UNIVERSITY of NICOSIA

Session:

Ethereum and Programmable Blockchains

We are here

Session	Session Name	Professor
1	Introduction to Blockchain and Web3	George Giaglis
2	Bitcoin and Digital Money	Garrick Hileman
3	Ethereum and Programmable Blockchains	Apostolos Kourtis
4	Digital Wallet Security and Privacy	Charles Guillemet **(Ledger)**
5	Decentralized Finance (DeFi)	Lambis Dionysopoulos
6	Prediction Markets	Apostolos Kourtis & Lambis Dionysopoulos
7	The Geopolitics of Cryptocurrency	Garrick Hileman
8	Tokenization and Stablecoins	Lambis Dionysopoulos & Lauren Berta **(Ripple)**
Advanced Track & MSc Students		
9	NFTs and Digital Ownership	Punk6529 & Mohsen El-Sayed **(Ledger)**
10	Regulation and Policy of Digital Assets	Jeff Bandman
11	Advanced Topics in Web3	Apostolos Kourtis
12	The Future: Convergence of Blockchain, AI, and IoT	George Giaglis

Session Objectives

By the end of this session, you will be able to:

- Understand **Ethereum's purpose, architecture**, and **role** in the blockchain ecosystem
- Examine **Ethereum's historical development**, from its whitepaper to its current form
- Analyse Ethereum's key **technical components**, including **smart contracts** and the **Ethereum Virtual Machine (EVM)**
- Explain the transition from **Proof of Work (PoW)** to **Proof of Stake (PoS)** and its impact on **security, decentralization,** and **scalability**
- Discuss Ethereum's role in **dApps** and **tokenization**
- Compare Ethereum with other popular **programmable blockchains**

Agenda

1. Overview of Ethereum
2. Key Ethereum Concepts
3. Smart contracts, dApps & Tokenization
4. EVM-Compatible and Programmable Blockchains
5. Conclusions
6. Further Reading

Session: Ethereum and Programmable Blockchains

1. Overview of Ethereum

Ethereum: A decentralized world computer

Ethereum is a decentralized, open-source blockchain platform that enables smart contracts and decentralized applications (**dApps**).

- It can be viewed as a **general-purpose computing platform** that operates without central intermediaries. It instead relies on a vast network of participant computers.
- Launched in 2015, Ethereum was designed to address Bitcoin's limitations by enabling **programmable blockchain applications**.
- **Ether (ETH)** is Ethereum's native token, currently the **second-largest cryptocurrency** by market capitalization.

ethereum

Source: Ethereum Whitepaper | ethereum.org, Vitalik Buterin Created Ethereum Following World of Warcraft Debacle (businessinsider.com)

Ethereum's vision

Ethereum's vision

Programmable Money
Smart contracts that hold assets and execute transfers automatically based on pre-set rules

Decentralized Applications
dApp development on a shared infrastructure without the burden of launching separate blockchains

World Computer
A general-purpose decentralized computing platform for DAOs, identity, and other non-financial applications

Realised Vision (Feb 2026)

Over 5,000 dApps, $50B+ DeFi TVL, and a thriving ecosystem of NFTs and DAOs.

From programmable money to general-purpose computing

"Gavin can also be largely credited for the subtle change in vision from viewing Ethereum as a platform for building programmable money, with blockchain-based contracts that can hold digital assets and transfer them according to pre-set rules, to a general-purpose computing platform. This started with subtle changes in emphasis and terminology, and later this influence became stronger with the increasing emphasis on the "Web 3" ensemble, which saw Ethereum as being one piece of a suite of decentralized technologies, the other two being Whisper and Swarm."*

Vitalik Buterin,
Ethereum Founder

*Gavin Wood
Ethereum Co-founder, Polkadot Founder

Ethereum vs. Bitcoin (1/2)

Like Bitcoin, Ethereum is **open**, **borderless**, **censorship-resistant**, **immutable and transparent**

At the same time, it aims to overcome several of Bitcoin's limitations

Design Limitations of Bitcoin Targeted by Ethereum
Lack of Turing completeness – Bitcoin's scripting language is intentionally limited
Value-blindness – Bitcoin cannot easily execute based on external values
Lack of state – Bitcoin UTXOs don't store application state efficiently
Blockchain-blindness – Bitcoin scripts cannot interact with other blockchain data

Ethereum vs. Bitcoin (2/2)

Dimension	Bitcoin (BTC)	Ethereum (ETH)
Philosophy	Conservative stability; prioritizes predictability, security, and backward compatibility.	Rapid innovation; prioritizes functionality and experimentation, at the cost of backward compatibility.
Consensus Mechanism	Proof of Work (PoW); energy-intensive but tested, secured by physical hardware and hash power.	Proof of Stake (PoS); >99% energy reduction, secured through economic incentives and staking.
Scalability Approach	Layer 2–focused (e.g., Lightning Network); base layer ~7 TPS	Layer 2 but with Rollup-centric roadmap; base layer ~15-30 TPS
Primary Role	Store of value and censorship-resistant money.	World computer; platform for dApps, DeFi, NFTs, and programmable money.
Use Cases	Payments, savings, hedge against monetary debasement; minimal on-chain programmability.	Smart contracts, decentralized finance, tokenization, stablecoins, DAOs, and enterprise use cases.

