The UTXO Models Handbook

All Flavors of UTXO models in one menu

Authors:
Ignacio Calderon de la Barca [UTXO Alliance],
Steph Macurdy [WBL, UTXO Alliance]

Contributors:
Rupert Whitehead [IOG]
Matt Quinn [Nervos Network]
Kushti [Ergo]
Alan Orwick [Quai Network]
Jon Woodard [WBL]
Yan Martins [Hathor]

2025/01/03

Contents

Overview

It's been a decade since Bitcoin with its underlying computing model, the **Unspent Transaction Output (UTXO) model**, brought the most prominent application of blockchain technology today - a decentralized financial system. These simple set of words encompass various virtues that make this new type of system much better than legacy financial systems.

Bitcoin's seminal work started the first wave of projects that offer a system with the attributes described above. Years later this pioneer breakthrough inspired subsequent revolutions reimagining socio-economic systems beyond the uni-dimensional feature of supporting transactions in a peer-to-peer manner. A second wave introduced **smart contracts**, which allows users to put conditions on financial transactions written in verifiable code which paves the way for **decentralized applications (dApps)** to emerge. Furthermore, a third wave of projects embedded self-governing mechanisms in the system. Today, a competitive blockchain must offer a network protocol that meets all previous advancements, in addition to having a detailed roadmap that is highly focused on maximizing all three dimensions of the decentralization-security-scalability triad (also known as the **'blockchain trilemma'**).

The approach that promising blockchain projects take is to first lay solid foundations weaved by multi-diciplinary research that latter on compounds into better quality of software. This distinction is characteristic of all UTXO Alliance members. The Alliance has, at its core, the goal of advancing blockchain technology by fostering collaboration towards interoperability, research, commercial adoption and education. The latter sets the tone of this manuscript.

How to read this document?

Chapter 1 presents the core components of a blockchain as a visual language (also referred to as *ingredients* in this handbook) and it is meant to be read sequentially. Whereas Chapter 2 focuses on explaining how all components are assembled to form a fully functionning blockchain. Aditionlly, we present diagrams for all UTXO Alliance chains highlighting their high-level technical specification differences and smart contract implementation. At last, Chapter 3 contains Supplementary Material including Section 3 which explains the basic *Account model* and compares it to the UTXO model. If you're curious about a particular chain jump right ahead to check Bitcoin (Section 2), Cardano (Section 2), Ergo (Section 2), Nervos Network (Section 2), Quai Network (Section 2), Topl (Section 2), Alephium (Section 2), Digibyte (Section 2) and Hathor (Section 2).

Goals

1 This handbook aims to be a presentation card of all the different chain designs of UTXO Alliance members. Aiming to be a handy resource for blockchain enthusiasts with minimal technical knowledge, entrepreneurs and commercial partners
2 Explain the key aspects of the UTXO model and the functioning of a blockchain using simple visual diagrams
3 Demonstrate the rich variety and benefits present in the variance of UTXO design patterns implemented by differnt members of the UTXO Alliance

Introduction to Blockchains

We recommend these three educational introductions to get up to speed for what you're about to read. These three resources are evergreen (meaning they've aged well with time) and include:

1 Digestable explainer on How does Bitcoin Work? by learn me a Bitcoin
2 Explainer on How Blockchains work by Anderson Brown
3 Technical YouTube video about Bitcoin by 3Blue 1Brown

Chapter 1: Ingredients of a Blockchain

At its heart, a blockchain is a special kind of database that keeps track of information in a way that's secure, open for anyone to see, and nearly impossible to retroactively change. Unlike traditional databases managed by a single organization, blockchains are maintained by a network of computers working together. This design ensures that no single person or group has complete control over the information, see Fig. 1.2.

Understanding its core components is crucial for grasping how blockchains function and how different implementations or "flavors" of blockchain systems offer different benefits. In this section, we'll break down the essential "ingredients" that make up a UTXO-based blockchain. By examining these building blocks – cryptographic primitives, transactions, blocks, and the chain itself – we can better appreciate the innovations and variations in different blockchain designs.

As we explore each ingredient, we'll see how they contribute to the overall functionality and security of the blockchain. We'll also touch on how various blockchain projects have modified or extended these basic components to create their own unique characteristics, leading to a rich ecosystem of blockchain "flavors" all stemming from the same fundamental recipe. Lets navigate through Fig.1a) we have the most high-level construction of a blockchain. Each of the color boxes are the core concepts to construct a blockchain, we'll call these our 'ingredients'. So we have 6 ingredients in total:

1 **Hash functions (H) and cryptographic signatures (S)** (blue boxes) - The reason that these two concepts can be labeled under the same ingredient is because signatures use hashes in their internal mechanism. Each of these are crucial and highly used in many other parts of the blockchain

2 **Transaction (Tx)** (red box)- A transaction allows to change data, ie. move crypto value from one party to another

3 **UTXO Set & Ledger** (green boxes) - The aggregation of outputs for tracking global and local state

4) **Block (B)** (orange box) - The container of transactions, verification of consensus, and compressed record of history

4 **Protocol** (Π) (purple box) - The formal system by which nodes reach agreement

5 **Network (**N**)** (gray box) - The system by which nodes communicate and ultimately facilitate agreement

Five components/ingredients make any recipie/blockchain

Despite the variability in the design of UTXO-based chains, as displayed by different UTXO Alliance members, all blockchains have the same essential components.

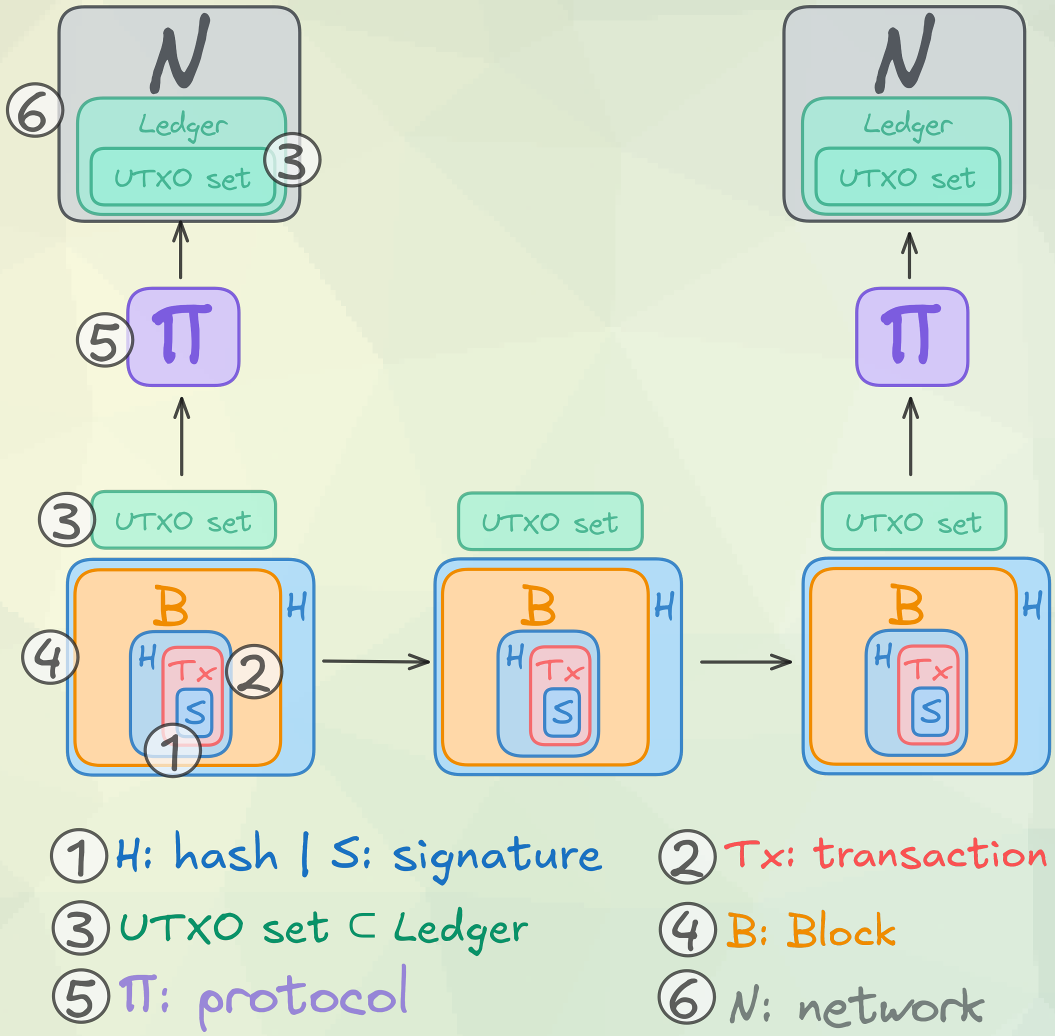


Figure 1.1: Simplified view of blockchain structure based on the composition of our simple blockchain ingredients.

Cryptographic Primitives

Cryptographic primitives form the foundation of blockchain security and functionality. Two crucial elements are hash functions (H) and digital signatures (S).

Hash functions are one-way mathematical operations that convert any input into a fixed-size output; see Fig. 1.3 (top). In blockchains, they create unique identifiers, link blocks, and support consensus mechanisms such as Proof-of-Work.

Digital signatures Fig. 1.3 (bottom) are ge/nerated using hash functions and provide identity in blockchain systems. Unlike traditional systems where identity is tied to government-issued documents, blockchain identities can be pseudonymous, offering a new paradigm for verification and authentication in digital networks.

