SMART CONTRACTS

DEFINITION

RICARDIAN CONTRACTS

OVERVIEW OF THE PRESENTATION

- Introduction to Smart Contracts
- Key Features of Smart Contracts
- Benefits & Real-World Applications

OVERVIEW OF THE PRESENTATION

- Introduction to Ricardian Contracts
- Key Properties of Ricardian Contracts
- Implementation Process of Ricardian Contracts
- Bowtie Model – Linking Law & Accounting
- Ricardian Contracts vs. Smart Contracts

Introduction to Smart Contracts

- A **smart contract** is a **secure and unstoppable** computer program that represents an agreement and executes automatically.
- It is written in a **machine-readable language** and encodes business logic between parties.
- Execution is **triggered by predefined conditions**, eliminating the need for intermediaries.

Key Features of Smart Contracts

1. Automatic Execution:

- ➢ Smart contracts run autonomously and execute predefined actions without requiring manual intervention.
- ➢ Once the specified conditions are met (e.g., payment received, verification completed), the contract executes automatically.

Key Features of Smart Contracts

2. Self-Enforcing:

- Unlike traditional contracts, which rely on legal enforcement, smart contracts are self-enforcing.
- The code ensures that the agreement is fulfilled exactly as programmed, removing the need for intermediaries.

Key Features of Smart Contracts

3. Trustless Transactions:

- Participants do not need to trust each other because execution is **guaranteed by blockchain consensus** mechanisms.
- The contract removes the risk of manipulation, fraud, or breaches by ensuring **decentralized enforcement**.

Key Features of Smart Contracts

4. Transparency & Auditability:

- ➢ All transactions and contract terms are recorded on a **public ledger**, allowing anyone to verify the contract's execution.
- ➢ This **reduces disputes, builds trust, and ensures accountability** in financial and business agreements.

Key Features of Smart Contracts

5. Interoperability & Integration:

- All transactions and contract terms are recorded on a **public ledger**, allowing anyone to verify the contract's execution.
- This **reduces disputes, builds trust, and ensures accountability** in financial and business agreements.

Benefits & Real-World Applications

•**Efficiency & Speed** – Automates processes, reducing time delays and errors.

•**Cost Reduction** – Eliminates intermediaries, reducing transaction costs.

•**Decentralization & Fairness** – Ensures fair execution without external influence.

•**Applications** – Used in finance (DeFi), supply chains, insurance, legal agreements, and more.

Introduction to Ricardian Contracts

Origin & Background:

- Proposed by **Ian Grigg** in the late 1990s in the paper *Financial Cryptography in 7 Layers*.
- Initially used in **Ricardo**, a bond trading and payment system.
- Aims to create contracts **understandable by both courts of law and computer systems**.

Introduction to Ricardian Contracts

Purpose:

- Addresses the challenge of **issuing value over the internet**.
- Ensures contracts are **legally binding** while remaining **machine-readable**.
- Identifies the **issuer and captures all contractual terms** in a structured document.

Key Properties of Ricardian Contracts

- **Legally Recognizable** – Designed to be **readable and enforceable** by courts.
- **Machine-Readable** – Contains **structured data tags** for software processing.
- **Digitally Signed** – Signed by the **issuer’s private key**, ensuring authenticity.

Key Properties of Ricardian Contracts

- **Unique & Secure Identifier** – Uses **hashing** to create a distinct fingerprint of the contract.
- **Issuer & Holder Roles** – Clearly defines **who issues** the contract and **who holds** the rights.
- **Integration with Accounting Systems** – Helps businesses **track transactions securely**.

Implementation Process of Ricardian Contracts

- A **single document** contains:
- Legal prose (human-readable)
- Machine-readable tags (software-parsable)
- The **contract document is digitally signed** by the issuer.
- The document is **hashed using a message digest function**, generating a unique **identifier hash**.
- This hash is used to **link every transaction** performed under the contract.

Bowtie Model – Linking Law & Accounting

World of Law (Left Side)

- The **legal contract document** originates here.
- Written in **legal prose** with **machine-readable elements**.
- The document is **hashed to generate an identifier**.

World of Accountancy (Right Side)