Ethereum applications

Financial Applications:

- **Cryptocurrency transactions** and **DeFi (Decentralized Finance)** services.
- **Lending, borrowing, stablecoins and asset management** without traditional intermediaries.
- **Asset tokenization:** Converting real-world assets (e.g., real estate) into blockchain-based tokens.

Non-Financial Applications:

- **Data integrity:** Ensuring tamper-proof records for healthcare, government, and banking
- **Digital identity management** without reliance on centralized databases.
- **Decentralized voting systems** for online communities.
- **Decentralized governance models** for DAOs (Decentralized Autonomous Organizations).

The history of Ethereum (2013-2016)

2013 Vitalik Buterin publishes the Ethereum whitepaper (general-purpose smart-contract blockchain)

2014 Gavin Wood releases the Yellow Paper (formal spec). Ethereum ICO raises $18 million

2015 Ethereum mainnet launches (Frontier/genesis block). Ethereum introduces the "difficulty bomb" designed to eventually make Proof of Work (PoW) mining unsustainable and facilitate the transition to Proof of Stake (PoS)

2016 The DAO hack occurs, resulting in a hard fork: Ethereum and Ethereum Classic

The history of Ethereum (2017-2021)

2017 Byzantium fork improves efficiency and delays the difficulty bomb

2019 Constantinople upgrade: EIP-1234, preventing the blockchain from freezing before the PoS transition, and EIP-1014 to enable interactions with future addresses.

2020 Muir Glacier delays the difficulty bomb; Beacon Chain launches

2021 London upgrade implements EIP-1559 (fee burn); Altair upgrades the Beacon Chain (sync committees + updated penalties).

The history of Ethereum (2022-2025)

2022 Gray Glacier delays the difficulty bomb; Bellatrix + Paris complete **The Merge** (PoW ends, PoS begins)

2023 Shanghai + Capella ("Shapella") enable withdrawals of staked ETH and rewards

2024 Dencun (Cancun–Deneb) introduces EIP-4844 (proto-danksharding), lowering rollup data costs via blobs.

2025 Pectra (Prague–Electra) improves staking + account functionality. Fusaka (Fulu–Osaka) goes live, upgrading consensus + execution layers and focusing on scaling/security/UX improvements

Session: Ethereum and Programmable Blockchains

2. Key Ethereum Concepts

Ethereum accounts and the Ethereum Virtual Machine (EVM)

Ethereum accounts

Externally Owned Accounts (EOAs)

Controlled by private keys held by users.

Used to **send and receive Ether (ETH)** and interact with smart contracts.

Typically accessed through **wallet applications (e.g., MetaMask, Ledger).**

Contract Accounts (Smart Contracts)

Controlled by **program code** rather than private keys.

Cannot initiate transactions—**only execute functions when called** by an EOA.

Operate autonomously and store **logic & state** on the blockchain.

Ethereum Virtual Machine (EVM)

- **A global decentralized computing engine** that processes Ethereum transactions
- **The execution environment** where all smart contracts run
- Every Ethereum node **runs the EVM to maintain network consensus.**
- The EVM is **completely isolated**, meaning:
  - It **cannot access external networks** or read system files.
  - It **only interacts with on-chain data**, ensuring security.

Ether: the native token of Ethereum

ETH is Ethereum's native cryptocurrency, used to power transactions and smart contracts. The smallest unit of ETH is **Wei (1 ETH = 10^{18} Wei)**

Ethereum logo (freepngimg.com)

Capital Asset

ETH represents a share of the network. By staking ETH, validators secure the protocol, can process transactions and collect yield

Commodity

ETH acts as "Digital Oil". It is required to pay for gas fees (computation and storage). This base fee is permanently burned, reducing supply.

Store of Value

ETH is the key collateral of the DeFi ecosystem. It is trustless, censorship-resistant, and used to back stablecoins and secure loans.