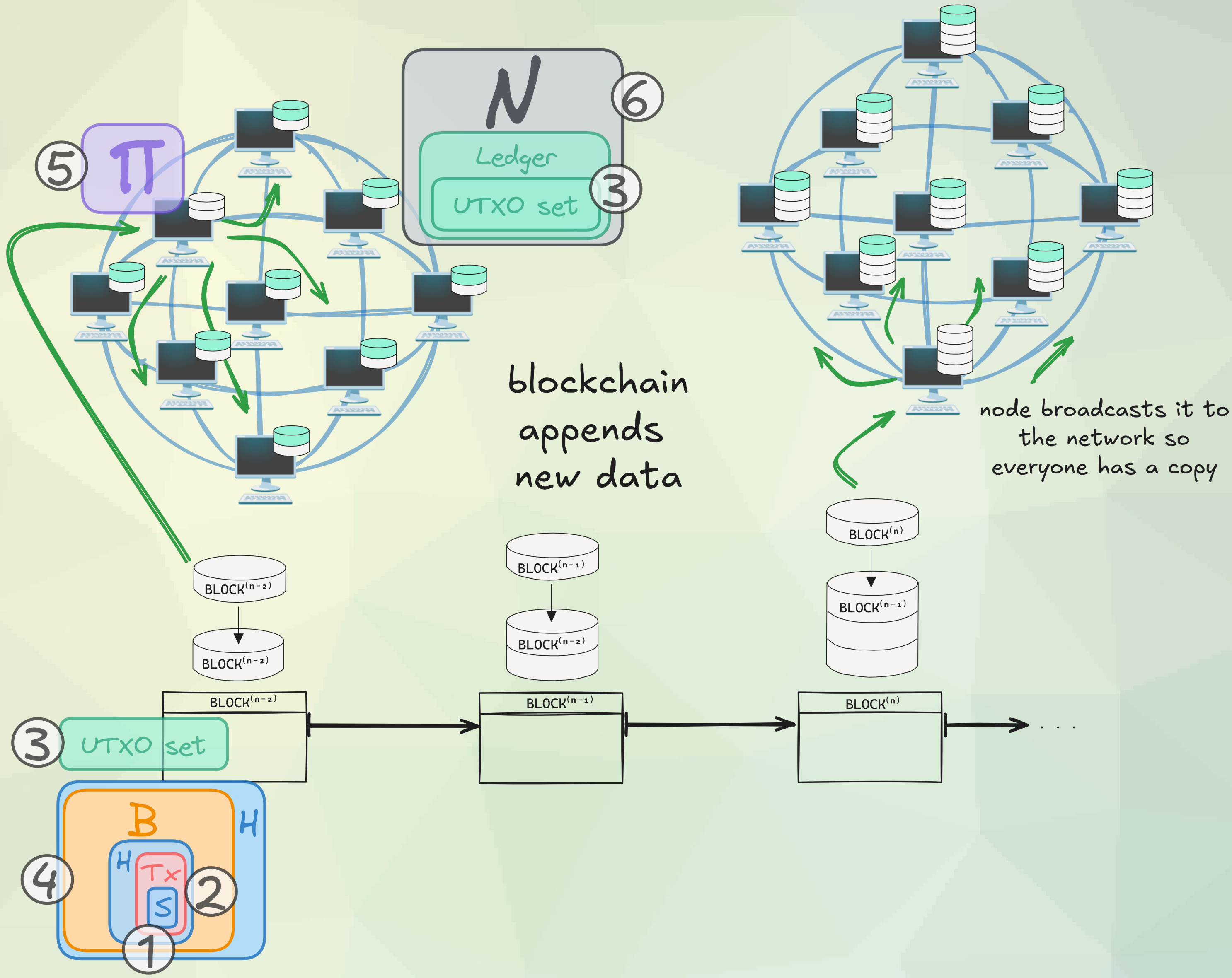


Figure 1.2: General functioning of a blockchain.

Transactions

Transactions are the basic units of state/value transfer in a blockchain. In the UTXO model:
a) Structure: Fig. 1.4 transactions consist of inputs (black arrows) and outputs (red arrows). b) Inputs: Reference previous transaction outputs (UTXOs) being spent. c) Outputs: Specify new UTXOs being created, including recipient addresses and amounts. d) Conservation Law: The sum of inputs must equal or exceed the sum of outputs (minus a transaction fee). e) Signatures: Each input must be signed by the owner of the corresponding UTXO.

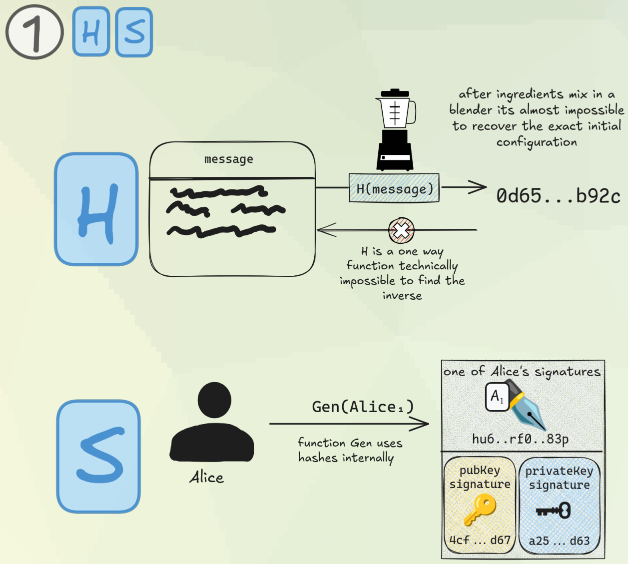


Figure 1.3: We can see that our label ingredient H (top) refers to cryptographic hash functions which are one-way functions. On the other hand our label ingredient S (bottom) refers to digital signatures, where a private number is transformed into a key pair, one capable of being shared publicly and one forever kept private and secure. The public key can be used like a mailing address or identity, and the private key can perform operations privately which confirm approval, ownership and identity. This functionality is used in many internet and computer protocols besides blockchains.

The UTXO Set and Ledger

The UTXO set plus additional data is what ultimately the Ledger stores. The slicing of UTXO selection is what a user's wallet performs in order to determine how many outputs an address holds. The Ledger refers to the long running concatenation of transactions via blocks, the UTXO set represents the proverbial "tip of the chain" which is the most relevant perspective of current state. See Fig. 1.5

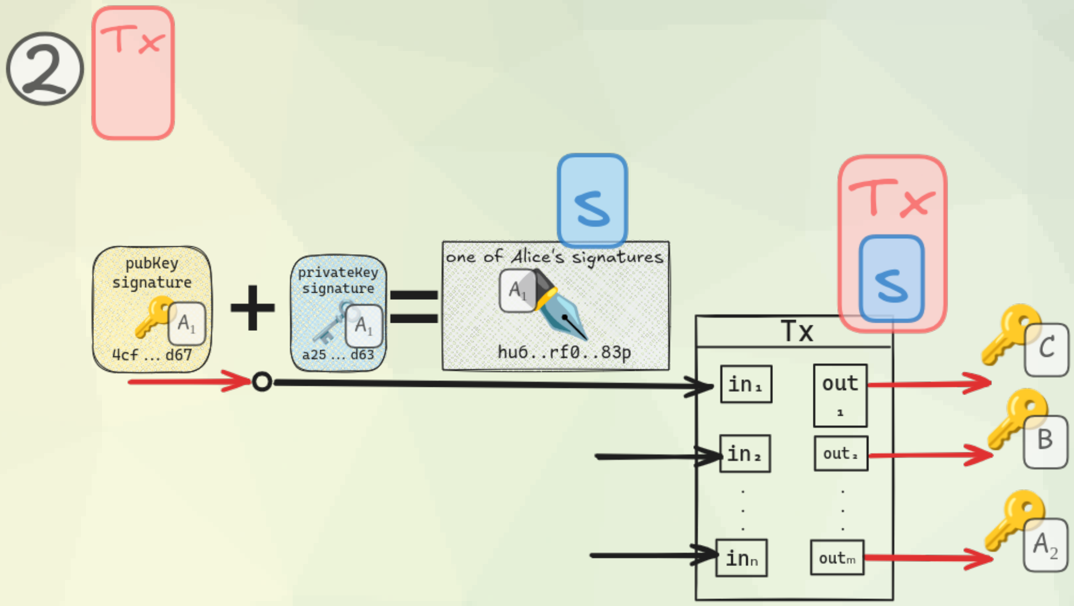


Figure 1.4: A transaction in the UTXO model.

Block

Blocks are containers for transactions and form the backbone of the blockchain: Structure: Typically includes a header and a list of transactions. b) Block Header: Contains metadata such as:

- Previous block hash (creating the chain)
- Merkle root of transactions
- Timestamp
- Nonce for consensus protocols such as Proof-of-Work (PoW). PoW is a mechanism to make block creation computationally expensive, ensuring security.
- Coinbase Transaction: A special transaction in each block that creates new currency and collects transaction fees.

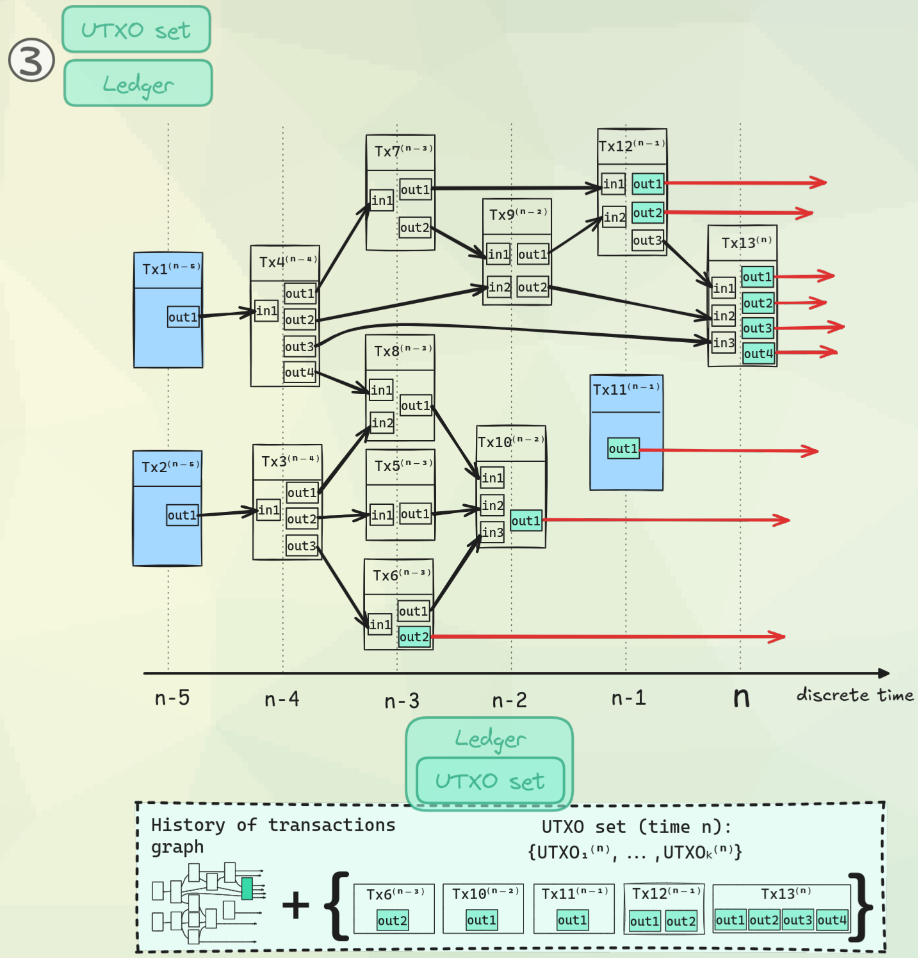
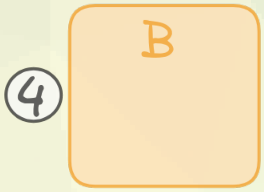


Figure 1.5: The UTXO set and the UTXO model graph.