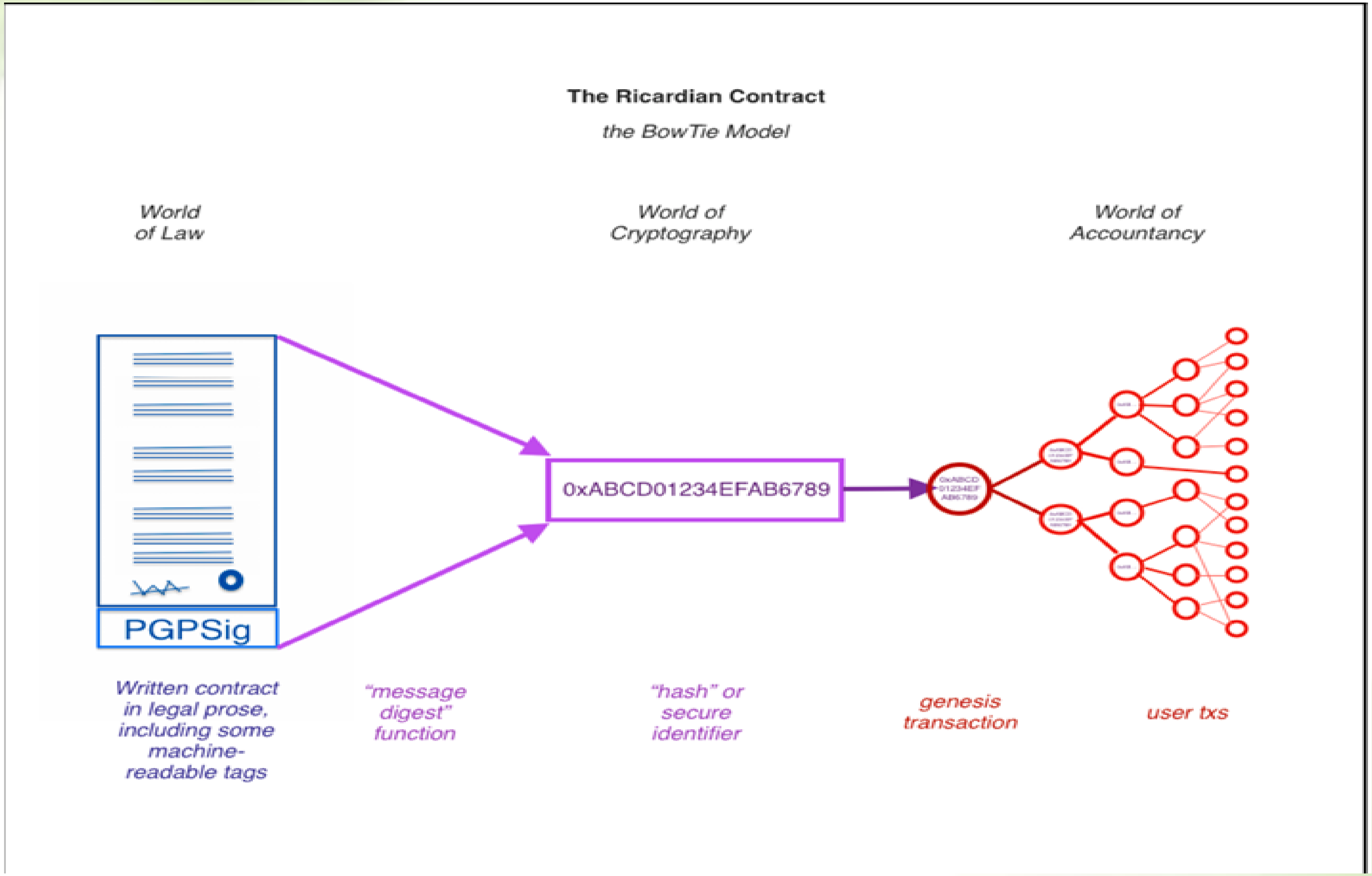
- The **identifier hash is used in transactions** as a reference.
- Represents **trading, accounting, and financial systems**.
- Ensures that every transaction is **securely linked to the original contract**.

Bowtie Model – Linking Law & Accounting

Genesis Transaction

- The **first transaction** includes the contract’s identifier hash.
- All subsequent transactions use this hash, creating a **secure and verifiable link**.