EIP-1559 Economics

Since the London Hard Fork (2021), Ethereum's monetary policy has been dynamic. High network usage increases the burn rate, potentially making ETH deflationary

Issuance		Burn		Result
Block Rewards	-	**Base Fees**	=	**Net Supply Change**

Ethereum's PoS consensus mechanism

- Ethereum previously used **Proof of Work** (**PoW**), but transitioned to **Proof of Stake** (**PoS**) in 2022 via **The Merge**.
- PoS replaces miners with **validators**, who secure the network by staking ETH.
- PoS **reduces hardware requirements and lowers energy consumption**. However, it has also been criticized for **favoring those who hold more tokens**

How Proof of Stake Works

1. Validators stake a minimum of 32 ETH to participate in block validation.
2. A validator is randomly selected (proportionally to stake) to propose the next block.
3. Other validators verify the block and reach consensus.
4. Honest validators earn ETH rewards, while malicious ones risk slashing (lose their stake).
5. Finality is achieved when 2/3 of the active stake confirms a checkpoint.

Finality and security in PoS

- **Time structure: 12s slots** grouped into **32-slot epochs (~6.4 min)**; validators propose blocks and attest each slot.
- **Finality:** Attestations vote on **epoch boundaries (checkpoints)**. With **≥2/3 stake**, checkpoints become **justified** → **finalized** (typically within ~2 epochs).
- **Economic security:** Validators stake ETH; **misbehavior** (e.g., conflicting votes) trigger **slashing**, making finalized reorgs economically prohibitive.
- **Resilience:** If many validators go offline, **inactivity leak** lowers the effective threshold to restore finality; **fork-choice + attestations** limit reorgs and make censorship detectable/costly.

Proof of Stake vs. Proof of Work

Key PoS benefits

Energy Efficiency

Energy use is cut by more than **99.9%**

No mining hardware

Participation shifts from GPUs/ASICs to **staking** (solo or via pools)

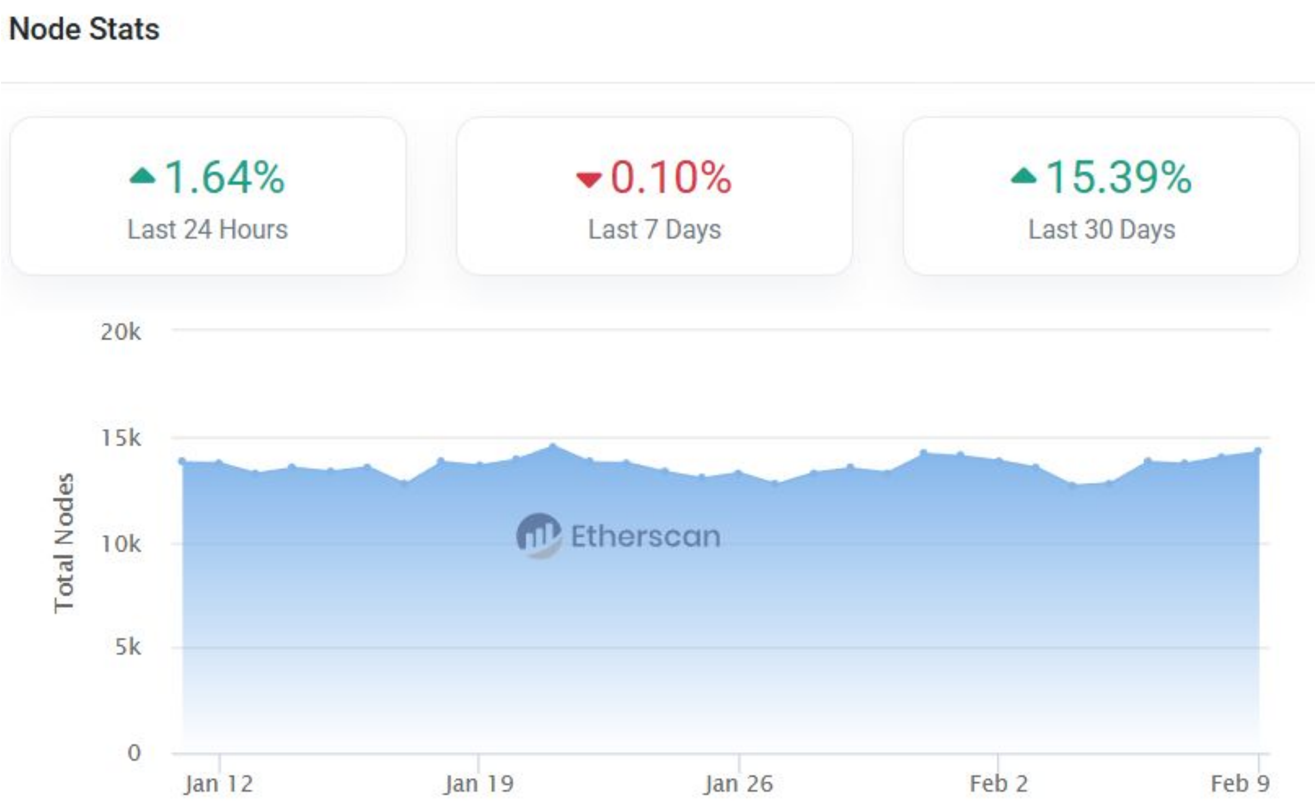
Supports scaling roadmap

Layer 2 scaling solutions like rollups and sharding.