The Chain

The chain is the ordered sequence of blocks, forming the whole historical ledger:

a) Genesis Block: The first block in the chain, often hardcoded into the software. b) Longest Chain Rule: In case of forks, nodes typically follow the chain with the most accumulated proof-of-work. c) Consensus: The chain represents the agreed-upon history of transactions. d) State: The current set of UTXOs, derived from processing all transactions in the chain.



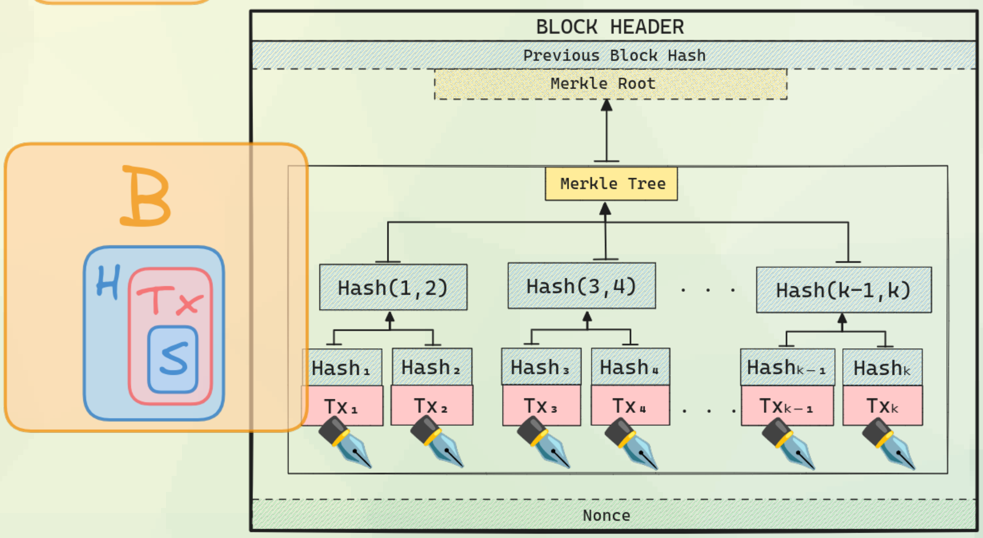


Figure 1.6: A filled block with transactions packaged efficiently using a Merkle Tree data structure.

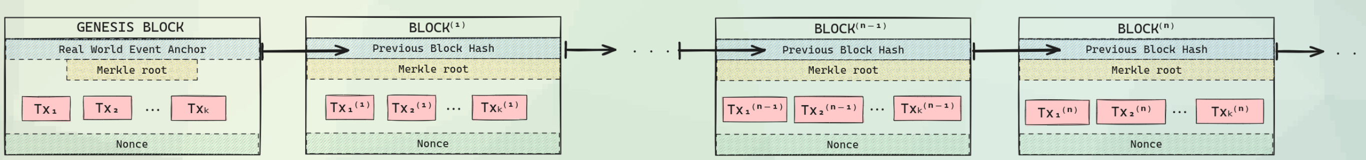


Figure 1.7: An irreversible sequence of blocks linked by a pointer reference to a previous Block hash ID.

Chapter 2: The Blockchain

Bitcoin's UTXO model - the vanilla flavor of UTXOs

Bitcoin, introduced by Satoshi Nakamoto in 2008, aimed to create a decentralized digital currency system that could operate without the need for intermediaries such as banks or governments. Its primary goal was to enable peer-to-peer electronic transactions in a trustless environment, solving the double-spending problem through a distributed ledger (blockchain) and a consensus mechanism (Proof of Work).

However, Bitcoin's groundbreaking design also came with limitations. Its relatively simple scripting language limits complex smart contract functionality. Scalability issues, evident in low transaction throughput and high fees during network congestion, hinder its use for everyday transactions. Bitcoin's Proof-of-Work consensus, while secure, is energy-intensive and leads to mining centralization. Additionally, Bitcoin's fixed monetary policy, while appealing to some, lacks the flexibility to adapt to varying economic conditions. These limitations have inspired a new generation of blockchains to explore alternative consensus mechanisms, more expressive smart contract capabilities, improved scalability solutions, and innovative governance models.

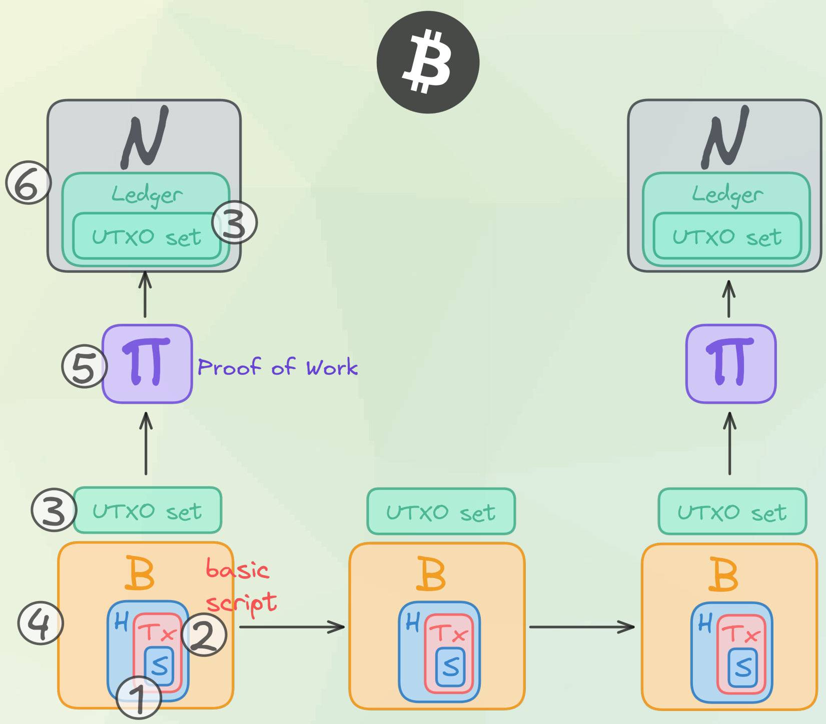


Figure 2.1: The Bitcoin blockchain viewed as simple composition of our ingredients.

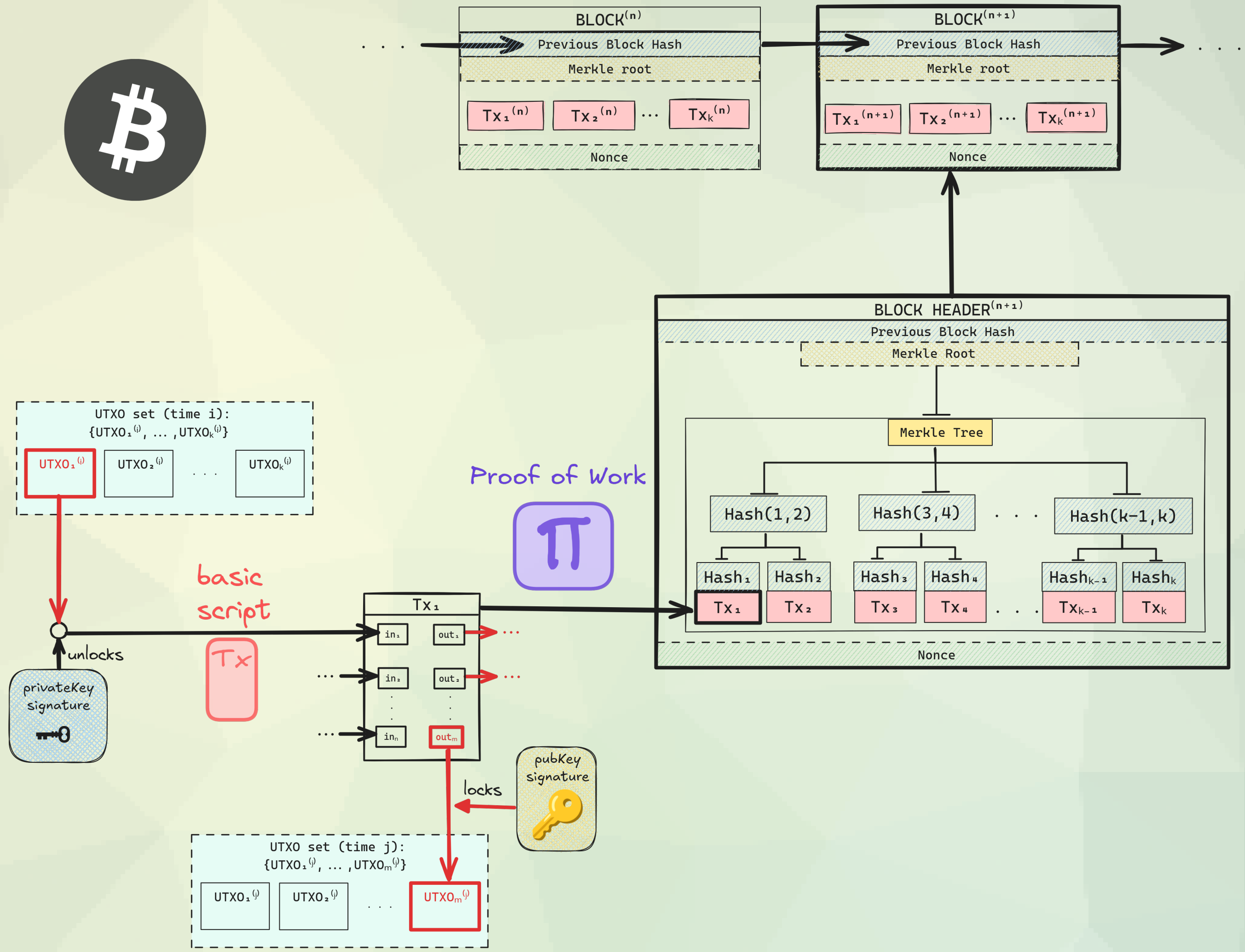


Figure 2.2: Detailed visual of Bitcoin.

DIGIBYTE

DigiByte is one of the longest-running blockchain platforms, launched in 2014 as a secure, fast, and highly decentralized blockchain focusing on digital payments and asset transfers. The platform is notable for its implementation of five distinct mining algorithms and its advanced difficulty adjustment system.

