***blockchain*** es un libro de contabilidad distribuido que **se ejecuta sobre varios tipos de sistemas** convencionales que incluyen almacenamiento, comunicación y computación.

# Descentralización

Una **organización descentralizada (DO)** es un programa de *software* que se ejecuta sobre una *blockchain* y está basada en un modelo real de organización con personas y protocolos. Una vez añadida a la *blockchain* en la forma de un *smart contract*, se descentraliza y comienza a interactuar con otras DOs en base a su programación.

- Una **organización autónoma descentralizada** (DAO) es una DO completamente autónoma. Las DO necesitan intervención humana para ingresar o ejecutar lógica de negocios. No tienen fines de lucro.
- Una **corporación autónoma descentralizada** (DAC) es una DAO pero con fines de lucro.
- Una **sociedad autónoma descentralizada** (DAS) es una sociedad entera que puede funcionar sobre una *blockchain* con la ayuda de múltiples y complejos *smart contracts* junto a una combinación de DAOs y aplicaciones descentralizadas (DApps) ejecutándose autónomamente.

## DApps *(Decentralized Applications)*

Las DAO, las DAC y las DO pueden ser consideradas como **DApps que se ejecutan sobre una *blockchain*** en una red de pares.

Las DApps pueden alcanzar el consenso mediante algoritmos tales como ***Proof Of Work* (PoW) o *Proof of Stake* (PoS).**

Las DApps pueden distribuir tokens (monedas) mediante minado, recaudación de fondos o desarrollo.

Las DApps son **softwares** que se ejecutan según determinadas maneras: de Tipo 1, Tipo 2 o Tipo 3.

- **Tipo 1**: se ejecutan en su propia *blockchain* dedicada. Por ejemplo: dApps basadas en *smart contracts* sobre Ethereum. Si lo necesitan, pueden utilizar sus *tokens* nativos (en este caso, ETH).
- **Tipo 2**: utilizan una *blockchain* preexistente establecida. Por ejemplo: DAI está construida sobre Ethereum pero utiliza su propia criptomoneda, mecanismos de distribución y control. Otros ejemplos son Golem y OMNI.
- **Tipo 3:** utilizan los protocolos de las dApps de Tipo 2. Por ejemplo: la red SAFE utiliza los protocolos de red de OMNI.

## Principios fundamentales para una dApp

- Estar totalmente **descentralizada**.
- Ser ***open source*** / ***free software***.
- Ser criptográficamente **segura**.
- Debe incentivar su **disponibilidad**.
- Debe dar prueba de valor mediante generación de ***tokens***.

## Sistemas de almacenamiento

La *blockchain* no es adecuada para almacenar grandes cantidades de datos. Suelen utilizarse tablas de *hash* distribuidos (DTH).

Los sistemas de almacenamiento más comunes son **IPFS, Ethereum Swarp, Storj, MadSafe, BigChainDB.**

## *Smart contracts*

Un ***smart contract*** es un programa de *software* que se ejecuta generalmente sobre una *blockchain* debido a los beneficios de seguridad que esta tecnología le brinda. Contiene algo de la lógica del negocio y una limitada cantidad de datos. La lógica del negocio se ejecuta si se cumplen determinadas condiciones. Los actores de la *blockchain* usan los *smart contracts*, o se ejecutan en forma autónoma en su nombre.

Un **agente autónomo (AA)** es una entidad de *software* que actúa en representación de su dueño para alcanzar una meta con intervención mínima o nula del mismo.

# ¡Sigamos trabajando!