Economic Security

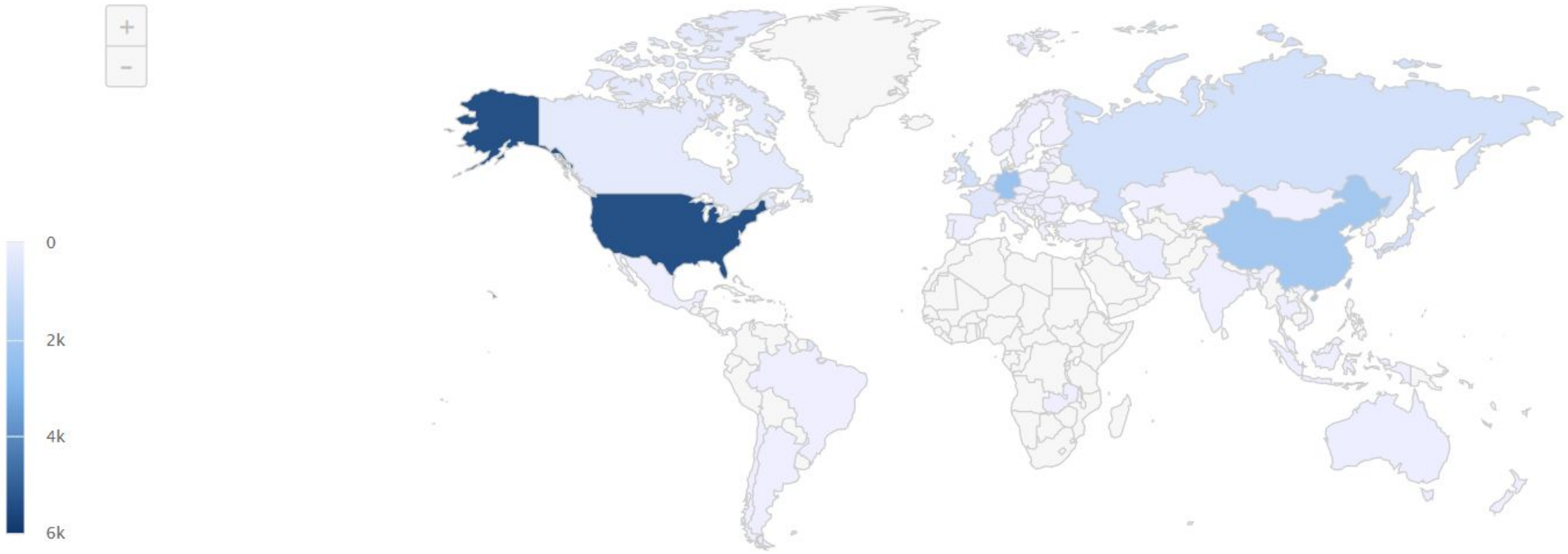
Attacks require large stake and are deterred by **slashing**; finalized reorgs are economically prohibitive.

Node statistics (February 2026)



source: https://etherscan.io/nodetracker

Geographical distribution of nodes (February 2026)