At its core, DigiByte employs a multi-algorithm mining approach called MultiAlgo. This system uses five different mining algorithms (Scrypt, SHA256, Qubit, Skein, and Odocrypt) operating simultaneously, which helps maintain decentralization by preventing any single type of mining hardware from dominating the network. This approach also provides enhanced security against 51The blockchain architecture features a three-layer system:

- Core Protocol Layer - handling network operations and security
- Digital Asset Layer - managing transfers and data
- Applications Layer - supporting decentralized applications

DigiByte's block time is notably fast at 15 seconds, significantly quicker than Bitcoin's 10 minutes. This rapid block time, combined with the MultiAlgo system, allows for quick transaction confirmations while maintaining security. The platform implements MultiShield real-time difficulty adjustment for each algorithm, occurring every block rather than at longer intervals.

The network's scaling approach includes SegWit, which allows for effective block size usage. This has enabled DigiByte to maintain high transaction throughput while keeping fees low.

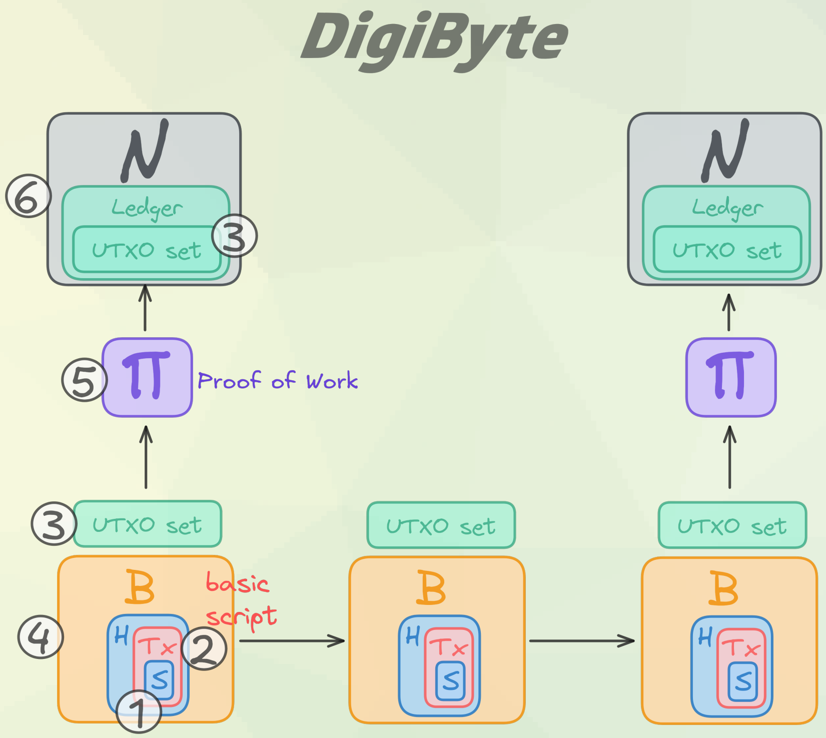


Figure 2.3: Simple Digibyte blockchain schematic.

The Extended UTXO Model - Cardano

The Extended UTXO (EUTXO) model is a significant enhancement of the traditional Unspent Transaction Output (UTXO) model, originally used in Bitcoin. Cardano's EUTXO model introduces additional functionalities that facilitate more complex smart contracts while maintaining the benefits of the UTXO paradigm. This model allows for better scalability and flexibility in transaction processing. The extended part refers to the fact that each UTXO can now carry: *(i)* A value (ADA and native tokens), *(ii)* a datum (arbitrary data) and *(iii)* a validator script (the smart contract logic).

Key Features of the EUTXO Model:

1 *Enhanced Data Handling* - In the EUTXO model, each UTXO can carry additional data known as datum. This data can be used to store information relevant to a transaction, such as smart contract state or other contextual information. The inclusion of datum allows developers to implement more sophisticated logic within transactions without compromising the integrity and simplicity of the UTXO model.

2 Redeemer Context - The redeemer context is a critical aspect of the EUTXO model that specifies how a datum is utilized during transaction validation. When a transaction attempts to spend a UTXO, it must provide a corresponding redeemer that indicates how the datum should be interpreted and used. This mechanism ensures that only valid transactions can access and manipulate the associated datum, adding a layer of security and correctness to smart contract execution.

3 *Transaction Validation* - In the EUTXO model, each transaction is validated based on its inputs (the UTXOs being spent), the associated datum, and the redeemer provided. This validation process checks whether the redeemer correctly corresponds to the expected operations defined by the datum. Thus, it allows for complex interactions while ensuring that all conditions are met before a transaction is executed.

Extended UTXO model (EUTXO)

Script Attachment: In the EUTXO model, scripts (or smart contracts) are attached directly to outputs. This is represented by the 'Script' component in the Output structure. This allows for more complex validation logic to be associated with each UTXO.

Datum and Redeemer: (i) Datum: This is arbitrary data that can be attached to an output. It represents the state of the script and is stored on-chain. (ii) Redeemer: This is provided by the transaction that wants to spend a UTXO. It contains the arguments for the script execution.
Context-Aware Validation: The EUTXO model provides rich context to the validator script, including information about the entire transaction and even parts of the blockchain state. This is represented by the 'Context' in the diagram.
Validity Interval: Transactions in the EUTXO model can specify a validity interval (represented by 'Slot Height' in the diagram). This allows for time-sensitive smart contracts.
Script Execution: The validator function takes three inputs: Datum, Redeemer, and ScriptContext. It returns a boolean indicating whether the transaction is valid or not.

Key Advantages of EUTXO:

1 Increased Expressiveness: The addition of Datum and Redeemer allows for more complex state management in smart contracts.
2 Local State Validation: Each input can be validated independently, which allows for better parallelization and scalability.
3 Predictability: The outcome of script execution can be predicted off-chain, reducing the risk of failed transactions.
4 Fee Prediction: The deterministic nature of script execution allows for accurate fee prediction.1.
5 Enhanced Privacy: The UTXO model inherently provides better privacy compared to account-based models.
6 Time-Sensitive Logic: The validity interval allows for time-based contract logic.

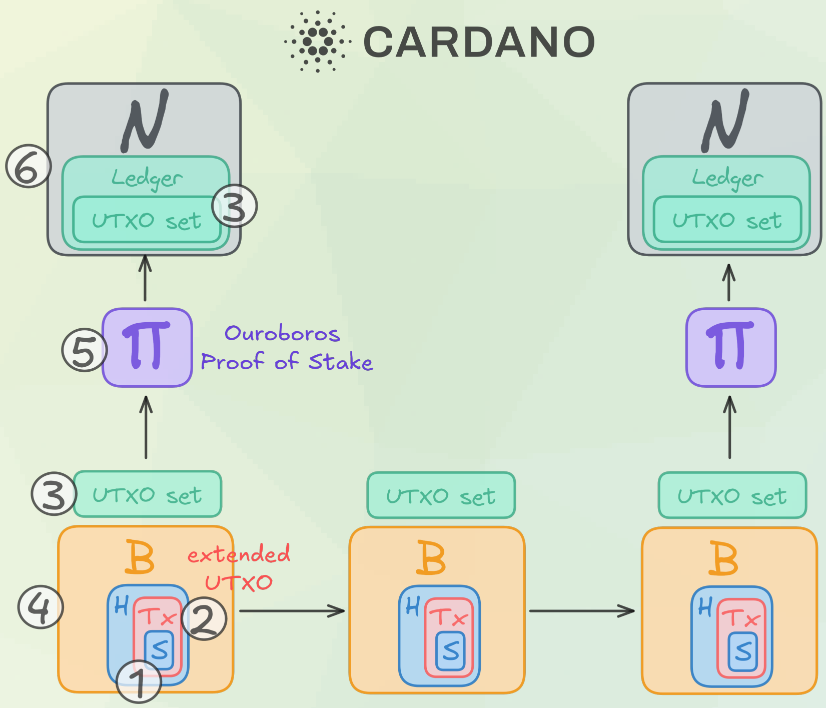


Figure 2.4: Simple Cardano blockchain schematic.

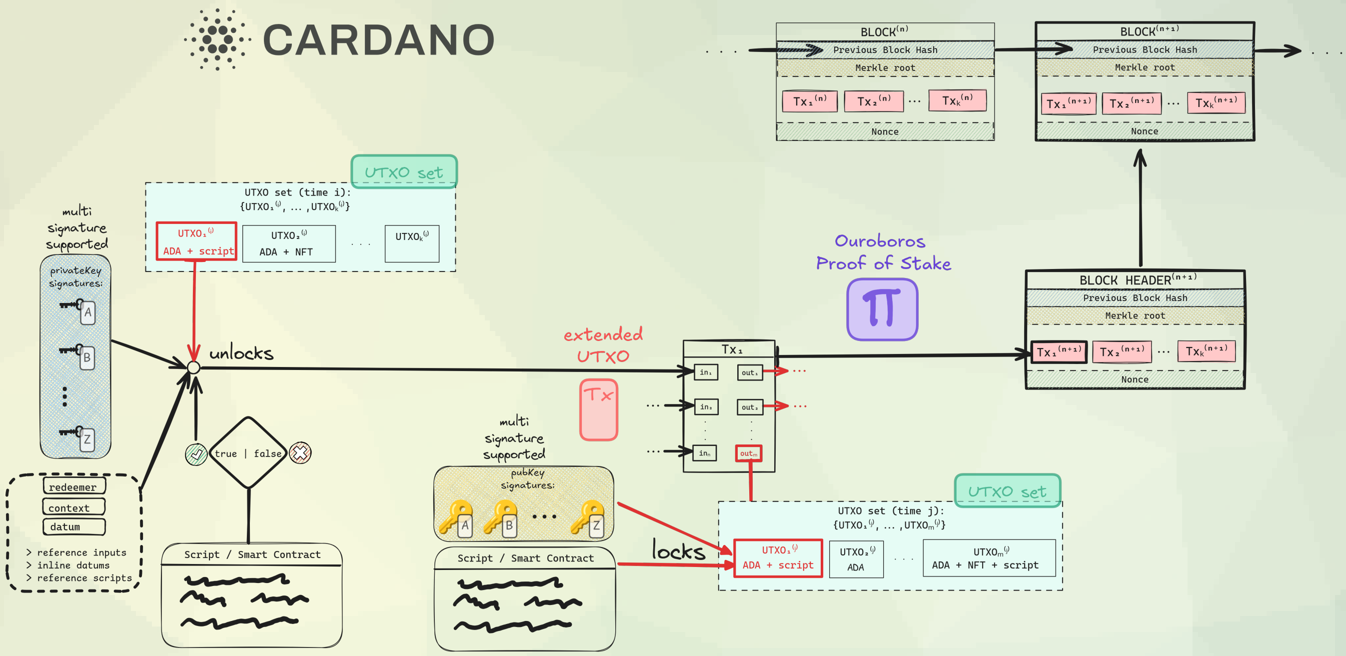


Figure 2.5: A detailed view of the Cardano blockchain and its extended UTXO.