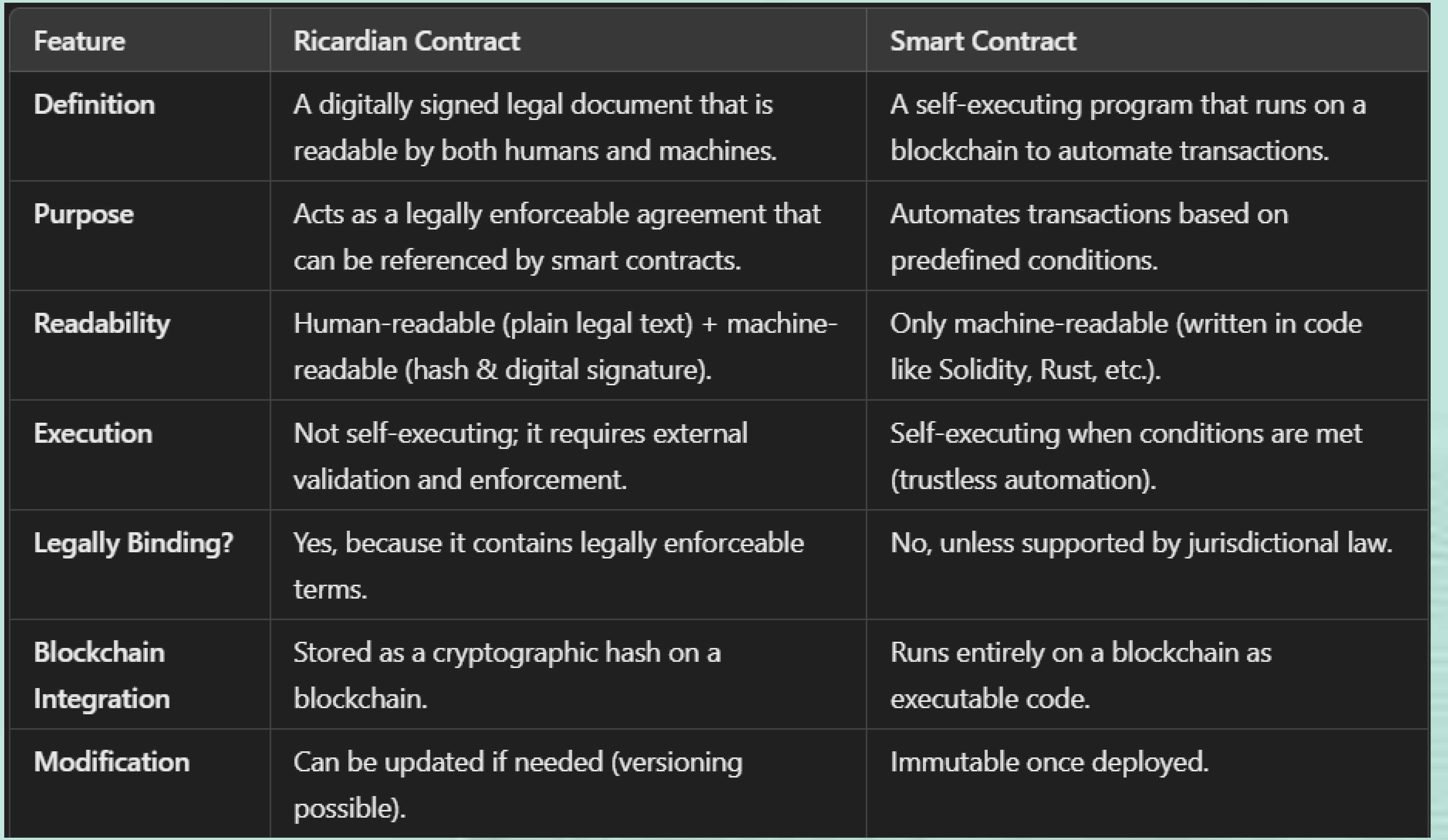
Bowtie Model – Linking Law & Accounting



Introduction to Ricardian Contracts vs. Smart Contracts

- **Smart Contracts** focus on execution and automation without legal documentation.
- **Ricardian Contracts** emphasize **semantic richness**, combining legal prose with machine-readable data.
- Introduced by **Ian Grigg**, Ricardian contracts aim to be **both legally binding and computer-readable**.

Key Differences Between Ricardian and Smart Contracts



Feature	Ricardian Contract	Smart Contract
Definition	A digitally signed legal document that is readable by both humans and machines.	A self-executing program that runs on a blockchain to automate transactions.
Purpose	Acts as a legally enforceable agreement that can be referenced by smart contracts.	Automates transactions based on predefined conditions.
Readability	Human-readable (plain legal text) + machine-readable (hash & digital signature).	Only machine-readable (written in code like Solidity, Rust, etc.).
Execution	Not self-executing; it requires external validation and enforcement.	Self-executing when conditions are met (trustless automation).
Legally Binding?	Yes, because it contains legally enforceable terms.	No, unless supported by jurisdictional law.
Blockchain Integration	Stored as a cryptographic hash on a blockchain.	Runs entirely on a blockchain as executable code.
Modification	Can be updated if needed (versioning possible).	Immutable once deployed.

Semantics in Contracts – Operational vs. Denotational

Operational Semantics

- Defines **how a contract executes step by step** in a computational system.
- Ensures that the **contract runs correctly**, following predefined rules.
- Critical for **smart contracts**, where automated execution is essential.
- Used in **blockchain environments** where transactions must be **deterministic and predictable**.

Denotational Semantics (Real-World Meaning Focused)

- Describes **what a contract means in the legal and business context**.
- Ensures that contract terms are **understandable to humans**, including courts and regulatory bodies.
- Helps in **bridging the gap between legal agreements and automated execution**.
- Found in **Ricardian contracts**, which encode both legal prose and machine-readable elements.

Ricardian Contracts – Legal Semantics & Human Readability

- Designed to be **legally enforceable** and **understandable by courts**.
- Contain **natural language text** along with machine-readable elements.
- Help in **compliance, legal documentation, and regulatory acceptance**.
- Used in **financial agreements, legal documents, and business contracts**.

Smart Contracts – Performance-Oriented Execution

- **Fully automated**, executing actions based on predefined conditions.
- Do not require **human intervention or third-party enforcement**.
- **Immutable** once deployed, ensuring **trustless execution** on blockchain networks.
- Used in **decentralized finance (DeFi), automated transactions, and self-executing contracts**.

THANK YOU!