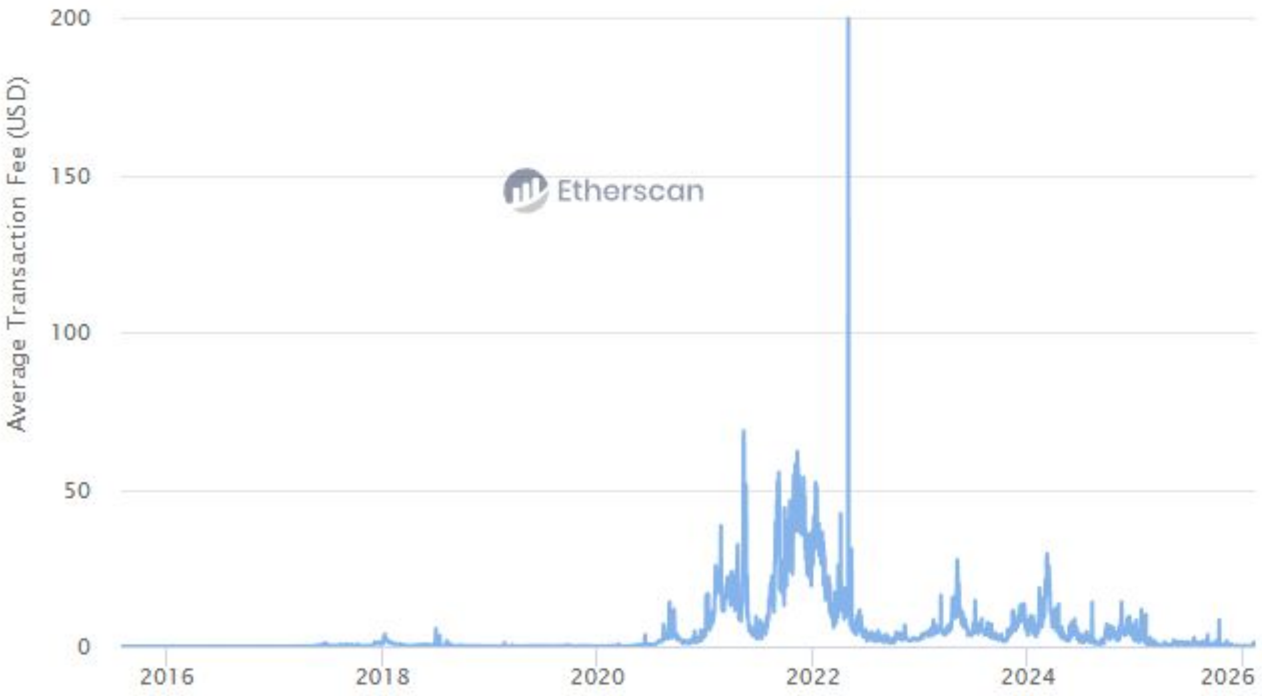
source: https://etherscan.io/nodetracker

The cost of computation: Understanding gas fees

- **Gas fees** are required to execute **transactions & smart contracts**, to prevent **spam attacks** and to compensate **validators (security)**
- **Users set a gas limit,** which determines how much computational work they are willing to pay for. If the gas limit is too low, the transaction fails
- During periods of high demand (e.g., DeFi and NFT booms), **Ethereum base-layer fees surged**, accelerating the adoption of **Layer 2** scaling solutions

Total Fee	=	**Gas Units**	×	**(Base Fee**	+	**Tip)**
ETH Paid		Work Required		Network Demand		Urgency

Transaction fees



Average Transaction Fee, 2015-2026

Source: https://etherscan.io/chart/avg-txfee-usd

The Blockchain Trilemma

Introduced by **Vitalik Buterin** in 2017, the **Blockchain Trilemma** highlights the trade-offs in blockchain design. Blockchain networks can generally prioritize two out of three key attributes:

- **Decentralization**, i.e., the extent of control and decision-making distributed across the network

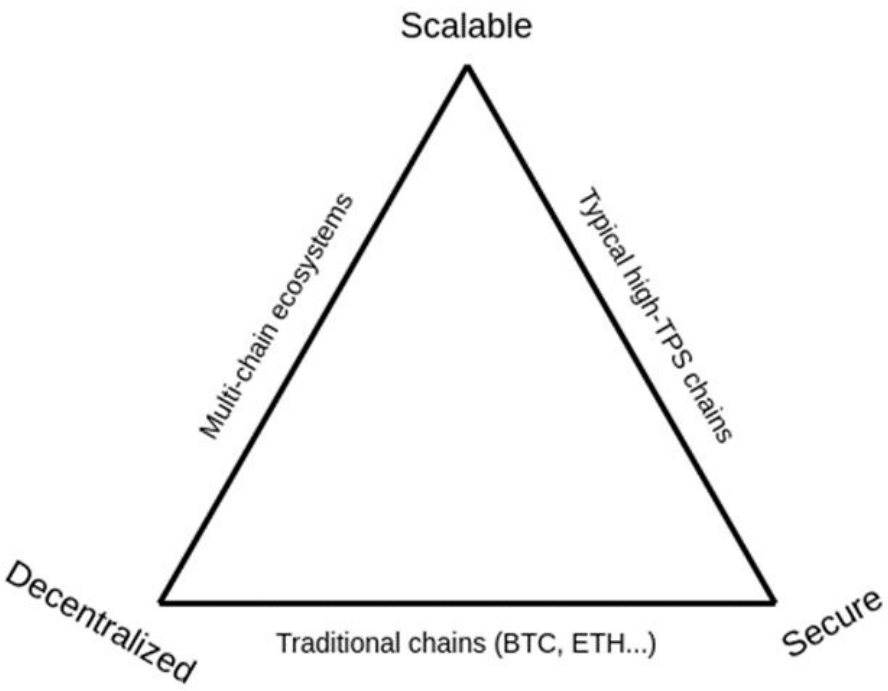
  However, achieving high decentralization can result in reduced operational efficiency due to the need for distributed consensus