The Extended UTXO Model - Ergo

The Ergo blockchain implements smart contracts through its Extended UTXO (eUTXO) model, which enhances the basic UTXO model used in Bitcoin with additional programmability. Here's how the key components work together:

Extended UTXO Model:

The eUTXO model extends the basic UTXO model by adding contextual information and validation logic to transaction outputs. Each output can carry both a value and arbitrary data, allowing for more complex contract conditions. This is fundamentally different from Ethereum's account-based model and provides several advantages for smart contract execution.

Context: Represents the current transaction's environment and blockchain state. Includes information like block height, timestamps, and transaction inputs/outputs. Allows contracts to make decisions based on network conditions. The context is essential for time-locked contracts and other condition-based executions
Datum: Represents the persistent state stored with each UTXO, it contains arbitrary data that can be used to track contract state. It acts like a "memory" for the smart contract and can store things like ownership information, voting results, or game state
Redeemer: Provides the arguments needed to spend a UTXO. Contains the input parameters that satisfy the contract's conditions and is used to prove that spending conditions are met. The redemer can include signatures, proofs, or other validation data

Multisignature Implementation - Ergo's eUTXO model enables sophisticated multisignature schemes through:

Script Creation: Multiple public keys can be included in the contract script. The script can specify the required number of signatures (m out of n signatures). Different spending conditions can be combined with boolean logic
Signature validation: The redeemer must provide the required number of valid signatures. The context ensures signatures match the specified public keys. Note that additional conditions can be added (time locks, custom logic)
Flexibility: Supports both threshold signatures and more complex arrangements which can combine multisig with other contract conditions. Allows for dynamic participant sets through contract logic

This design allows Ergo to support complex smart contracts while maintaining the security and scalability benefits of the UTXO model. The combination of Context, Datum, and Redeemer provides the necessary components for sophisticated contract logic, while the eUTXO model ensures efficient and predictable execution.

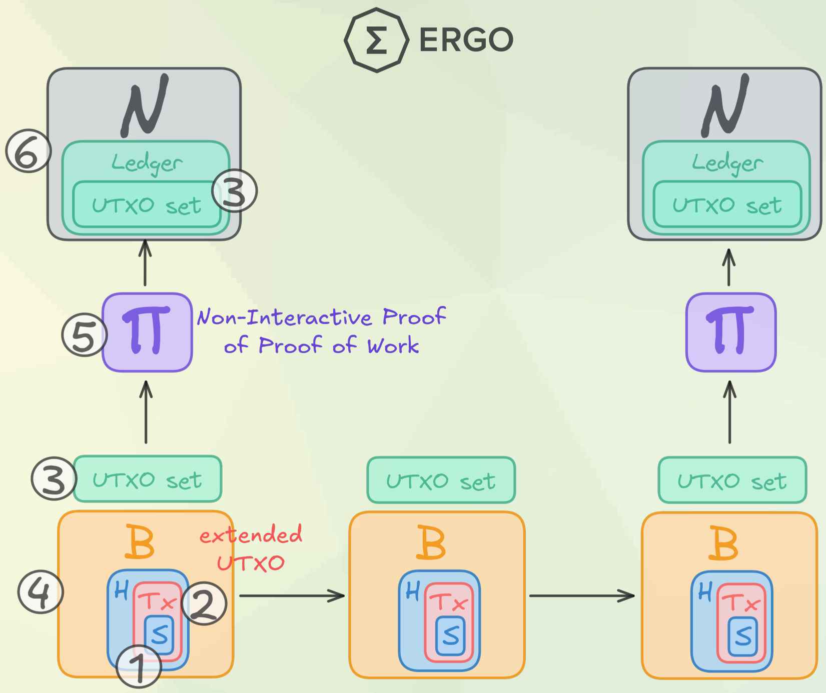


Figure 2.6: Simple Ergo blockchain schematic.

Common Knowledge Base (CKB) - Nervos Network

Nervos Network, launched in 2019, aims to solve blockchain trilemma issues (scalability, security, and decentralization) through a unique layered architecture. It seeks to provide a secure foundation for decentralized applications while allowing for scalable solutions on higher layers.

Key features of Nervos Network include:

1. Layered Architecture: Consists of a secure base layer (CKB) and flexible layer 2 solutions for scalability.
2. Cell Model: An enhanced UTXO model that allows for more complex state management and smart contract capabilities.
3. Native Token Economics: CKByte tokens represent state storage on the blockchain, creating an economic model that aligns with long-term network sustainability.
4. Proof-of-Work Consensus: Uses the NC-MAX algorithm, designed to be ASIC-neutral to maintain decentralization. Interoperability: Designed to facilitate cross-chain interactions and serve as a hub for other blockchain networks.
5. Nervos Network aims to provide a versatile platform that can serve as both a secure value storage and a foundation for scalable decentralized applications, addressing limitations in both Bitcoin-like and Ethereum-like blockchains.

Smart contracts in Nervos Network are implemented through a unique model that differs from account-based blockchains like Ethereum. Instead of having smart contract code stored on-chain that operates on state, Nervos uses a cell model with two types of scripts: **Lock-scripts** and **Type-scripts**.

- **Lock-scripts** - Function as ownership validators for cells (similar to Bitcoin's P2PKH scripts). This design allows control on who can spend/use the cells. Lock-scipts are executed during cell consumption that typically verify signatures or other ownership conditions. Must return a true boolean value for the transaction to be valid
  Type-scripts - Define and enforce rules for cell transformations, validate state transitions and implement business logic. Type-scripts are optional but powerful for complex applications that enable implementation of various token standards and DeFi protocols.

The **CKB-VM (Common Knowledge Base Virtual Machine)** - is fundamental to this architecture:

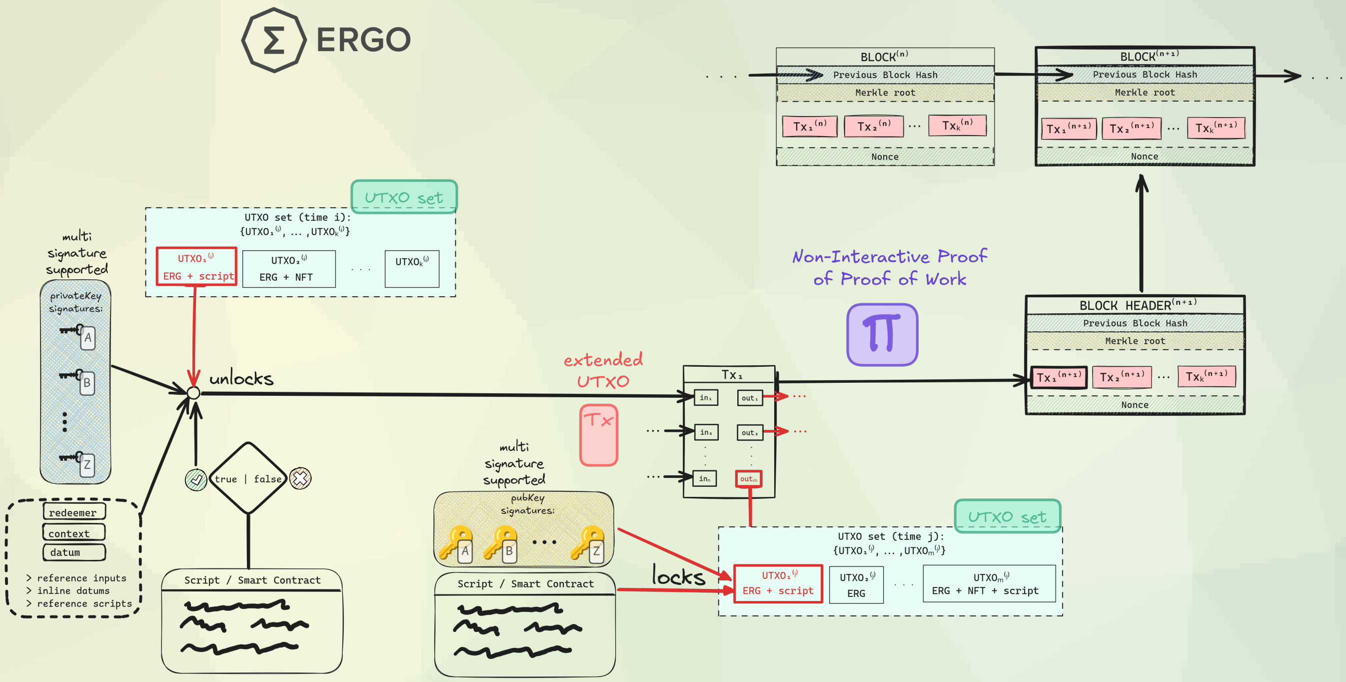


Figure 2.7: A detailed view of the Cardano blockchain and its extended UTXO.

It's a **RISC-V** based virtual machine in charge of executing both Lock-scripts and Type-scripts. It provides deterministic execution across the network. An amazing feature is that it supports multiple programming languages since any language that can compile to RISC-V can be used. Moreover, it offers high performance due to its simple instruction set and maintains security through its verified software stack.

The interaction between these components creates a flexible and powerful smart contract system:

When a transaction is initiated:

(i) Lock-scripts verify ownership/permissions. (ii) Type-scripts validate state transitions and (iii) CKB-VM executes both script types.

For complex applications multiple Type-scripts can be combined. Additionally different cells can have different rules and custom validation logic can be implemented at the cell level.