- **Security**, i.e., mechanisms that safeguard the network from attacks and ensure the integrity and immutability of the data

  Yet, complex security protocols may diminish throughput and scalability

- **Scalability**, i.e., the capacity of a blockchain network to accommodate large volumes of transactions and operations

  Still, pursuing higher scalability may introduce potential security compromises or increased centralization



Source: https://vitalik.eth.limo/general/2021/04/07/sharding.html

The Layer 2 Revolution

Users & Apps

Interact with fast, cheap interfaces (Uniswap, Aave). Experience is seamless.

↓ **Transactions**

Layer 2 (Rollups)

Optimism | Arbitrum | Base

Execution Layer. Batches transactions and posts compressed data and state commitment to Layer 1.

↓ **Proofs & Data**

Ethereum L1

Settlement Layer. Provides ultimate security, data availability, and dispute resolution.

*Blockchain scalability approaches will be further covered in the “**Advanced Topics in Web3**” session*

Session: Ethereum and Programmable Blockchains

3. Smart Contracts, dApps & Tokenization

What are smart contracts?

Smart contracts are self-executing programs stored on a blockchain.

- **They run automated processes** on **Ethereum's Virtual Machine (EVM)**
- **They are immutable** but can incorporate **upgradeability**
- **They do not involve intermediaries**, reducing costs and inefficiencies.
- **They are not "smart" in an AI sense** – they do not "think" but follow **if-then logic.**
- **They are not inherently legal contracts** – they enforce agreements through code, not law.

How Smart Contracts Work

Trigger (Input)
User sends funds or data to the contract address.

→

Logic (Code)
"IF amount > 5 ETH AND date is today..."

→

Action (Output)
"THEN release the digital asset to the user."

Smart contracts (applications)

DeFi (Decentralized Finance)

Example: Automated Market Makers (AMMs) allow users to trade tokens automatically based on liquidity pools, replacing traditional order books

NFTs (Ownership)

Example: Contracts can signal royalty information, enabling marketplaces to automatically distribute a percentage of resale revenue to creators

DAOs (Governance)

Example: Voting results automatically trigger actions (e.g., moving funds from a treasury) if a proposal passes, removing human discretion.

Advantages and challenges of smart contracts

Advantages	Challenges
Efficiency: Automate contract execution, reducing the need for manual intervention and streamlining processes.	**Expertise**: Developing and understanding smart contracts requires a certain level of technical expertise, limiting accessibility for non-technical users.
Reduced intermediaries: Smart contracts reduce the need for traditional intermediaries, streamlining processes and reducing costs associated with traditional contract enforcement.	**Dependence on Inputs**: Smart contracts may rely on external data sources (oracles), introducing additional trust assumptions and potential failure points.
Transparency: Transactions are recorded on a public blockchain, providing a transparent and auditable history.	**Scalability**: Scalability challenges can arise as blockchain networks grow, impacting transaction speed and cost.
Security: On-chain execution is hard to alter once confirmed, but overall security depends on code and key management.	**Security**: Vulnerabilities in code may lead to exploits, emphasizing the need for rigorous auditing and testing.
Trustless transactions: Enables parties to transact without relying on trust, as the contract's execution is enforced by code.	**Legal Recognition**: The legal status of smart contracts varies globally, and their enforceability may face challenges.
Immutability: Once deployed, the code of a smart contract is immutable, upgrades typically require explicit upgrade patterns and governance.	**Immutability**: Code is immutable, meaning errors or vulnerabilities cannot be (easily) rectified once deployed.
Programmability: Flexible and programmable, allowing for complex conditional statements and diverse applications.	**Evolving Technology**: Ongoing technological advancements may introduce uncertainties and necessitate updates to smart contract standards.

Tokens on Ethereum

ERC-20

Fungible

Tokens are identical and interchangeable, like fiat.

Primary Use Cases

Stablecoins, Governance Tokens, ICOs

ERC-721

Non-Fungible

Unique tokens with individual IDs and metadata, used for ownership of distinct items.