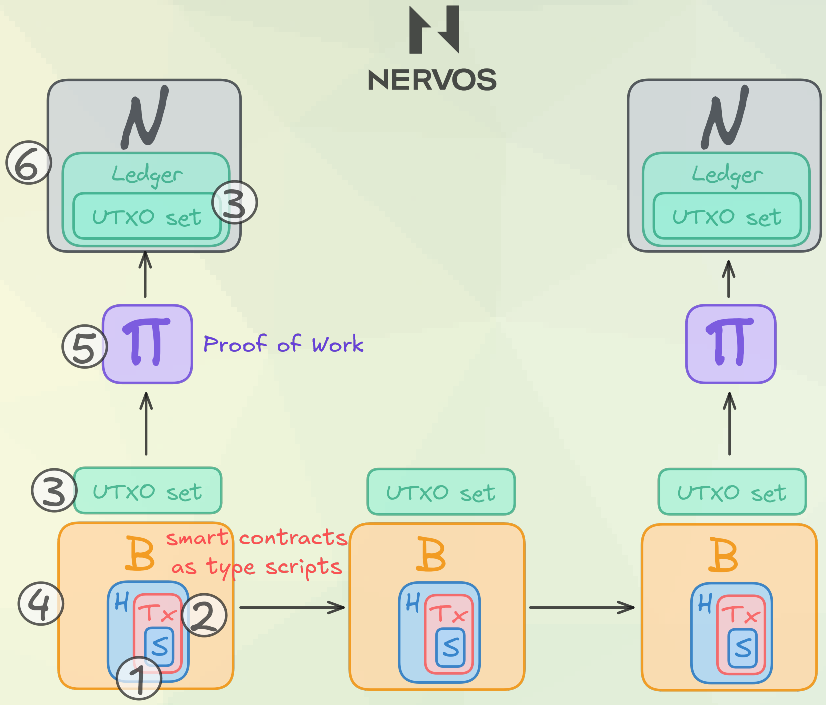


Figure 2.8: Simple Nervos blockchain schematic.

Quai Network

Quai Network is a decentralized blockchain ecosystem designed to enhance scalability and efficiency in cryptocurrency transactions. Its architecture is built around a multichain structure consisting of three main components: Prime, Paxos, and Cyprus. Additionally, the network employs a unique consensus mechanism called Proof of Entropy Minima (PoEM), which integrates these components into a cohesive system.

Quai has an *EVM Chain* component where the Solidity-compatible EVM portion is relatively straightforward, operating similarly to other EVM chains with smart contract capabilities, allowing developers tVo: deploy Solidity smart contracts and use familiar development tools and patterns. On the other hand, Quai has a *UTXO Chain Integration* which is its more novel aspect. In UTXO chains, smart contracts work differently than in account-based models.

Scripts are attached to individual UTXOs rather than residing at fixed addresses. Contract state is managed through UTXO ownership and movement. Validation occurs through script execution when UTXOs are spent.

Multichain Structure

> **Prime** serves as the foundational layer of the Quai Network. It is designed to handle high transaction volumes, boasting a throughput of over 50,000 transactions per second (TPS). This layer is optimized for decentralized applications (dApps) and supports Ethereum Virtual Machine (EVM) compatibility, allowing developers to easily migrate existing Ethereum-based applications to the Quai Network. The architecture of Prime ensures that it can scale effectively while maintaining decentralization, which is crucial for the network's overall security and performance.

> **Paxos** operates as a secondary chain focused on facilitating stablecoin transactions and maintaining price stability within the network. It aims to provide predictable value through its native stablecoin, which is essential for users seeking to transact without the volatility commonly associated with cryptocurrencies. Paxos enhances the liquidity of the Quai Network by enabling seamless conversions between various digital assets, thereby supporting both trading and everyday transactions.

> **Cyprus** acts as an auxiliary chain that enhances interoperability among different blockchains within the Quai ecosystem. It enables cross-chain communication and facilitates the transfer of assets across the various layers of the network. This capability is crucial for creating a unified experience for users who wish to interact with multiple chains without facing barriers or liquidity issues.

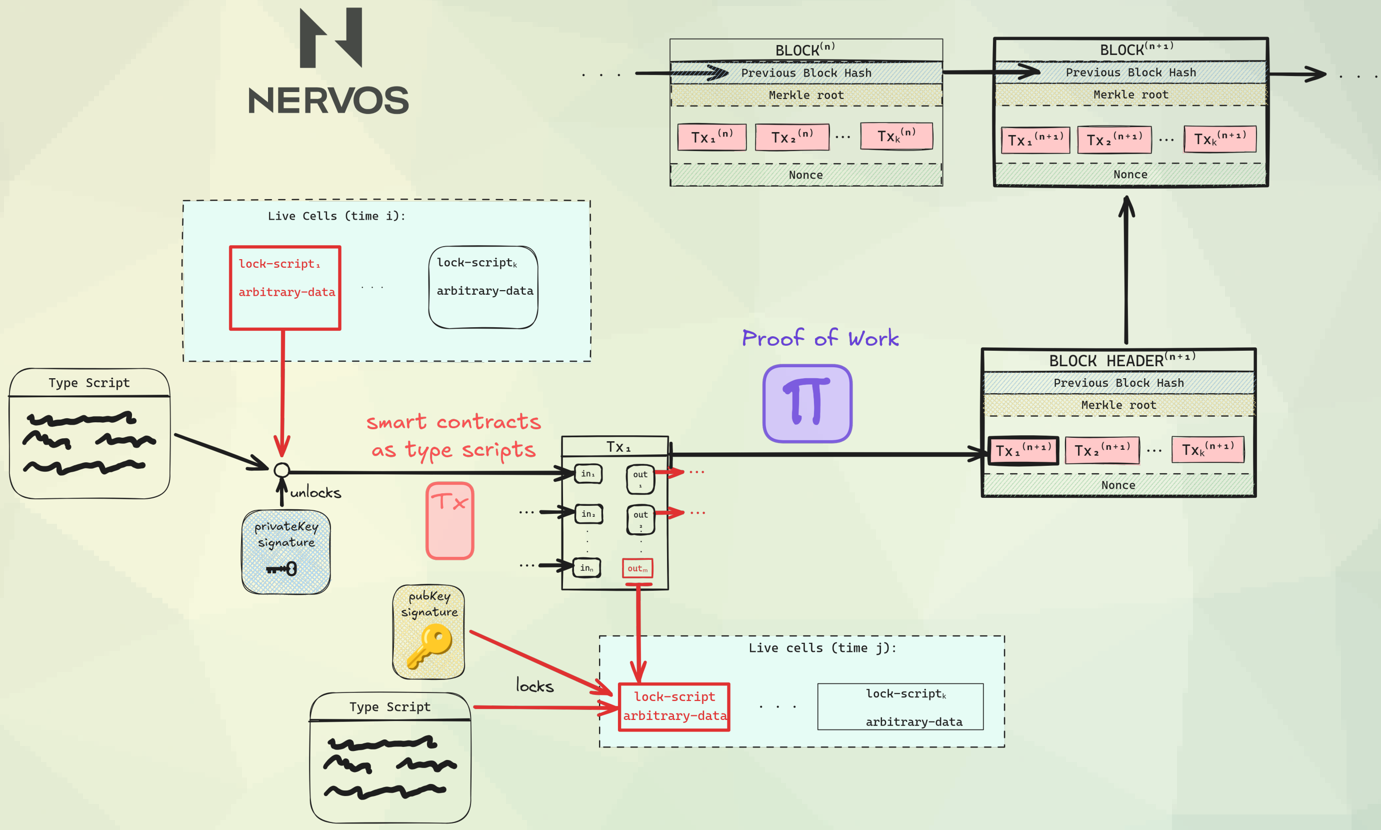


Figure 2.9: A detailed view of the Nervos blockchain and its Type-script and lock-script.

Consensus Mechanism: Proof of Entropy Minima (PoEM)

The PoEM consensus mechanism is central to how Quai Network operates. It combines elements of traditional proof-of-work systems with innovative energy management strategies. PoEM incentivizes participants to contribute computational resources while ensuring that energy consumption is optimized and environmentally sustainable. This approach not only secures the network but also aligns with global efforts towards more sustainable blockchain practices. In summary, Quai Network's multichain structure—comprising Prime, Paxos, and Cyprus—works in tandem with its PoEM consensus mechanism to create a scalable, efficient, and environmentally conscious blockchain ecosystem. This design allows for high transaction throughput, stable asset management, and seamless cross-chain interactions, positioning Quai Network as a forward-thinking player in the cryptocurrency landscape.

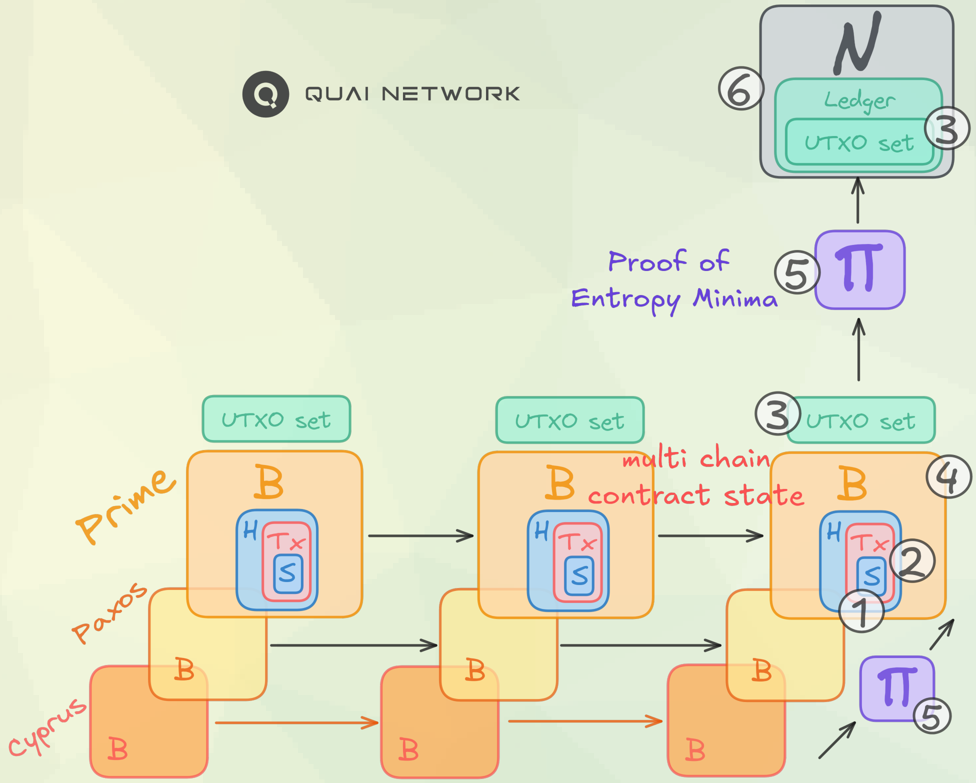


Figure 2.10: Simple Quai Network blockchain schematic.