Primary Use Cases

Digital Art, Identity, Real Estate Deeds

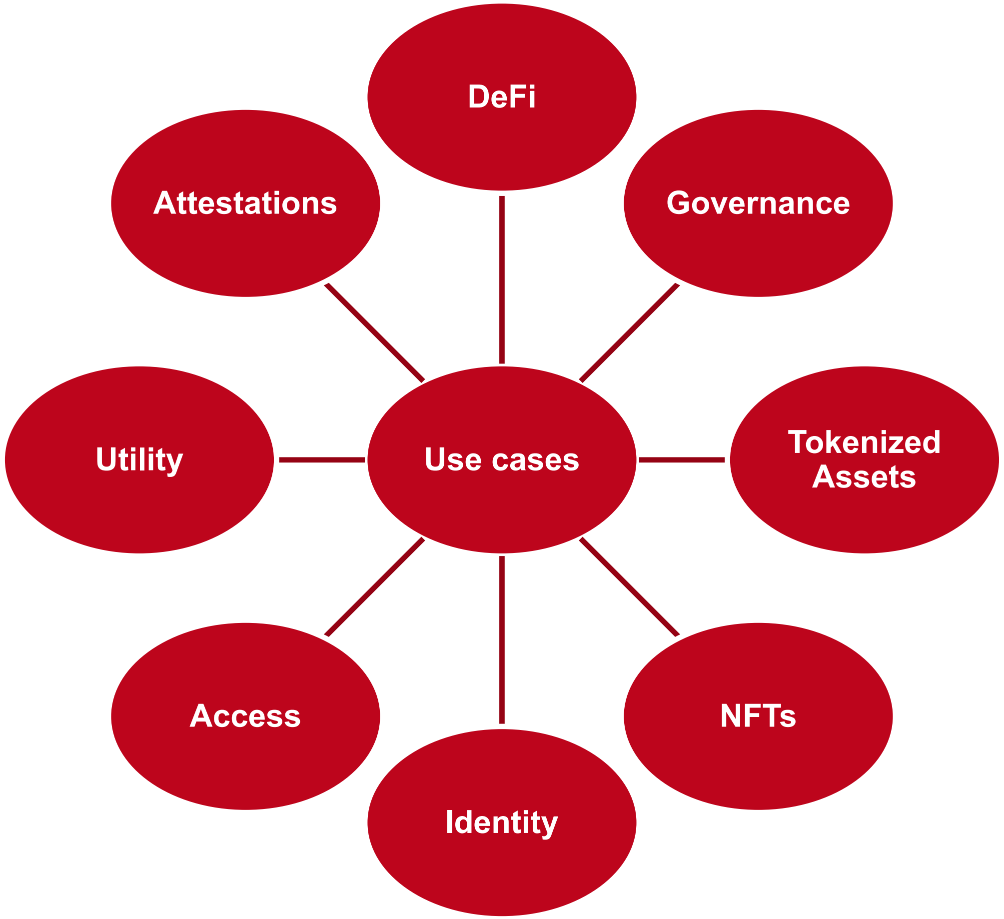
Real World Assets (RWA)

These standards also enable physical assets on-chain — offering 24/7 liquidity, fractional ownership, and transparent settlement.

Key Examples

Bonds, Real Estate, Gold

Tokens on Ethereum: Beyond money



Decentralized Applications (dAppps)

Smart contracts enable developers to create **decentralized** **applications (dApps).**
They are like normal applications, but the key difference is they run on a blockchain

dApps are the cornerstone of Web3:

- **Open-source:** No single entity controls the code
- **Decentralized storage:** Option to have critical data on-chain, and not in private databases
- **Incentivized via tokens:** Uses crypto-economics to reward users & developers

Tokens provide the on-chain assets (value, access, governance) dApps use.

dApps inherit most smart-contract tradeoffs (transparency, automation, reduced intermediaries) **but add app-layer dependencies** (frontends, wallets, oracles, bridges, governance), which can increase security and UX risk

Major Categories

- DeFi
- NFTs
- DAOs
- Gaming
- Social

Sources: Decrypt and District0x

Session: Ethereum and Programmable Blockchains

4. EVM-Compatible and Programmable Blockchains

The EVM-Compatible Ecosystem

Chains that are "**EVM-compatible**" allow developers to deploy Ethereum smart contracts with **minimal changes**, tapping into Ethereum's rich developer ecosystem

- **BNB Smart Chain (BSC):** Uses a proof-of-stake-authority model with a limited validator set. It is known for close integration with the Binance ecosystem.
- **Avalanche:** Layer-1 platform built on Avalanche consensus and a three-chain architecture (X-Chain, C-Chain, P-Chain). C-Chain is EVM-compatible. Supports custom subnets for application-specific blockchains.
- **Polygon:** EVM-compatible ecosystem offering multiple scaling approaches. Its Polygon PoS network operates as an EVM-compatible sidechain that periodically checkpoints (commits) state to Ethereum.