TOPL

Topl is a specialized blockchain platform designed primarily to verify and track impact investments and sustainable practices, with a particular focus on ESG (Environmental, Social, and Governance) initiatives. The blockchain stands out for its unique approach to combining impact verification with blockchain technology. The core of Topl's architecture is built around its Proof of Learning consensus mechanism, which differs from traditional Proof of Work or Proof of Stake systems. This mechanism is designed to be energy-efficient while maintaining security and decentralization, aligning with the platform's sustainability goals. The platform implements a unique asset model that allows for the creation and tracking of both fungible and non-fungible assets, with special emphasis on "proof of impact" tokens. These tokens can represent various forms of positive impact, from carbon credits to fair trade certifications or sustainable sourcing verifications. Topl's smart contract system, known as Genus, is specifically designed to handle impact investing and ESG-focused transactions. It allows for the creation of complex investment arrangements while maintaining transparency and verifiability of impact claims.

Key architectural components include:

- The Bifrost Virtual Machine for smart contract execution
- Asset registry for tracking impact metrics
- Verification protocols for impact claims
- Cross-chain interoperability features
- Native support for impact-linked financial instruments

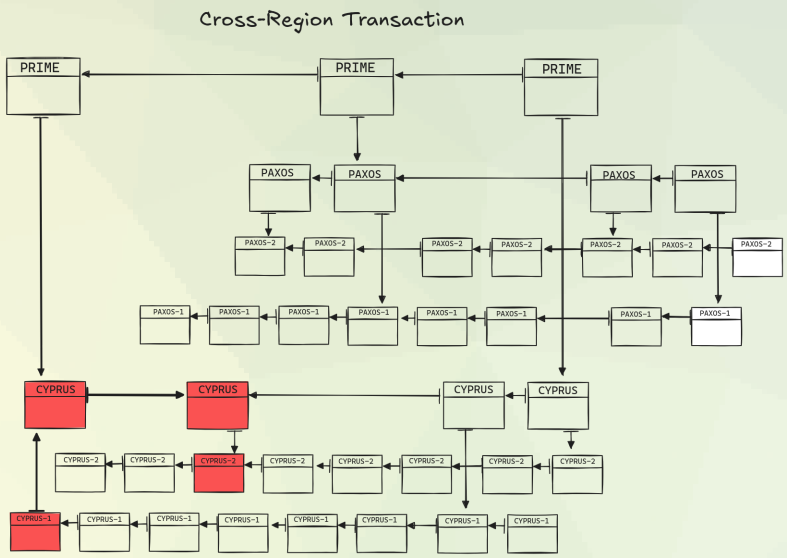


Figure 2.11: A detailed view of the Quai blockchain and its cross-region transaction.

Alephium

Alephium represents a novel approach to blockchain architecture, combining the benefits of UTXO-based systems with advanced smart contract capabilities. At its core, it employs a unique stateful UTXO model that bridges the gap between Bitcoin's UTXO system and Ethereum's account-based model. The Stateful UTXO Model:

Alephium introduces its custom virtual machine, Alphred, specifically tailored for executing smart contracts. This VM addresses several challenges faced by existing dApp platforms:

Smart Contract Implementation with Alphred VM (AVM)

The AVM serves as the execution environment for smart contracts, managing contract state access and modifications while enforcing security and resource constraints. It reads contract states from the State Merkle Tree, processes state transitions based on contract logic, and creates new UTXOs that reflect state updates. This process ensures deterministic execution while maintaining the blockchain's security properties.

When processing transactions, the AVM executes contract functions based on transaction inputs, managing gas consumption and resource limits. It ensures atomic execution of contract operations and handles rollbacks in case of failures. This sophisticated interaction between the three Merkle trees and the AVM creates a robust foundation for smart contract execution.

- Enhanced Security: Alphred provides a robust environment for smart contract execution, reducing vulnerabilities common in other platforms.
- Trustless P2P Transactions: It supports trustless peer-to-peer smart contract transactions, facilitating decentralized finance (DeFi) applications without requiring intermediaries.
- Dedicated Programming Language (Ralph): Smart contracts on Alephium are written in Ralph, a programming language inspired by Rust. Ralph simplifies the development process for creating efficient and secure smart contracts, making it particularly suitable for DeFi applications

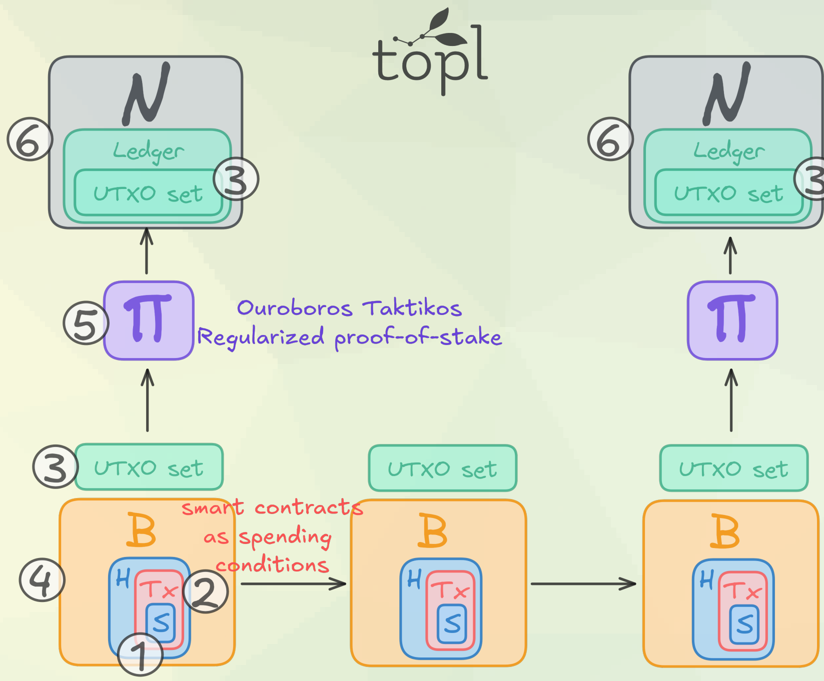


Figure 2.12: Simple Topl blockchain schematic.

Hathor

Hathor is a layer-1 protocol with a unique combination of high scalability and high decentralization. It has a novel architecture, using both directed acyclic graph (DAG) and blockchain technologies intertwined in its ledger.

Unlike traditional blockchains, transactions are not contained inside the blocks. Each transaction confirms 2 previous ones, forming a DAG of transactions. As in the case of Bitcoin, miners find new blocks which form a chain of blocks. Additionally, blocks also confirm 2 previous transactions, propagating the proof-of-work to the DAG.

This architecture allows Hathor to scale without giving up decentralization. Moreover, Hathor Network simplifies blockchain integration by presenting an easy-to-use interface for builders and developers. One does not need to be an expert blockchain developer to create a product using the network.

Hathor Network uses sha256 proof-of-work for its consensus mechanism, the same as Bitcoin. In fact, the network supports merged mining with the Bitcoin network, which allows miners to find blocks in both networks without extra work. This has allowed Hathor's hashrate to grow rapidly, giving more security to the network.

Among the main features of Hathor Network are:

- Highly scalable with no central coordinator or any single point of failure.
- Feeless and quick transactions.
- Merged Mining with Bitcoin has given Hathor one of the highest hashrates among all Proof-of-Work networks.
- Easy token creation, in just a few clicks.
- Smart Contracts written in Python, called Nano Contracts.

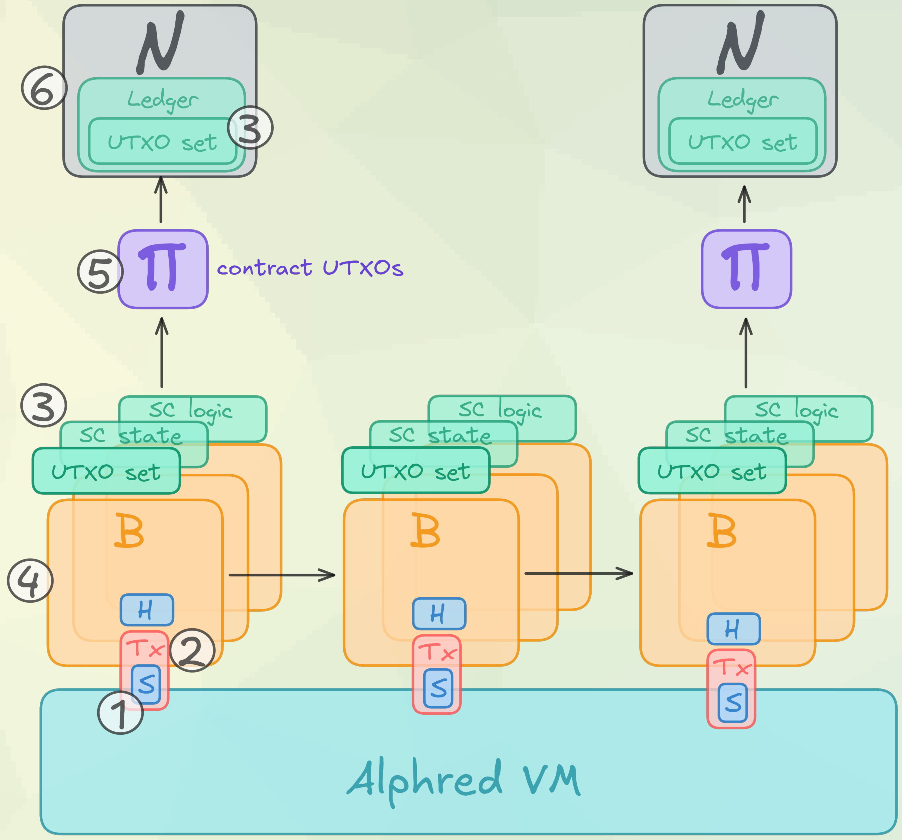


Figure 2.13: Simple Aelphium blockchain schematic.