Source: https://www.bnbchain.org/en/blog/what-are-evm-compatible-blockchains, https://messari.io/copilot/share/evm-compatibility-explained-b7fd9ee8-9b11-484f-8612-892c6ec3bbf2

Other programmable blockchains

- **Cardano:** Uses the Ouroboros proof-of-stake protocol developed through peer-reviewed research. Smart contracts run in the Plutus environment. It provides EVM compatibility via sidechains such as Milkomeda. Cardano emphasizes research-based design and formal verification.

- **Solana:** Combines Proof-of-History with proof-of-stake. Smart contracts (“programs”, commonly written in Rust) are executed by the Solana runtime. It prioritizes high throughput and low-latency processing.

- **Polkadot:** Built around a relay chain that provides shared security to parachains. Parachains can run smart contracts or application-specific logic. It centers around interoperability and cross-chain coordination.

- **Tezos:** Enables smart contracts and on-chain governance with protocol upgrades that avoid hard forks. Contracts compile to Michelson, supporting formal verification. It is known for governance-driven evolution and verifiable execution.

Other programmable blockchains

Platform	Ethereum	Cardano	Solana	Polkadot	Tezos
Type	General-purpose smart contract L1	General-purpose smart contract L1	High-performance smart contract L1	Relay chain + parachains (multi-chain platform)	General-purpose smart contract L1
Execution model	**EVM** (Solidity contracts)	**Plutus** on eUTXO	**Programs** on Solana runtime (often Rust)	**Parachains** (smart-contract or app-specific chains)	**Michelson** (verification-friendly)
Consensus	Proof of Stake	Ouroboros PoS	PoS + Proof of History	NPoS (relay chain) + shared security	Liquid PoS + on-chain governance
Strengths	Biggest ecosystem + tooling; composability	Research/formal methods culture	Throughput + low latency UX	Interoperability + app-specific chains	Governance-driven upgrades; formal verification focus
Typical tradeoffs	L1 fees at peak; scaling via L2	Smaller ecosystem; different dev model	Different model/tooling; high hardware requirements	More complex architecture	Smaller ecosystem than Ethereum; different tooling
Best fit	DeFi, NFTs, DAOs, standards	High-assurance apps; eUTXO designs	Consumer apps, games, high-frequency UX	Cross-chain systems; app-specific chains	Upgrade-heavy protocols; verifiable contracts

Towards a multi-chain/omni-chain future

Multi-chain refers to a world where many blockchains coexist and apps may deploy on several networks (often with different goals like cost, speed, privacy, or governance)

Omnichain describes an app/user experience that works across chains “as one,” enabled by interoperability tools that move **messages** and **assets** between networks

- **EVM compatibility** allows the same smart contracts to run on multiple chains
- **Cross-chain messaging & bridges** enable chains to exchange messages and transfer assets.
- **Sidechains and Layer 2 Solutions** improve scalability by processing transactions outside the main chain

Session: Ethereum and Programmable Blockchains

5. Conclusions

Summary

- **Ethereum** is a decentralized blockchain platform for running **smart contracts** and building **decentralized applications (dApps),** described as a **“world computer”**
- **Smart contracts** are self-executing code that automates rules/agreements without intermediaries and run on the **Ethereum Virtual Machine (EVM)**.
- **ETH (Ether)** is the native asset used to pay **gas fees**, **stake** in Ethereum’s P**roof-of-Stake** system, and serve as **collateral in DeFi**.
- **Ecosystem impact**: Ethereum powers major categories like **DeFi, NFTs**, and **DAOs**, enabling open finance, digital ownership, and on-chain governance.
- **Key challenge**: **scalability**. High demand can cause congestion and high gas fees, so Ethereum relies on Layer 2 scaling and other upgrades to increase throughput.

*Ethereum scalability approaches will be further covered in the “**Advanced Topics in Web3**” session*

Session: Ethereum and Programmable Blockchains

6. Further Reading

Further Reading

- Ethereum Whitepaper
- Antonopoulos, A.M. and Wood, G., 2018. Mastering Ethereum: building smart contracts and dApps. O'Reilly Media.
- The Ethereum DeFi ecosystem
- V. Buterin
- The Merge
- The Blockchain Trilemma
- Smart Contracts
- V. Buterin's Website
- The importance of Low-risk DeFi

Questions?

Contact Us:

Twitter: @mscdigital
Course Support:
digitalcurrency@unic.ac.cy
IT & Live Session Support: dl.it@unic.ac.cy