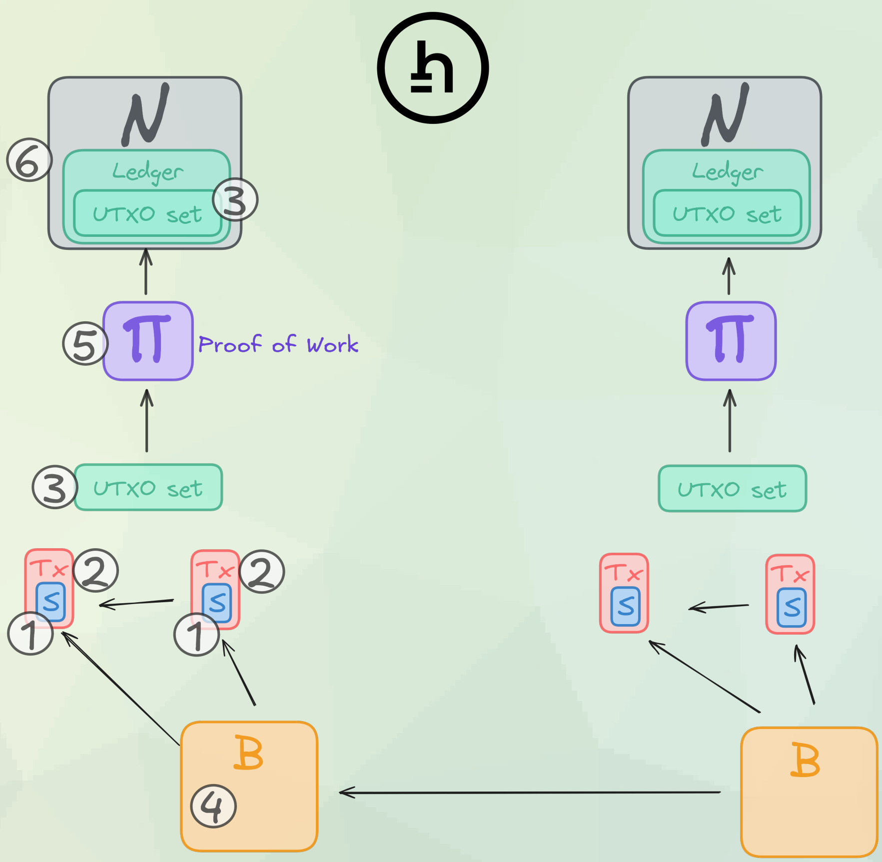


Figure 2.14: Simple Hathor blockchain schematic.

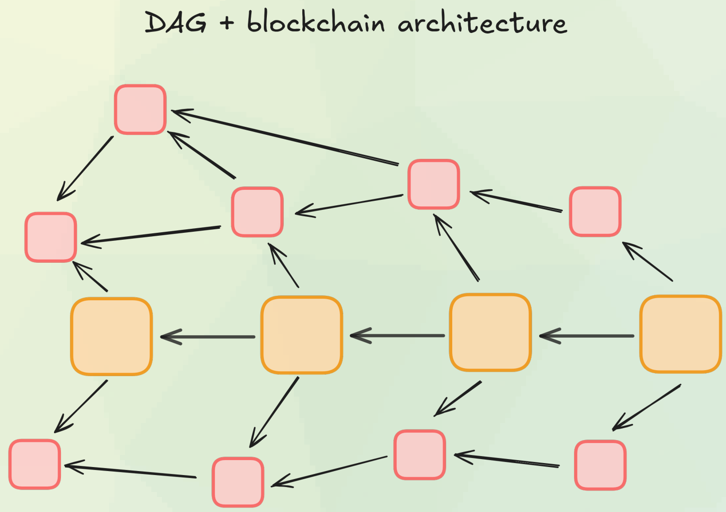


Figure 2.15: Simple Hathor blockchain schematic.

Chapter 3: Supplementary Material

The Account Model

The core focus of the handbook has been the UTXO model and its variants. However, the computing paradigm can be other design.

- How do we determine "who owns what?" in each blockchain model
- How a single transaction is handled in each blockchain model
- How multiple transactions are handled concurrently
- States of confusion: Origin of non-determinism

Figure 3.1: State transitions in both computing models, UTXO and Accounts.

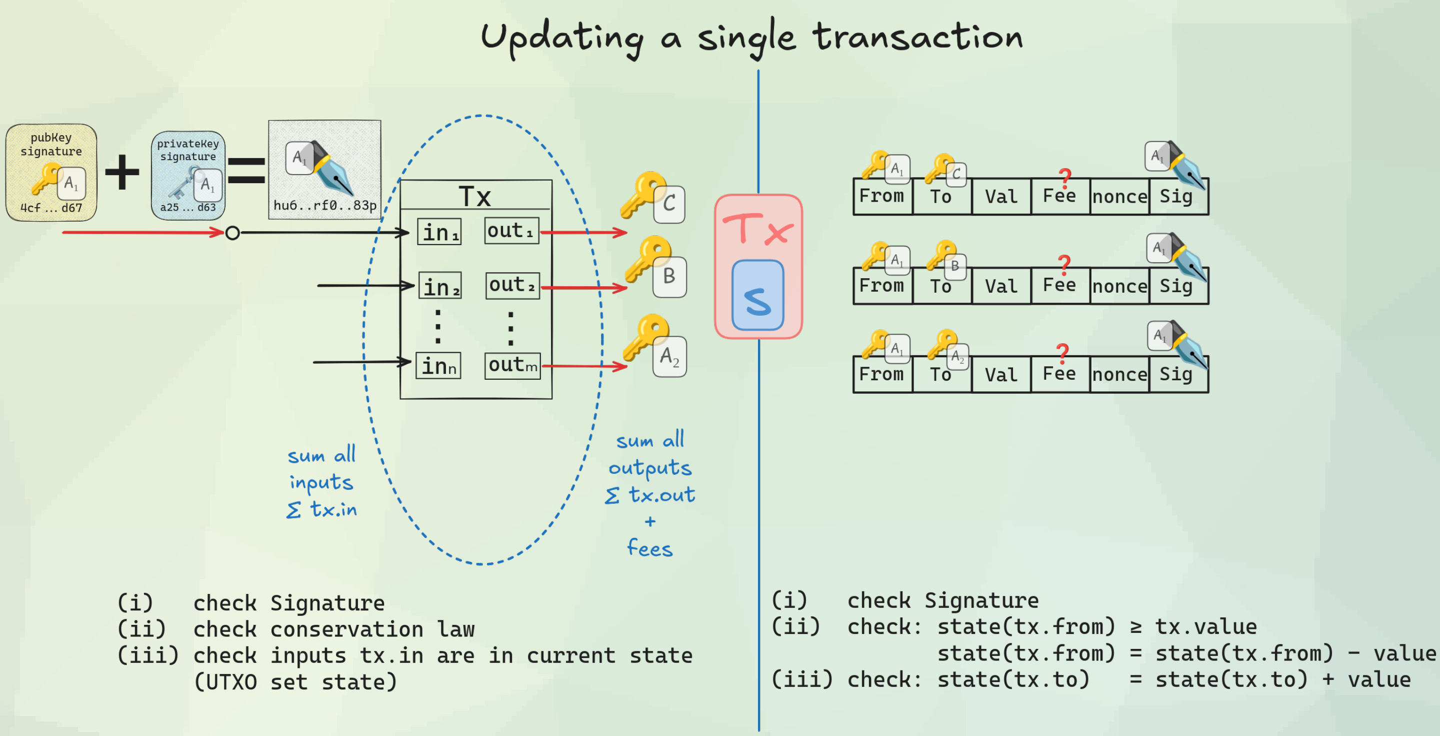


Figure 3.2: Single transaction comparison in both models.

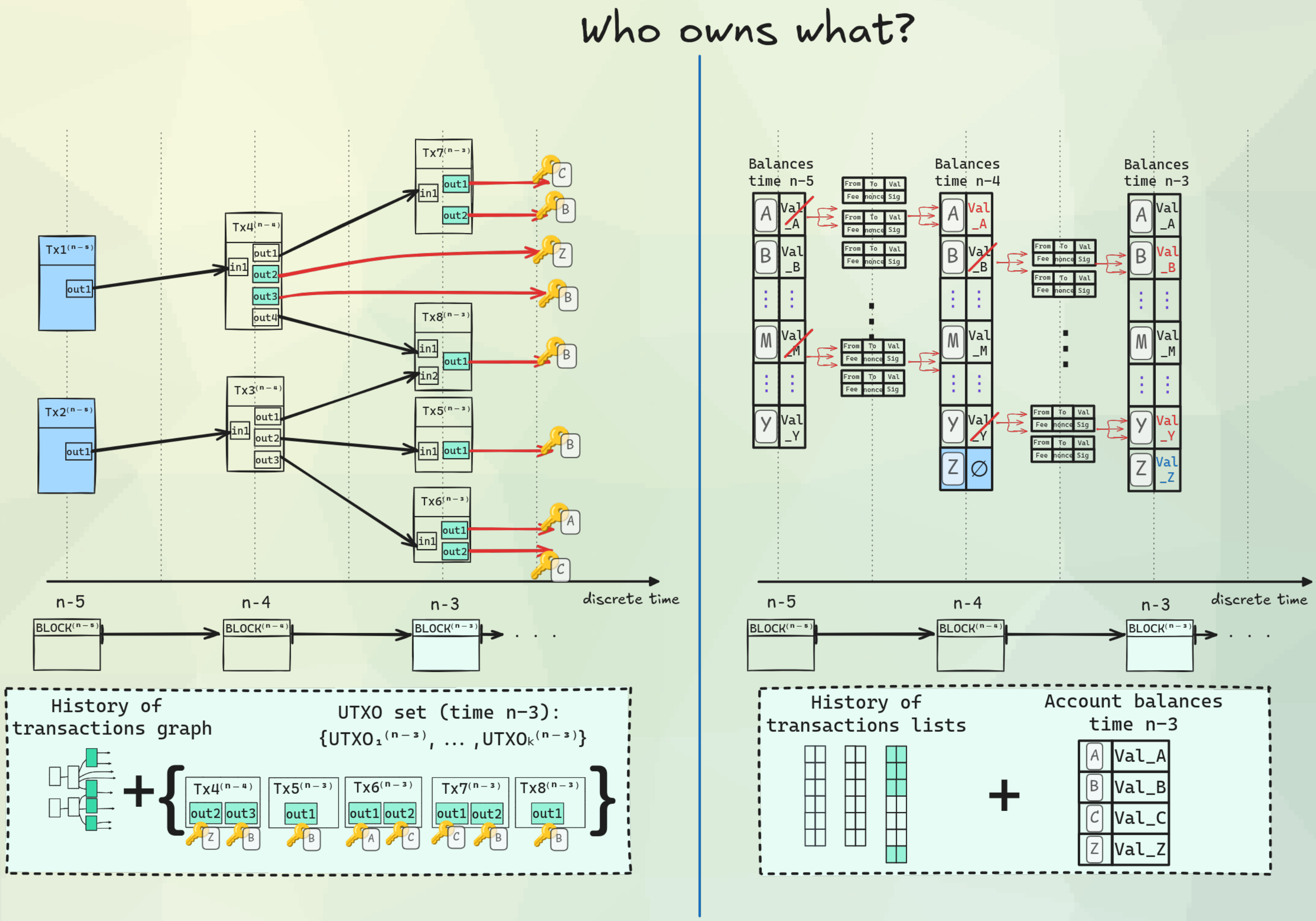


Figure 3.3: Time progression in both computing paradigms. A handful of transactions is shown for each model.